המדריך למתלמדים בפסקל

מאת: יגאל גרמן
ואגמי יוחאי.

המדריך למתחיל בפסקל

פרק 1: מבוא **4**
מהי שפת תיכנות 4
מהו תכנות 4
פסקל 4
ההיסטוריה של פסקל 4
באג 5
סביבת התכנות 5
תוכנית ראשונה 7
אלוגריתמיק: 8
פרק 2: עקרונות בסיסים בפסקל **11**
WRITELN , WRITE 11
תוכנית שניה 11
משתנים בפסקל 12
הערות בתוכנית (תיעוד) 14
פעולות על מספרים 15
מספרים בינאריים 15
פונקציות מתמטיות ופעולות מיוחדות 17
מספרים אקראיים 17
READLN 18
תרגילים לסיכום פרק 2 19
פתרונות 20
פרק 3: לוגיקה (תנאים) **22**
תנאים בפסקל (תרגול מילולי-לוגי) 22
IF 24
תוכנית V30 24
טבלת מעקב 26
בלוקים 26
תוכנית V3.1 27
שדה עריכה 27
הוראה מקוננת 28
תוכנית מתוקנת 28
משתנים בוליאנים 30
סדר פעולות לוגיות : 30
תרגילים לחזרה 30
פתרונות לתרגילי החזרה 31
פרק 4: לולאות **35**
FOR 35
V30תוכנית 36
האפשרות DOWNTO 37
ספירה 38
מכפלה 38
מקסימום 39
מינימום 40
השוואה לסיכום (כל הנושאים) 40
REPEAT 41
דגל 43
שדה קליטה 43

תרגילים 45
פתרונות 45
לולאה אינסופית 45
לולאות מקוננות 46
תרגילים לסיכום 47
פתרונות 47
פרק 5: טיפוסי נתונים 49
קבוצות 49
סקלרים 50
טיפוסים (TYPES) 50
רשומות 50
WITH 51
מערכים ARRAYS 52
מערך דו ממדי 53
אלכסונים במערכים דו-מימדיים 55
STRING 56
תרגילים לסיכום 62
פתרונות 63
פרק 6: פונקציות ופרצדורות 65
פונקציות 65
משתנים מקומיים 66
משתנה ערך לעומת משתנה יחס 66
פרוצדורה 68
פונקציה לעומת פרצדורה: 68
העברת פרמטרים ללא הגדרה סוג 68
העברת מחרוזות ומערכים כפרמטרים 68
פרק 7: נספחים 70
שימוש בקבצים חיצונים 70
שימוש במדפסת 70
ספריית CRT 70
שגיאות 73
כלים נוספים 74
פרק 8: תרגילי סיכום לספר 76
סוף דבר 79

פרק 1: מבוא

מהי שפת תיכנות

שפת תכנות- היא אוסף של חוקים תחביריים (Syntax) וסמנטיים (Semantic) שבאמצעותם ניתן להגדיר למחשב באופן מפורט את הפעולות שעליו לבצע במצבים שונים, על סוגי קלט שונים.
המושג שפת מחשב (Language Computer), הוא מושג רחב מאשר שפת תכנות (Programming Language) ולכן השימוש בו נפוץ יותר.

ישנם שלושה סוגי רמות של שפות תיכנות:

1. **שפת מכונה**- זוהי השפה הבסיסית ביותר אשר מורכבת מהמספרים הבנארים 1 ו-0.
2. **שפת סף**-זוהי שפה אשר ברורה יותר לאדם ומורכבת ממילים אשר מרמזות על הפעולה שהפקודה מבצעת לדוגמא באסמבלר (שפת תיכנות) הפקודה mov שמצעת העתקה.
3. **שפה עלית**- זוהי שפה שדומה לשפה אנושית לדוגמה בפסקל (מה שנתמקד במדריך).

הערה-אנו מציעים למי שאינו למד תכנות להתחיל מפסקל כיוון שזוהי שפה אשר נותנת את הכלים הבסיסיים למתכנת המתחיל ויותר. כמו כן תראו שפסקל זוהי תוכנה מאוד מסודרת מה שגם דרוש בתכנות.דבר נוסף שהפקודות בשפת פסקל הם כמו בשפה האנגלית לכן זה מאוד ברור .

מהו תכנות

תכנות מאפשר להשתמש בכלים ובעצם בכוח של שפת התכנות על מנת ליצור תוכניות, שבעזרתם ניתן לפתור בעיות שונות כגון: חישובים מדעיים, סימולציות וכו ...
תפקידו של המתכנת הוא ליצור תוכנות המבצעות את הנדרש בשימוש מינימלי של הזיכרון מחשב (אולם היום עקב כוח המחשב לא נדרשת רמת צמצום כמו בעבר).
תיראו בהמשך ,שלתכנות יש חוקים משלו ועליכם לעקוב אחריהם על מנת ללמוד כראוי כל שפת תכנות בעתיד.

פסקל

Pascal - היא שפת תכנות כללית, שפותחה על-ידי ניקלאוס וירת, וקרויה על-שמו של המתמטיקאי והפילוסוף בלז פסקל. זוהי שפה עלית שמתכנתים בה בסביבת דוס וישנו פן של פסקל שנקרא דלפי שמתכנתים בשפה זו בסביבת וינדוס.
ישנם מהדרים רבים בפסקל ובכל מיני גרסאות אנו נשתמש ב-Turbo Pascal.

ההיסטוריה של פסקל

Pascal היא שפת תכנות כללית, שפותחה על-ידי ניקלאוס וירת, וקרויה על-שמו של המתמטיקאי והפילוסוף בלז פסקל.

השפה היא פרוצדורלית וכוללת בקרת זרימה ומבני נתונים כמו בכל שפה עילית מודרנית, ואף הרחבה לתכנות מונחה עצמים.

השפה דומה מבחינת המבנה והיכולות שלה לשפת C, אף שהיא מוגבלת בכוונה בכמה היבטים, כך שהיא נותנת סביבת פיתוח פחות מורכבת ויותר מוגנת. בשל תפיסותיה היא שימשה כשפה לימודית באוניברסיטות פרק זמן לא מבוטל.

חברת בורלנד פיתחה בסביבת DOS סביבת פיתוח ומהדר ששמו היה טורבו פסקל. אבל כאשר פותחה מערכת ההפעלה Windows, טורבו פסקל ירד ממעמדו. יש לציין שעד היום שפה זאת אהודה בחברה זו, ולא במקרה סביבת הפיתוח של דלפי משתמשת בפסקל כשפת הקוד הפנימית שלה.

שפת פסקל תוכננה במקור כדי שהיא תהווה מקור למידה לתחביר, לתכנות, פיתוח חשיבה וכבסיס ללימוד שפות אחרות - היות והיא שפה בסיסית וקלה ללימוד.

בראשית שנות התשעים היה "טורבו פסקל" של בורלנד מהדר הפסקל המוביל. הייתה לו קהילת מפתחים גדולה, וכאשר בשנת 1994 היה כבר ברור שלא יימשך הפיתוח שלו, מתכנתים רבים נשארו ללא פתרון. אחדים מהם ייזמו פרוייקטים ליצירת מהדר חדש, אחד מהם, Pascal Free נולד בשנת 1993 והוא נקרא באותו הזמן: FPK-Pascal. המפתחים של Free Pascal היו שבעי רצון מטורבו פסקל, ומטרת המהדר הייתה ברורה: ליצור מהדר מודרני, תחליף 32 ביט לטורבו פסקל.

באג

באג הוא תקלה במחשב שמקורה בתוכנה שנכתבה בצורה פגומה.

באג עלול לנבוע מעיצוב פגום של התוכנה, מתכנון לקוי של האלגוריתם, מטעות בשלב התכנות ועוד. באג יכול לבוא לידי ביטוי בצורות אחדות:

- התוכנה מפסיקה את פעולתה בטרם עת, ללא תוצאות או עם תוצאות חלקיות בלבד. באג מסוג זה נגרם, למשל, בעקבות חלוקה באפס.
- התוכנית פועלת עד סופה ואף נותנת תוצאות, אך תוצאות אלה אינן תקינות.
- התוכנית נכנסת ללולאה אינסופית, כלומר היא ממשיכה בפעולתה, אך אינה נותנת כל תוצאה, או שהיא חוזרת על אותה תוצאה שוב ושוב.
- התוכנית מבצעת בהצלחה את כל המוטל עליה, אך יש בה פרצה המאפשרת את ניצולה לרעה על ידי גורמים עוינים (ראו אבטחת מידע).

סביבת התכנות

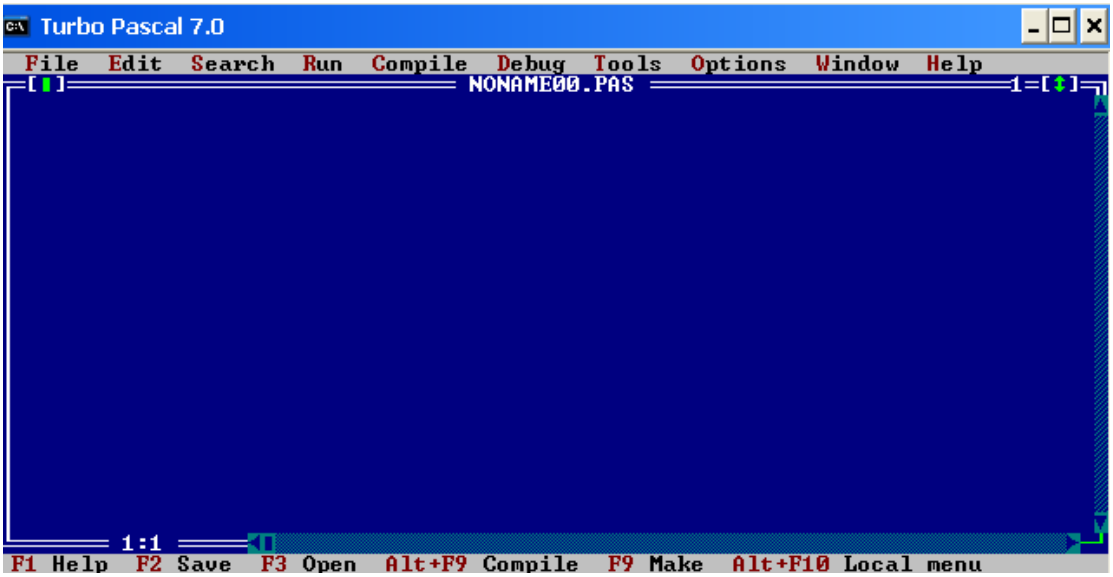
מהדר- **מהדר** (באנגלית *Compiler*) הוא תוכנת מחשב המתרגמת בין שפת מחשב אחת לשפת מחשב אחרת. המהדר הקלאסי מקבל כקלט תוכנית הכתובה בשפה עילית ומתרגם אותה לתוכנית בשפת מכונה.

אנו נשתמש ב- Turbo Pascal שזהו אחד המהדרים היותר מוצלחים ומוכרים. זהו בעצם מהדר שבעזרתו תוכלו לממש את שלמדתם במדריך ולנסות תוכניות.

- הערה - עליכם לתרגל את חומר הלימוד לא רק בתיאוריה אלא גם בפרטיקה כדי להבין כיצד לתכנת. קישור להורדת התוכנה :

http://www.simonhuggins.com/courses/progbegin/pascal/download/tp70.exe

מראה התוכנה:



קיצורי מקשים שימושיים בתוכנה:

פעולה	קיצור דרך	הסבר
File –Open	F3	פתיחה
File – Save	F2	שמירה
File - New	----	חדש
Cut	Shift +del	גזור
copy	Ctrl +ins	העתק
paste	Shift +ins	הדבק
Run	Ctrl+f6	לראות את פעולת התוכנה
User screen	Alt+f5	לראות את מסך דוס

תוכנית ראשונה

(v1.0.pas)

```
1) program v10;
2) begin
3) writeln('Hello World');
4) readln;
5) end.
```

הסבר :

Program v10- לכל תוכנית בכל שפת תכנות יש התחלה וסוף. כמו שאמרנו בהתחלה פסקל זאת שפת תכנות מאוד מסודרת לכן כשאנו כתובים תוכנה אנו צריכים לתת לה שם ובמקרה הזה השם הוא V10
;- הסימן "נקודה פסיק" יופיע אחרי פקודות בפסקל חוץ מכמה פקודות כגון:
Begin , else שעליהם נלמד בהמשך.
Begin- זאת הצהרה על התחלה של תוכנית.
* המספרים הם לא חלק התוכנית הם רק לשם הסבר.

*הערה-ניתן להתחיל תוכנית בפסקל גם בלי השם של התוכנית לדוגמא אם נשמיט את קטע הקוד העליון ונכתוב זאת כך:

```
2) begin
3) writeln('Hello World');
4) readln;
5) end.
```

זה יעבוד בלי שום בעיה.

writeln- זוהי פקודה להצגת פלט על הצג ,כרגע לא נסביר אותה כי היא תוסבר בהמשך. בואו נתמקד בטקסט שנכתב. כאשר אנו כותבים טקסט על הצג אנו משתמשים גם ברווח ואנטר. התווים האלה (אנטר ורווח) נקראים תווים לבנים, כלומר תווים שנמצאים אבל בעצם ריקים. הרי לרווח ולאנטר אין שום סימון אלא ריק. לכן
Hello world- מורכב גם מתו אחד לבן שזה רווח במקרה הזה.

readln-פקודה המשמשת לקליטת נתונים, גם על פקודה זו יוסבר בהמשך.
כאשר נשתמש בפקודה הזאת בתוכנית זו כמובן לא נקלוט נתונים ,המחשב יחכה עד שננסה לכתוב משהו ואחרי זה נלחץ על מקש האנטר וכך נוכל לראות את הפלט ההתחלתי. בלי הפקודה הזאות המחשב פשוט ייצא מהתוכנית ונצטרך להיכנס שוב בשביל לראות את הפלט
זה רק אפשרות כתוספת (לא חובה) .

End.–זוהי הפקודה שמסיימת את התוכנית כיוון שיש אחריה נקודה אם היה ; אז זה בעצם סיום של בלוק שגם על בלוק יוסבר בהמשך.

אלוגריתמיקה:

בימינו המחשבים תופסים מקום חשוב יותר ויותר מרגע לרגע וביכולתם כוח חישוב רב עוצמה אולם לא כל בעיה ניתנת לפתרון על-ידי המחשב. אם אתן למחשב לדוגמא להעריך שוויו של ציור או לבדוק מבחן בחיבור. הוא לא יוכל לעשות זאת. כפי שתראו בהמשך בתכנות תוכנית , בשפת תכנות ניתן יהיה להיתקל בבעיות לוגיות ובבעיות תחביריות. בעיות לוגיות אלו הם בעיות ברצף הפעולות ההגיוניות בתוכנה ובתנאים לדוגמא: אני כותב תוכנה שבה שעליך למיין את עצמך למין זכר או נקבה אז זה לא נכון מבחינה לוגית אם אני יאפשר לסמן גם את הנקבה וגם את הזכר. לעומת זאת הבעיות התחביריות הם פעולות שמהדר לא מכיר לדוגמא: אם ברצוני לחשב את תוצאת החיסור של שתי מספרים בשפה תכנות ובמקום להשתמש ב-'-' השתמשתי ב- "&" המהדר לא יבין מה ברצוני לבצע, ותהיה תקלה מבחינה תחבירית.

<u>אלגוריתם</u>- זהו רצף פעולות המיועד לפתרון בעיות בתכנות, על מנת להיות מתכנתים טובים עלינו להרגיל את עצמנו לתכנן אלגוריתם לכל בעיה בתכנות שאנו ניגשים אליה. הדבר זניח ברמות הנמוכות כלומר על מנת לכתוב טקסט על הצג אין צורך בכתיבת אלגוריתם אולם על מנת ליצור תוכנות מסובכות יותר או משחקים תראו שמאוד חשוב ליצור אלגוריתם ולא מה שנקרא ל"תכנת בספונטניות".
אם תחשבו על זה אנו משתמשים באלגוריתם בחיי היום יום גם אם אנו בכלל לא מודעים לזה :
לדוגמא כשאתה מוריד את הזבל אתה יודע מה לעשות בכל שלב ושלב.

1. לקחת את השקית.
2. לרדת למטה.
3. לזרוק את השקית לפח.

אם נסתכל ברצף הפעולות הנ"ל זהו סוג של אלגוריתם בעצם שאנו מבצעים יום יום. האלגוריתם הנ"ל אינו יכול להיות אלגוריתם מעשי כיוון שהוא כללי ובעצם מיוחס לבני-אדם אבל אם היה עלי לבצע אלגוריתם למחשב הייתי כותב אותו באופן הבא:

1. לקחת את השקית
2. לגשת לדלת הכניסה
3. לפתוח את דלת הכניסה
4. לרדת במדרגות
5. לפתוח את דלת הכניסה
6. לגשת אל הפח
7. לפתוח את הפח
8. להכניס את השקית בפנים

זהו כבר אלגוריתם יותר מפורט ונכון. תחשבו שאנו מבצעים אלגוריתמים כגון אלו יום יום ועוד הרבה יותר מסובכים אז אם תרצו או לא אינכם יכולים להימנע מאלגוריתמים גם בשפת המחשב. על מנת לכתוב אלגוריתם נכון עלינו להשתמש במספר קווים מנחים:

1. שהאלגוריתם יהיה נכון מבחינה תחבירית.
2. שהאלגוריתם יהיה מובן לכל מבצע (מתכנת, אזרח וכו...).
3. שהאלגוריתם יהיה נכון מבחינה לוגית.
4. שההוראות יהיו ספציפיות

אלגוריתם נכון מבחינה תחבירית- אלגוריתם נכון מבחינה תחבירית זהו אלגוריתם המשתמש בתווים הנכונים על מנת לבצע את רצף הפעולות בתוכנה.

אלגוריתם מובן לכל מבצע- כאשר אדם כותב אלגוריתם קודם כל עליו לדעת למי הוא כותב את האלגוריתם על מנת שאותו אחד שיעקוב אחר ההוראות יידע מה לעשות.
לדוגמא הוראה של מפקד לחייל קח את הנ"מ והתחל לירות במטוסים זוהי הוראה בהחלט נכונה וריאליסטית לחייל אך לא ניתן לצפות שאזרח יוכל לעשות זאת בלי לדעת איך לתפעל את הנ"מ ולירות במטוסים.

אלגוריתם נכון מבחינה לוגית- אלגוריתם נכון מבחינה לוגית זהו אלגוריתם שסדר פעולותיו נכון ושהוא נכון מבחינה חישובית ומתנאים. לדוגמא:

1. פלוט "תן מספר"
2. אם 8>צ אז
3. פלוט "נכון"

האלגוריתם הוא שגוי מבחינה לוגית כיוון שדבר ראשון איני משתמש במספר שכתב המשתמש ודבר שני הוא שלא ניתן להשוות בין אות צ לבין 8 ולכן לא ניתן לדעת מה גדול יותר.

<u>הוראות ספציפיות:</u>

על מנת ליצור אלגוריתם נכון עלינו להגיד במדויק מה אנו דורשים וכך נוכל לקבל את התוצאות הנדרשות לדוגמא:

1. קבל מספר מהמשתמש
2. חלק את המספר בשתיים
3. הצג את התוצאה לאחר מספר רגעים

האלגוריתם הנ"ל לא נכון כיוון שלא ידוע מה זה מספר רגעים חצי שעה?, שעה? , שנים? , הבנתם את הרעיון.

לעומת זאת אם אני אכתוב זאת כך:

1. קבל מספר מהמשתמש
2. חלק את המספר בשתיים
3. הצג את התוצאה אחרי שלוש שניות .

האלגוריתם הנ"ל בהחלט נכון!!!.

הכתיבה של האלגוריתם נעשית בשביל כל שפות התכנות באותו האופן לכן על מנת שכתיבת האלגוריתם תהיה נכונה עליכם לעקוב במדויק אחר ההוראות!.
אם נסתכל על החיים האמיתיים נראה שכל דבר מורכב בעצם מורכב ממרכיבים יותר קטנים לדוגמא:

דלת- זוהי בעצם מערכת שמורכבת ממנעול, צירים, שכבת פלדה או עץ,ידית.
אותו הדבר מיושם גם באלגוריתם, כלומר אם יש מרכיבים קטנים שקשורים למרכיב יותר גדול אז הם יהיו מקושרים במספרים אל המרכיב הנ"ל.

<u>דוגמא:</u>

עליי לעשות אלגוריתם שבו עליי לקבל נתון האם המשתמש הוא זכר הוא נקבה ובהתאם לשאול שאלה את גילו של המשתמש.
האלגוריתם:
1. פלט- סווג את מינך זכר או נקבה
2. קלט למשתנה-איקס

3. אם איקס=זכר אז
3.1 פלט- בן כמה אתה
4. אם איקס=נקבה אז
4.1 פלט- בת כמה את.

ניתן לראות שעל מנת לכתוב אלגוריתם אני צריך כמה שיותר ,שההוראה תהיה ספציפית ברורה וקצרה, כלומר לא להתחיל לכתוב סיפורים שלמים אלא לכתוב בקיצור ולעניין. (זה מאוד חשוב להקפיד על כך!!!).

ניתן לראות שיש הוראה בתוך הוראה, בדוגמא הנ"ל בתנאים.
עקב כך הפלט של שאילת הגיל (לא משנה אם זכר או נקבה) קשור לתנאי ולכן הוא ממוספר בהתאם (4.1,3.1). וזאת אומרת שאני מייחס אותו לתנאי ולא להוראה בפני עצמה(זה נקרא הוראה מקוננת).

דבר נוסף שחשוב באלגוריתם כאשר הוא מרוכב יותר , הוא פירוק לתת-משימות. אם נתייחס לתוכנית כמשימה אז על מנת לפתור חלק חלק בנפרד (כדי שהעבודה תהיה יותר קלה וברורה יותר) עלי לעשות זאת באמצעות תת-משימות. התת-משימות כתובות בכלליות לא כמו באלגוריתם, חשוב לפרק למספר תת-משימות מותאם לתוכנית, כלומר אם יש לי תוכנית שעלי לתכנת לא לפרק למספר מועט של תת-משימות, אך גם ללא לפרק ליותר מדי תת-משימות ביחס לתוכנית כי אז זה כבר לא תת-משימות.
דוגמא: עלי לכתוב תוכנית שמקבלת 10 ציונים של מקצועות לתמיד ומחזירה את ממוצע הציונים.
הפירוק לתת-משימות יתבצע כך:

<u>תת-משימות:</u>

1. קליטת הציונים
2. חיבור המספרים וחילוק בעשר

ניתן לראות שלפי הדוגמא תתי-המשימות כלליות יותר מאשר האלגוריתם ובעצם מפרקות את משימת העל (בניית תוכנה) למשימות כך שיהיה יותר קל לבנות את התוכנה. בהמשך יורחב עוד יותר על אלגוריתם, כתיבת אלגוריתם, ואלגוריתם במחשבים.

פרק 2: עקרונות בסיסים בפסקל

WriteLn , Write

Writeln ו- write אלו הם פקודות המאפשרות הצגת פלט על מסך המשתמש.

* פלט- הנתונים שהתוכנית מציגה על הצג.

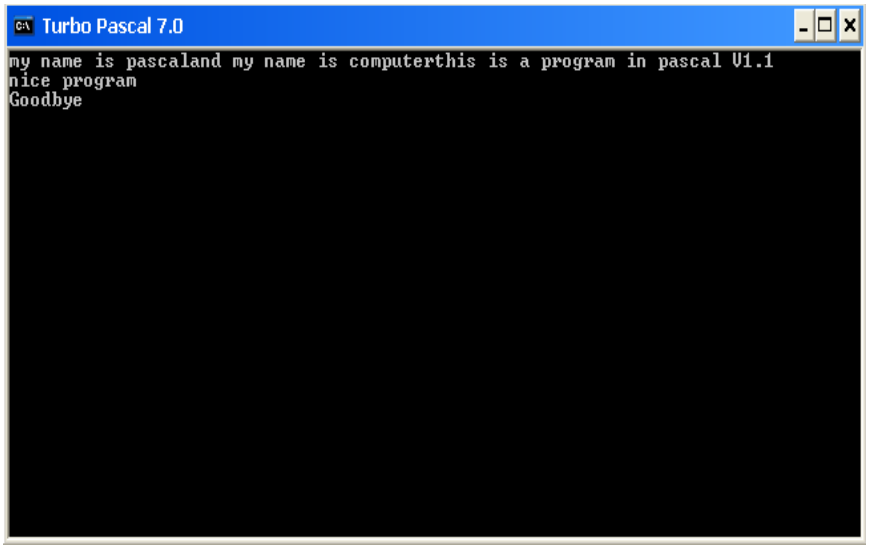
ההבדל בין writeln ל- write הוא מאוד פשוט. Write פקודה הפולטת את הטקסט על הצג , ואחרי זה נשארת באותה שורה. Writeln פקודה הפולטת את הטקסט ואחרי זה יורדת שורה זאת אומרת הדבר הבא שנכתוב יהיה בשורה מתחת.

תוכנית שניה

(v1.1.pas)

```
1) program v11;
2) begin
3) write('my name is Pascal');
4) write('and my name is computer');
5) writeln('this is a program',' in Pascal v1.1');
6) writeln('nice program');
7) writeln;
8) write('Goodbye');
9) end.
```

הפלט על הצג:



הסבר לקוד:
3- נכתב הטקסט my name is pascal ולאחר מכן אין ירידת שורה לכן השורה הבאה נמצאת באותר השורה.
4- נכתב הטקסט and my name is computer ולאחר מכן גם אין ירידת שורה.
5- נכתב הטקסט this is a program in pascal v1.1 ולאחר מכן גם אין ירידת שורה כיוון שירידת השורה מתבצעת רק לאחר הפקודה writeln ולא לפניה.
ניתן לראות שיש שימוש בפסיק בתוך הפקודה writeln וזאת על מנת העיקר להוסיף נתונים של משתנים לפלט ולא משתמשים בזה בשביל לחבר טקסטים כמו שנעשה בתוכנית הזאת.
*הערה- אם נרצה להוסיף נתונים של משתנה צריך לעשות זאת מחוץ לגרש, אחרת זה ייכתב כתו.
דוגמא מקוצרת:
נקרא ל- x משתנה ששווה 8.
Writeln('there are x people in the room');
הפלט יהיה: there are x people in the room
לעומת זאת אם אני ישתמש בפסיק התוצאה תהיה שונה, לדומא:
Writeln('there are ',x, ' in the room');
הפלט יהיה: there are 8 people in the room
6- נכתב הטקסט nice program בשורה מתחת כיוון שלפני הפקודה הזאת הייתה פקודה writeln שהורידה את מה שאחריה שורה.
7- נכתב Goodbye מתחת לשורה כיוון שהפקודה שלפניו (הכוונה לפקודה מספר 6) היא writeln.

*לפעמים בחלק מהגרסאות של התוכנה יש בעיה בירידה שורה לכן נמליץ על גירסה 7 ומעלה... כמו בשורה 7

'מילה מסוימת' - זה בעצם גרשיים שמטרתם לייצג משתנה מסוג string – שזה בעצם מילה או משפט מסוים ואם אין גרשיים אז זה משתנה או פקודה מסוימת .
WriteLn - בפקודה זו אפשר לכתוב גם מילים וגם משתנים ובניהם פסיקים שאינם מופעים בפלט.

משתנים בפסקל

משתנה- בתכנות משתנה הוא חלק מהזיכרון בתוכנית המכיל נתון שיכול להשתנות בזמן הריצה על פי הפקודות הניתנות לו. המשתנה מאפשר שמירת נתונים ,והם לא ימחקו עד שלא נשנה אותם או שנסגור את התוכנית .
על מנת להציב ערכים למשתנים עלינו להשתמש ב- =: , לאחר שמציבים ערך למשתנה הערך הקודם שלו נעלם. דוגמא:

```
X:=4;
X:=6;
```

הסבר:

בהתחלה הצבנו את הערך ארבע באיקס ולאחר מכן הצבנו 6 באיקס לכן הערך הסופי של איקס יהיה 6 כיוון שהערך ההתחלתי (4) נעלם.

הגדרת משתנה תתבצע כך :
1) בראש התוכנית
2)

```
Var   שם משתנה  :  סוג;
```

סוגי משתנים:

שם	גודל	הסבר
Byte	בית 1 (8 ביטים)	מכיל ערכים בין 0 ל-255 רק השלמים.
Integer	2 בתים (16 ביטים)	מכיל ערכים בין 32768- עד 32767 מכיל ערכים מפריים שלמים
Longint	4 בתים (32 ביטים)	מכיל ערכים בין 2147483648- עד 2147483647 מכיל ערכים מספריים שלמים
Real	6 בתים (48 ביטים)	מכיל ערכים בין 2.9*10E-39 עד 1.7*10E38 מכיל ערכים מספריים עשרוניים ויכול לנוע בגבולותיו (גלישה) עד תוספת 12 ספרות משמעותיות
Word	2 בתים (16 ביטים)	מכיל ערכים בין 0 עד ל-65535 ערכים שלמים בלבד.
Char	בית 1 (8 ביטים)	תו אחד בלבד (לפי טבלת אסקי שתצויין בהמשך).
String	באחד יותר מהאורך	תווים. בהמשך יוסבר בהרחבה.
Single	4 בתים (32 ביטים)	Real קצר
Double	8 בתים (64 ביטים)	real ארוך
Extanded	10 בתים (80 ביטים)	real מאוד ארוך - נע בין 3.4*10E-4932 עד 1.1*10E4932 ומיועד לחישובים מאוד גדולים
Comp	8 בתים (64 ביטים)	מכיל ערכים מספריים שלמים גדולים מאוד. נע בין -(2*10E63)+1 עד (2*10E63)-1
Boolean	----------	ערכים false או True (יוסבר בהמשך יותר)

*הערה- רצוי להשתמש במשתנה הנדרש לפי הצרכים. אין צורך להשתמש במשתנים גדולים מדי בשביל להציב ערכים קטנים מדי וההפך. לדוגמא לא צריך להציב ערכים של ציונים במשתנה כגון: comp. Integer בהחלט מספיק!!!

משתנים:

על מנת ליצור יותר אפשרויות ויותר פיקנטיות בתוכניות עלינו להשתמש במשתנים אשר בעצם מעשירים את התוכנית. המשתנים הם לצורך נוחות שמירת הנתונים וכדי להשתמש בהם בעתיד בשביל שלא "נשכח אותם".

מהם הדרכים החוקיות לנתינת שמות למשתנים ושמות בכלל (פונקציות,תוכניות וכו)

1. כל שם חייב להתחיל באות
2. מותר בשם שיהיו התווים הבאים: אותיות, ספרות וקו תחתון
3. אסור להשתמש במילים שמורות כגון begin, end.
4. מספר התווים המשמעותיים בשם הוא 8, כלומר אם נכתוב שני שמות משתנים בעלי 15 תווים כאשר עשרת התווים הראשונים הם זהים, אז המהדר לא יבחין שאלו משתנים בעלי שמות שונים.

הערות ודגשים:

- כדאי מאוד לקרוא לשמות בשמות שמתאימים לתפקיד זה לא חובה אך זה יחסוך מכם בעיות בעתיד כמו בלבול בין משתנים.
- דגש: לא ניתן לקרוא לשתי מרכיבים באותו השם!!! (שם תוכנית,משתנים, פונקציות ועוד...)
- משתנה מגדירים אחרי המילה program במקום שנקרא var.
- חשוב לדעת שבפסקל אין הבדק בין אותיות גדולות לקטנות בשמות אך בשפות אחרות זה לא חייב להיות כך.

לדוגמא:

```
Program dugma;
Var num1,num2:integer;
……
```

דבר נוסף הוא שכדי לקרוא להם בשמות מתאים לא סתם למשל מספר זה num.

הערות בתוכנית (תיעוד)

כאשר אנו כותבים תוכנית בפסקל עלינו להסביר מה התוכנית עושה ומהם התפקידים של המשתנים השונים וכמו כן הפרוצדורות והפונקציות שעליהם נלמד בהמשך. על מנת להיות מתכנת טוב עליך לכתוב תיעוד (היערות) כך שאנשים אחרים יוכלו לעקוב אחריי תוכניתך ולדעת את התפקידים השונים של הגורמים בתוכנה. למה זה נחוץ?
נגיד שאתה התקבלת לחברת ענק ועליך הופל לבנות תוכנית מסוימת אתה עובד עליה כמה זמן ולאחר מכן מתכנת אחר ממשיך את עבודתך. על מנת שהוא ימשיך את עבודתך כראוי עליו להיכנס ל"ראש שלך" ולדעת בדיוק איך התכוונת לכתוב את התוכנה. בשביל שאותו מתכנת יידע איך להמשיך הוא צריך לדעת את הגורמים השונים שצריכים להיות בתוכנה ומהם התפקידים שלהם וגם את מטרת התוכנה. לכן יש צורך בהערות (תיעוד) ולכן אתה (המתכנת הצעיר) צריך לדעת כיצד לעשות זאת.

כיצד לשים הערות בתוכנית:

בפסקל ישנן 2 דרכים לכתיבת הערות:

1. באמצעות סוגריים מסולסלות {} כאשר בתוך הסוגריים כותבים את התיעוד של התוכנה.
2. (* *) באמצעות סוגריים עגולים וכפל. המהדר יתעלם מהתוכן שבתוך סוגריים אלה.

דוגמה להערות:

```
Program dogma;
{input: two numbers output: the amount of the numbers}
Var num1,num2:integer;
(*num1-input a number type integer num2- input a number type integer
*)
Begin
Writeln('input two numbers');
Readln(num1,num2);
Writeln('the sum is: ,',num1+num2);
End.
```

הסבר:
בהתחלה כתוב מהי מטרת התוכנית שזה, לחשב את הסכום של שתי מספרים.
לאחר מכן כתבנו מהו סוג המשתנים בתוכנית ואת תפקידיהם שזה קליטת המספרים מהמשתמש.

פעולות על מספרים

סימן	הסבר	דוגמא
+	זהו סימן החיבור	X:=5+5;
-	זהו סימן החיסור \ מספר שלילי	X:=7-6; X:=5+-8;
*	זהו סימן הכפל	X:=5*5;
/	זהו סימן החילוק. חייב להציב במשתנה שהוא יכול להכיל מספרים עשרוניים (כמו real לדוגמא אם תוצאת החילוק היא 2 אז יוצב 2.0	X:=10/2;

מספרים בינאריים

ספירה על **בסיס בינארי** היא ספירה על בסיס 2. היא פותחה במקור ע"י גוטפריד וילהלם לייבניץ במאה ה-17. היא משמשת כיום בעיקר בתחום המחשבים - זאת מכיוון שמחשב מכיר רק שני מצבים לכל ספרה - 0 (כבוי) ו-1 (דולק).

דוגמא: המספר הבינארי 1101 שווה ל- 13 (בייצוג הרגיל שהוא על בסיס העשרוני).

כיצד הופכים מספר שלם לבסיס 2?

עושים חלוקה של המס' הנתון ב 2 רושמים את התוצאה ואת השארית (את השארית ניתן להכפיל פי2 (אם המספר אי זוגי פשוט החסירו '1' , אל תשכחו לכתוב את ה- 1 כשארית)) את ה1 שמים בשארית, אם אין שארית רושמים "0" . וכך מחלקים עד שהתוצאה היא "0". מסתכלים על השארית... מה שיצא לכם בסוף אתם רושמים בתחילת המס' הבינארי.

לדוגמא : המספר 8 בבינארי. 8\2 שווה 4 לכן רושמים 0 כי אין שארית. 4\2 שווה 2 בלי שארית לכן שוב רושמים 0 , 2 חלקי 2 שווה 1 ושוב רושמים 0 ו1\2 שווה חצי ולכן רושמים 1 כי יש שארית . שימו לב שהתוצאה היא 0 ובגלל זה אנו מספיקים את החלוקה ויש בידנו את התוצאה הסופית !!!
התוצאה היא :1000.

עוד דוגמא:

	תוצאה	שארית
271\2	135	1
135\2	67	1
67\2	33	1
33\2	16	1
16\2	8	0
8\2	4	0
4\2	2	0
2\2	1	0
1\2	0	1

והמס' הוא: 100001111

<u>פעולות בין שתי מספרים בינארים</u>

<u>XOR</u>

	1	0
1	0	1
0	1	0

*הסבר אם יש לכם 1ו 1 התוצאה תמיד תהייה "0" אותו הדבר לגבי "0,0". רק "1" ו-"0" שווים ל-1 אחרי ביצוע פעולת קסור.

<u>AND</u>

	1	0
1	1	0
0	0	0

*הסבר: כמו כפל פשוט... "1" ו "1" שווה 1...וכך הלאה...

<u>OR</u>

	1	0
1	1	1
0	1	0

*כמו חיבור רק ש 1+1 שווה 1 ולא 2 .

<u>NOT</u>

הפעולה הופכת 1 ל-0 ו-0 ל-1. לדוגמא ;not(1) מחזירה 0 .

*הערה: מספרים בינאריים שימושים רק ברמה גבוה אז בינתיים לא חייב להתעמק אבל כדאי לדעת!!!

פונקציות מתמטיות ופעולות מיוחדות

שם	דוגמא	הסבר
DIV	X:=10 DIV 3	בודק את מספר הפעמים שמספר מסוים בשלמותו נכנס בתוך מספר אחר. 3 נכנס שלוש פעמים בשלמותו ב-10 ולכן התוצאה היא 3.
MOD	X:=101 mod 10	מחזיר את השארית לדוגמא אם עושים מאה ואחד חלקי 10 אז השארית היא אחד ולכן הערך של המשתנה יהיה אחד.
ABS	X:=ABS(-2)	מחזיר את הערך מוחלט
COS	X:=COS(180)	מחזיר את כוסינוס
SIN	X:= sin(45)	מחזיר את הסינוס
trunc	X:= trunk(4.2)	מקצץ מחזיר רק את השלם (מעגל תמיד כלפי מטה)
ROUND	X:=ROUND(6.7)	מעגל את המספר על פי חוקי מתמטיקה אך אם יש מספר כגון 6.5 הוא יעוגל כלפי מעלה.

- ישנם פעולות נוספות נרחיב בהמשך(בפרק הבא נשתמש יותר בדברים האלו!!!)

מספרים אקראיים

הפרוצדורה Randomize:

כדי ליצור מספרים אקראיים שונים בכל הרצה של התוכנית יש לקרוא לפרוצדורה (ידובר בהמשך על פרוצדורה) Randomize פעם אחת בהתחלת התוכנית הראשית.

על מנת ליצור מספריים אקראיים לא מספיק רק לקרוא לפרוצדורה Randomize אלא גם להשתמש בפקודה Random.
כאשר אנו מעלים באקראי מספרים עלינו להשוות את אותו מספר אקראי למשתנה כדוגמה הזאת:
x:= random(n);,מספר כלשהו-N.

שימושים שונים בפקודה Random:
1. 0<=x<n על מנת להגריל מספרים בטווח הזה נעשה זאת כך x:= random (n);
2. a<=x<n על מנת להגריל מספרים בטווח הזה נעשה זאת כך x:= random (n-a)+a;
3. a<=x<=n על מנת להגריל מספרים בטווח הזה נעשה זאת כך x:= random (n-a+1)+a;
4. 1<=x<=n על מנת להגריל מספרים בטווח הזה נעשה זאת כך x:= random (n)+1

*הערה- כאשר מגרילים מספרים אקראיים הטווח יתחיל מ0- אולם זה לא יהיה כך אם תשנה את הטווח.

נכתוב עכשיו תוכנית דוגמא שתמחיש את הדבר כאשר התוכנית תפלוט באופן אקראי מספר.

```
1)Var a: integer;
2) Begin
3) Randomize;
4) a:= random(10);
5) Writeln(a);
6) End.
```

הסבר תוכנית:

1- הצהרה על משתנה a מסוג integer.
2- התחלת התוכנית.
3- הצהרה על הפרוצדורה randomize בתוכנית בשביל שיוגרלו בכל פעם שהתוכנית רצה מספרים אחרים.
4- השוואות המשתנה a למספר האקראי.
5- כתיבת המספר האקראי.
6- סיום התוכנית.

Readln

זהוי פקודה אשר קולטת נתונים למשתנים אפשר לקלוט מספר נתונים ע"י הפרדה בפסיק כמו בדוגמאות הבאות :

```
Readln(x);
Readln(X,Y,Z);
READLN(ST);
```

עד שהמשתמש לא יכתוב נתונים וילחץ על אנטר התוכנית לא תמשיך לרוץ

readln- הפקודה קולטת נתונים למשתנים ובסוף קליטת הנתונים יורדת שורה.
read - הפקודה קולטת נתונים למשתנים ובסוף קליטת הנתונים לא יורדת שורה.

תרגילים לסיכום פרק 2

1) כתוב תוכנית שתקלוט שם וסיסמה ותיתן כפלט את השם והסיסמה
2) תכתוב קטע תוכנית שתקלוט שתי מספרים ותפלוט :חיבור .
3) תיצור מספר אקראי בין 8 ל 100 ותפלוט אותו
4) מה הערך הבינארי של 100 ו של 76?
5) מה הפלט של קטע התוכנית הבא:

```
X:=1
X:=not(x);
Y:=1 and 1;
Writeln(x,y);
```

6) כתוב תוכנית הקולטת מספר ונותנת כפלט את ספרת היחידות , ספרת העשרות ומאות של אותו בספר (נניח שהמספר שנקלט הוא תלת ספרתי)
7) נסו לבדוק מה ההבדל בין ROUND ל- TRUNK .
8) כתוב קטע תוכנית שקולט 4 מספרים ובודק מה הממוצע שלהם (החיובי בלבד)(ערך מוחלט) והשלם
9) עשו טבלת מעקב לקטע תוכנית הבא:

```
X:=6;
Write(x);
X:=x+1;
Write(x);
X:=x div 2;
X:=x mod 2;
Write(x);
```

גלו מה הפלט בסוף (מה מופיע על המסך בסוף ריצה התוכנית)

10) עשו טבלת מעקב לקטע תוכנית הבא:

```
X:=1;
Y:=0;
Z:=x or y;
A:=x and y;
Write(x,y,z,a);
```

גלו מה הפלט בסוף (מה שמופיע על המסך בסוף).

11) האם שם המשתנה נכון?

```
1) 'ppp
2) hello
3) 13$$32fc
4) jop&&
5) num
```

פתרונות

1)

```
Program tar1;
Var name,password:string[10];
Begin
Writeln('type your name and password);
Readln(name);
Readln(password);
Writeln('the name is ',name,'the password is ',password);
End.
```

הסבר אנחנו מגדירים שתי משתנים מסוג סטרינג שזהו בעצם משתנה שיכיל אוסף תווים
/אחרי זה אנחנו קולטים ופולטים אותם

2)

```
Writeln('type two numbers')
Readln(n1,n2);
Writeln('the sum is ',n1+n2);
```

3)

```
Randomize;
X:=random(100)+8;
```

4)הערך הבינארי של 100 הוא: 1100100
הערך הבינארי של 76 הוא :1001100
5)

01

6)

```
Writeln('type a num');
Readln(n);
Writeln('yhidot:',n mod 10);
Writeln('asrot:',(n div 10)mod 10);
Writeln('mhot',n div 100);
```

7) ROUND הוא מעגל אם יש אחרי הנקודה יותר מ5 אז זה מעגל מעלה והשני תמיד מטה.

8)

```
Readln(n1,n2,n3,n4);
M:=round((n1+n2+n3+n4)/4)
M:=abs(m);
Writeln('the avg is:',m);
```

9)

פלט	x
6	6
7	6+1=7
אין	7 div 2 =3
1	3 mod 2 =1

הפלט על המסך:

671

*כדי לבדוק אם מספר הוא זוגי עושים מוד 2 כמו בדוגמא
3 MOD 2 =1 זאת אומרת שהוא אי זוגי אם הוא זוגי אז התוצאה היא 0

(10

פלט	A	Z	Y	X
				1
			0	1
		1 or 0 =1	0	1
	1 and 0 =0	1	0	1
1010	0	1	0	1

הפלט הוספי הוא 1010

(11

1)לא נכון\אסור סימנים
2)נכון
3)אסור סימנים לא נכון
4)אסור סימנים לא נכון
5)נכון

*חזור טוב טוב על הפרק הזה כדי שתוכל להבין את הפרק הבא!!!

פרק 3: לוגיקה (תנאים)

תנאים בפסקל (תרגול מילולי-לוגי)

על מנת להסביר את הדבר שאני הולך לדבר עליו עלי להסביר לכם קודם מהם תנאים. ובכן תנאי כמו בחיים האמיתיים מותנה במרכיב שיקרה או לא יקרה (אלו הבסיסיים ביותר) וכמובן שאחרי כל תנאי חייבת להיות הפעולה או התוצאה הבלתי נמנעת. לדוגמא:
1.אני אלך לבית-הספר רק אם אח שלי הקטן לא יהיה חולה.
2. אם דני או יוסי הולכים לטיול אז אני גם אלך לטיול.
בואו נפרק את המשפטים הנ"ל:

משפט 1:
תנאי: אח שלי הקטן לא יהיה חולה.
פעולה\תוצאה: ללכת לבית-הספר.
משפט 2:
תנאי: אם דני או יוסי הולכים לטיול.
פעולה\תוצאה: גם אני אלך לטיול.

כמובן יכולים להיות משפטים הרבה יותר מורכבים מבחינת תנאים ולפיכך גם בשפת המחשבים אך כרגע זה מספיק לעכשיו.
דוגמא תחבירית לתנאי:

אם [X>8] אז כתוב: מה נשמע?.

פירוק דוגמא תחבירית:
התנאי: הוא X>8
הפעולה\תוצאה: פלט: מה נשמע?

אך יכולים להיות גם מרכיבים לוגיים שיכולים להיות בתנאי. ועל זה עכשיו נדון.
בפסקל כמו בכל שפת תכנות אחרת ישנם גם תנאים לוגיים מילוליים. למדנו על התנאים החשבוניים כגון: <, >, =, <>. אך ישנם גם תנאים מילוליים כגון: או, וגם, שלילת תנאי (לוגיקה הפוכה) ועוד....

על מנת להבין כיצד התנאים הללו עובדים בפסקל עלינו להבין קודם כל את השימוש בהם והכתיבה התחבירית שלהם
:And
And- זהו תנאי לוגי שמחייב את התקיימותם של כל התנאי שהוא מייצג כל התנאים שהוא מייצג צריכים להתקיים לדוגמא:

אם (8<X) And (x<16) אז
פלט-ניסיון

*הערה חשובה מאוד!: כאשר אנו משתמשים בתנאי אם ויש איזשהו תנאי לוגי המקושר לפחות לשני תנאים (כמו בדוגמא) יש לתחום את התנאים בסוגריים הללו ( ) .

בדוגמא הנ"ל And מקשר בין שני תנאים 8<X וגם X<16 לכן בשביל שהתנאי הזה יתקיים צריך לקרות שגם איקס גדול מ-8 וגם איקס קטן משש עשרה.

פירוק תחבירי:

תנאי: (x<16) And (8<X)
פעולה\תוצאה: פלט-ניסיון

:Or
Or- זהו תנאי לוגי שעל מנת שהוא יתרחש צריך להתקיים תנאי אחד לפחות (או). בדוגמא זה יראה כך:
אם (X>8) או (16<X) אז
פלט-ניסיון

התנאי הלוגי הנ"ל יתקיים רק אם לפחות תנאי אחד יתקיים. זאת אמרת אסור שאיקס יהיה קטן מ-9. לכן תווך המספרים שהתנאי הלוגי הנ"ל יתקיים הוא בין 9 לאינסוף (כלפי מעלה).

פירוק תחבירי:
תנאי: (X>8) או (16<X)
פעולה\תוצאה: פלט-ניסיון

:Not
Not- תנאי זה משום מה הוא תמיד מבלבל את המתלמדים ולכן גם נדיר מאוד לראות בכלל את השימוש בתנאי הלוגי הזה אצל מתלמדים ואף אצל מתקדמים (אולי בגלל שהוא מתקיים בלוגיקה הפוכה).
על מנת שהתנאי הלוגי הנ"ל יתקיים צריך שהתנאי שאנו שמנו לא התקיים. קצת מבלבל?????? אני בטוח שעם הדוגמא הבאה יהיה יותר קל להבין.

אם not (x>9) אז הפלט - נסיון

הסבר: על מנת שהתנאי הנ"ל יתקיים אז אסור ש-X יהיה גדול מ-9. זאת אמרת שהתנאי הלוגי הנ"ל יתקיים רק אם התנאי עצמו לא מתקיים.

פירוק תחבירי:
תנאי: not (x>9)
פעולה\תוצאה: פלט- ניסיון.

:Xor
Xor- תנאי זה אתם גם תתקשו למצוא אצל מתלמדים ומתקדמים ואם בכלל בתוכניות שונות. אולי בגלל שהוא חופף קצת ל-or. על מנת שהתנאי הלוגי הנ"ל יתקיים צריך להתקיים רק תנאי אחד אבל רק תנאי אחד!!!.

לדוגמא: אם (Y<20) Xor (X>8) אז פלט-ניסיון.

הסבר: על מנת שהתנאי הנ"ל יתקיים יכולות להיות שתי אפשרויות.
אפשרות 1: X>8 ו-y>20
אפשרות 2: X<8 ו-y<20

פירוק תחבירי:
תנאי: (Y<20) Xor (X>8)
פעולה\תוצאה: פלט-ניסיון

כבר למדנו ליצור משתנים, לקלוט פלט ולתת פלט.
נגיד ואנו קולטים מספר איך אנו נבדוק אם הוא גדול מ-2 וניתן תשובה בהתאם?
בשביל זה יש לנו את ההוראות אשר נקראות תנאים. ישנם בפסקל שני סוגי תנאים : IF ו CASE.

נתחיל בהתמקדות ב- IF שהוא התנאי העיקרי והשימושי ביותר.

IF

```
1)IF X=1 THEN
2)WRITELN('X=1')
3)ELSE
4)WRITELN('X<>1');
```

בשורה הראשונה בדקנו אם איקס שווה ל-1 כמובן אפשר לבדוק דברים אחרים ולא רק שיווין את זה נסביר בהמשך. בשורה השניה כתבנו שאיקס שווה אחד, דבר זה קורא רק אם איקס שווה ל-1.
בשורה השלישית כתבנו הוראה שמה שאחריה קורה במקרה שהתנאי לא נכון.

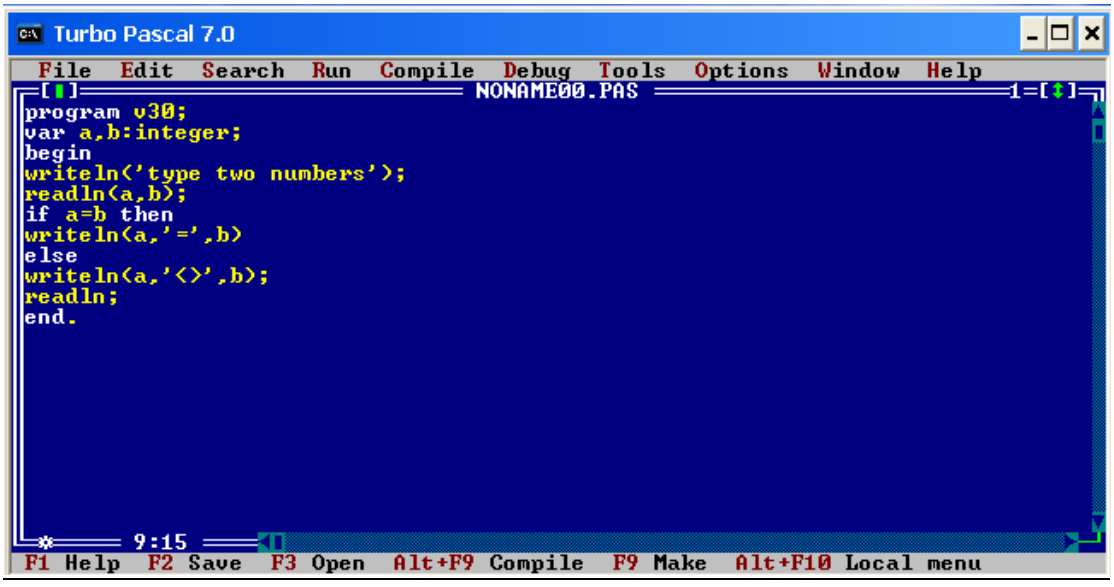
בשביל להסביר טוב יותר נשתמש בטבלת מעקב (כלי שעוזר להבין טוב יותר תוכניות)
במקרה שה-איקס שווה 1 אז הטבלה נראת כך

פלט	X=1	x
X=1	X=1 true	1

במקרה שהאיקס שווה 0 אז

פלט	X=1	X
X<>1	X=1 FALSE	0

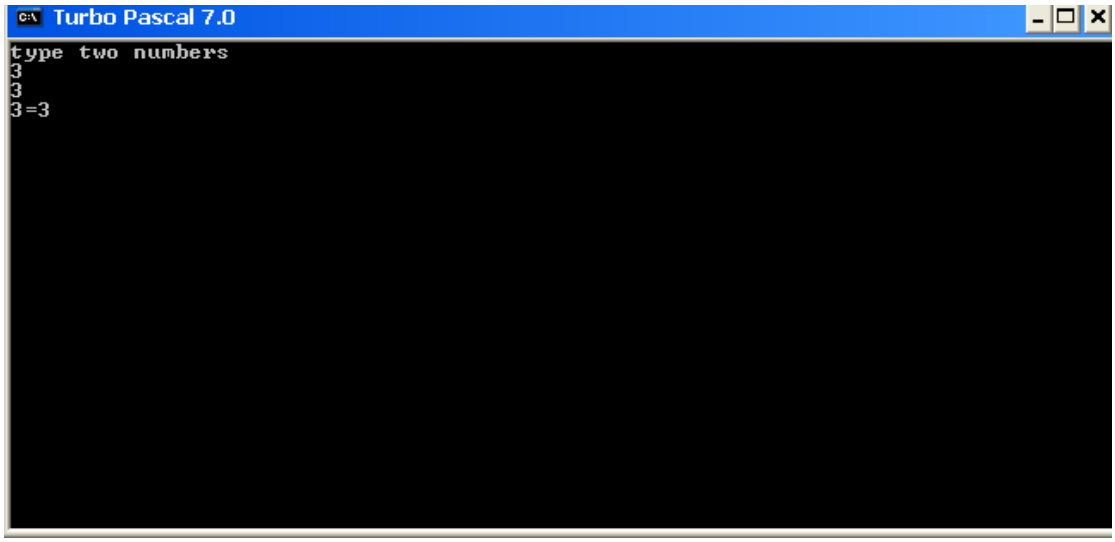
תוכנית V30



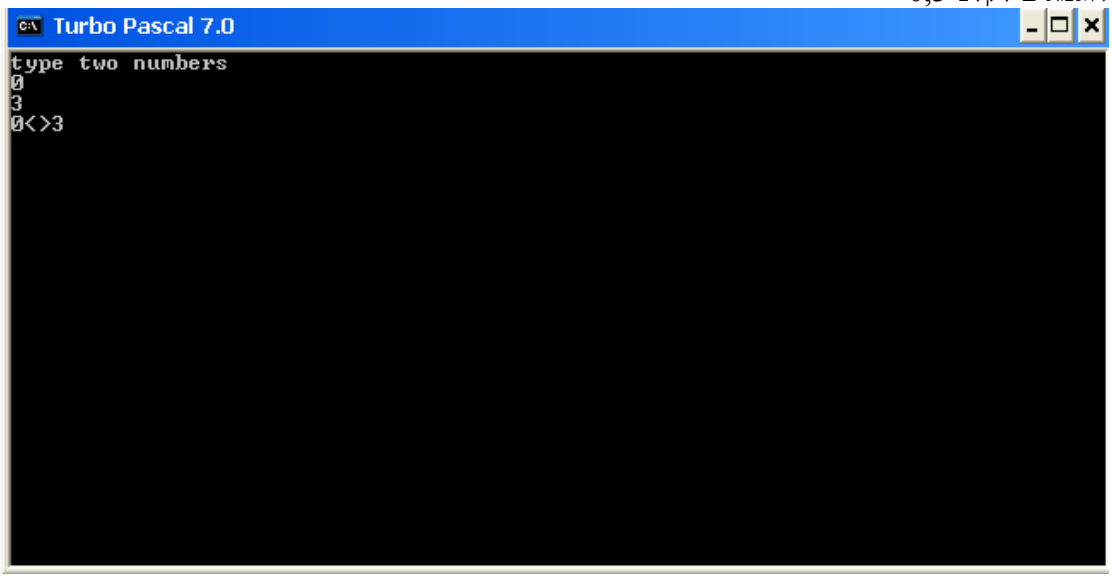
בהתחלה כתבנו פלט "כתבו שתי מספרים"

*הערה :אם לא יבוא פלט שיבוא ויסבר מה לעשות המשתמש לא ידע מה לעשות.
אחר-כך קלטנו שתי מספרים
אחרי זה בדקנו שיוויון ונתנו פלט בהתאם .

דוגמא א לקלט :3,3



דוגמא ב לקלט 0,3



טבלת מעקב

על מנת לעקוב בצורה נכונה אחר תוכניות שונות עלינו להשתמש בכלי הנקרא טבלת מעקב. הכלי הוא מאוד טכני ורק לאחר תרגול תוכלו לשלוט בכלי זה בלי שום בעיה.
טבלת מעקב מורכבת מטבלה שבה אנו עוקבים אחר התנאים השונים שבתוכנית המשתנים ואחר הפלט של התוכנית ובנוסף לכך ניתן לעקוב באמצעות כלי זה אחר מטרת התוכנית. כרגע אנו נבצע טבלת מעקב פשוטה לתוכנית הבאה. בהמשך יהיו תרגילים עם טבלאות מעקב הרבה יותר מסובכות.

התוכנית:

```
program v...;
var num1,num2:integer;
begin
      writeln('enter two numbers');
      readln(num1,num2);
      if num1>num2 then
            writeln(num1>num2);
      if num1<num2 then
            writeln(num2>num1);
end.
```

הקלט בתוכנית הנ"ל הוא: num1=7 ו- num2=9.

Num1	Num2	num1>num2	Num2>num1	פלט
7	9	7>9 (false)	9>7 (true)	7<9

בלוקים

כאשר אנו רוצים לדוגמא שאם תנאי מסוים יתקיים ויתבצעו מספר דברים אז חייב לשים את הפקודות בין BEGIN ל-END כמו בדוגמא הבא :

```
If x=1 then Begin
      Writeln('hi');
      Readln(y);
End;
Writeln('GoodBye');
```

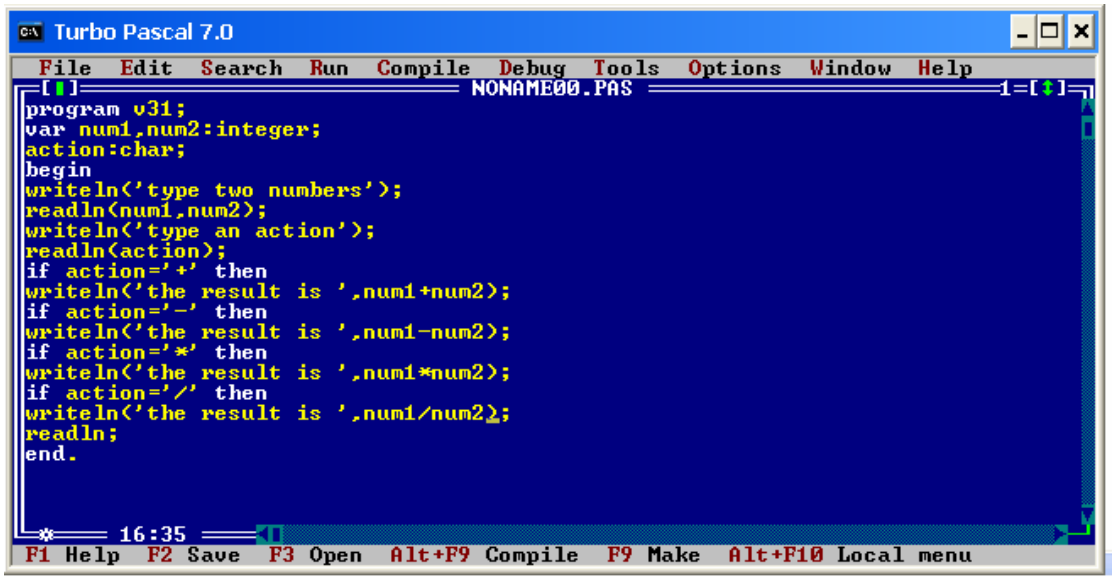
הסבר:
אם איקס שווה אחד אז מה שבין "הבלוק" יתקיים ייכתב Hi וייקלט מספר ל-Y ואחרי כן בכל מקרה ייכתב GoodBye.

```
If x=y then
      Writeln('x=y');
Write('hi'):
```

המילה Hi בכל מקרה תיכתב כי אחרי IF מתבצעת רק הוראה אחת בלבד, ורק אם יש בלוק אז מתבצע כל מה שבבלוק.

תוכנית V3.1

בתוכנית זו אנו נקלוט שתי מספרים ותו שמציין את הפעולה ביניהן ובסוף יינתן כפלט הפתרון זאת אומרת נבנה מחשבון פשוט .



הסבר:
בתוכנית נקלט שתי מספרים ותו ומתבצעת פעולה בהתאם לתו . תריצו את התוכנית ותראו שזה עובד אבל ישנן מספר בעיות לכן עכשיו נשפר מעט את התוכנית ונמנע שגיאות בעתיד

שדה עריכה

על מנת ליצור תוכניות עם מראה מרווח של טקסט בהתאם, עלינו להשתמש בשדה עריכה שייפה את הרווחים שיהיו בהתאם בין הטקסטים.

שדה עריכה עבור משתנים מסוג שלם ו-string.

```
Begin
Writeln('How are you?':20);
End.
```

הדבר יראה כך:

ניתן לראות שיש 20 תווים לפני כתיבת הטקסט How are you?. ובעצם שדה עריכה במשתנים מסוג integer ו-string מגדיר את מספר הרווחים (התווים הלבנים) לפני הטקסט.

שדה עריכה עבור משתנים מסוג Real הוא שונה כיוון שכאן אנו מתייחסים לשתי דברים:
1. גודל שדה העריכה (מספר התווים הלבנים שלפני הטקסט)
2. מספר הספרות שיופיעו אחרי הנקודה העשרונית

בשביל להבהיר זאת נשתמש בקטע הקוד הבא:

```
begin
      writeln(8/7:20:3);
end.
```

הסבר: השדה הראשון (20:) מתייחס למספר התווים הלבנים לפני הטקסט הכתוב ואילו השדה השני (3:) מתייחס למספר הספרות שיופיעו אחרי הנקודה העשרונית.

דבר נוסף הוא אם נעשה חלוקה באפס אז תיווצר שגיאה לכן צריך לעשות הוראה מקוננת .

הוראה מקוננת

```
If action='/' then
      If num2<>0 then
            Writeln('the result is ',num1/num2)
      Else
            Writeln('you can't do that' );
```

בהתחלה אנחנו בודקים אם התו הוא חלקי אם כן בודקים שהמספר שונה מ-0. אם כן הוא כותב הודעה את התשובה ואם לא (אחרת) אז הוא כותב שיש שגיאה בקלט.

תוכנית מתוקנת

```
program v31;
var num1,num2:integer;
action:char;
begin
writeln('type two numbers');
readln(num1,num2);
writeln('type an action');
readln(action);
if action='+' then
writeln('the result is ',num1+num2)
else
if action='-' then
writeln('the result is ',num1-num2)
else
if action='*' then
writeln('the result is ',num1*num2)
else
if action='/' then
if num2<>0 then
writeln('the result is ',num1/num2)
else
writeln('you cant...')
else
writeln('the the action is error');
readln;
end.
```

הסבר:
בתוכנית הנ"ל השתמשנו בתנאים המורכבים גם בפקודה אחרת. עכשיו בואו נעקוב אחר התוכנית הנ"ל.
אנו קולטים שני מספרים ובוחרים בפעולה שנו רוצים. לאחר אנו מגיעים לתנאי הראשון אם זה חיבור אם התנאי לא חיבור אז ממשיכים הלאה לחיסור ואם זה לא חיסור אז ממשיך הלאה. בסופו של דבר אם הפעולה היא לא אחת מהפעולות החשבוניות . התוכנית פולטת ,הפעולה היא שגיאה.

כמו שראיתם בתנאים אפשר לא רק לעשות שווה אפשר גם לעשות פעולות השוואתיות אחרות לפי הטבלה הבאה.

פעולה	הסבר	דוגמא
=	שווה ל...	X=8
<>	שונה מ...	X<>8
<	קטן מ...	X<8
>	גדול מ...	x>8
<=	קטן או שווה ל...	X<=8
>=	גדול או שווה ל...	x>=8
Not	שלילת תנאי	If not X=8 then
IN	בתחום מסוים	If x in [1..5] then
And	ו...(שתי התנאים)	If x=8 and y>9 then
Or	או... (תנאי אחד או יותר)	If x>8 or y<9 then
Xor	רק אחד נכון	If x>9 xor y>9 then
not	לא(שליללת תנאי)	If not(x>9) then

ה-NOT זה בעצם לוגיקה הפוכה לדוגמה NOT X>8 = X<8 .

כפי שציינו בהתחלת הפרק, ישנו תנאי נוסף הנקרא case.

אפשר לעשות את המחשבון כך :

```
Case action of
'+' : writeln ('the result is ' , num1+num2);
'-' : writeln('the result is ' ,num1-num2 );
'*' : writeln('the result is ' ,num1*num2);
'/' : begin
 if num2 <> 0 then
Writeln('the result is ,num1/num2);
End;
Else
Writeln('error');
End;
End.
```

הסבר תוכנית:

case….of זוהי פקודה שלפיה אפשר לבדוק תנאים רבים בנוחיות רבה יותר. בהתחלה הגדרנו ש Action זהו המשתנה הנבדק למקרים של חיבור,חיסור,כפל וחילוק. כאשר הגענו לפעולת החילוק בדקנו

האם המספר השני הוא אפס (במתמטיקה אסור לחלק מספר כלשהו באפס כי אם תחשבו על זה, זה די חסר היגיון הרי חלקי מייצג את מספר הפעמים שמספר מסוים נכנס במספר אחר. אם כך כמה פעמים 0 שזה כלום נכנס במספר כלשהו?????? הרי מחלקים בכלום!!!!.)
ואם כן אז התוכנה לא מאפשרת פעולת חילוק וכותבת, טעות מתמטית.
לאחר שנבדקים כל התנאים הללו וכולם לא מתבצעים אז מגיע התנאי האחרון שאומר שאם הפעולה שנבחרה היא לא פעולה חשבונית אז נכתב, יש טעות.

משתנים בוליאנים

כאשר מגדירים מספר מסוג boolean יש אפשרות להציב בו true או false.
יש אפשרות להציב גם בו תנאי לדוגמא :

```
Var d:Boolean;
Begin
D:= 5<8;
End.
```

ולהשתמש בו אחרי זה בתנאים כי התשובה של התנאי תוצב במשתנה :אמת או שקר.

סדר פעולות לוגיות :

1) () סוגרים
2) NOT
3) And
4) Or

הסבר : כאשר משתמשים בפעולות האלו יש סדר כמו במתמטיקה וזהו הסדר.

תרגילים לחזרה

1)כתוב תוכנית הקולטת שלושה מספרים ומדרגת אותם לפי סדר עולה (מהקטן לגדול) ובנוסף בודקת את הממוצע של המספרים החיוביים מביניהם.

2)א. בית-הספר יוצא לטיול, בכל אוטובוס יש מקום ל-30 איש ובמונית ל-10 אנשים. עליך לקלוט את מספר האנשים שיוצאים ולכתוב כמה מוניות ואוטובוסים צריך לטיול.
ב. אוטובוס עולה 200 שקל ליום ואילו מונית עולה 100 שקל ליום עליך לקלוט לכמה ימים הטיול ובהתאם לכתוב את עלויות הנסיעה ואם עלות הנסיעה גדולה מ-1500 שקל אז פלט בודקים חברה אחרת.

3) לפניך קטע תוכנית:

```
Begin
      Readln(x);
      If x>8 then
            Y=x*x
      Else begin
            X=x*x;
            Y=0;
      End;
      Readln(z);
```

```
      If z=y then
            Z=z+1;
      Writeln(x,y,z);
End.
```

א.לפניך הקלט ,ערוך טבלת מעקב וכתוב מה הפלט הסופי

X:=8,Z:=3

ב. לפניך הקלט ,ערוך טבלת מעקב וכתוב מה הפלט הסופי

X:=3,z:=0

ג. לפניך הקלט ,ערוך טבלת מעקב וכתוב מה הפלט הסופי

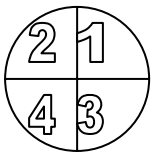
X:=3,.z:=1

4) לפניך קטע תוכנית :

```
Begin
      Readln(n);
      If n<10 then
            Writeln(n)
      Else
            If n>10 and n<100 then
                  Writeln( n mod 10 ,n div 10);
            Else
            If n>100 and n<1000 then
                  Writeln(n mod 10,(n div 10) mod 10,n div 100);
            Else
                  Writeln('the number too big');
```

עשה טבלת מעקב הבאים : 3,100,3000,45,91,18
מה מטרת התוכנית? והפלט של כל אחד מהקלטים?

5) כתוב תוכנית הקולטת זווית במעגל אם הזויות לא נכונה אז תהיה הודעת שגיאה אחרת תוחזר באיזה רביעי היא נמצאת במעגל כמו בדוגמא הבא :



פתרונות לתרגילי החזרה

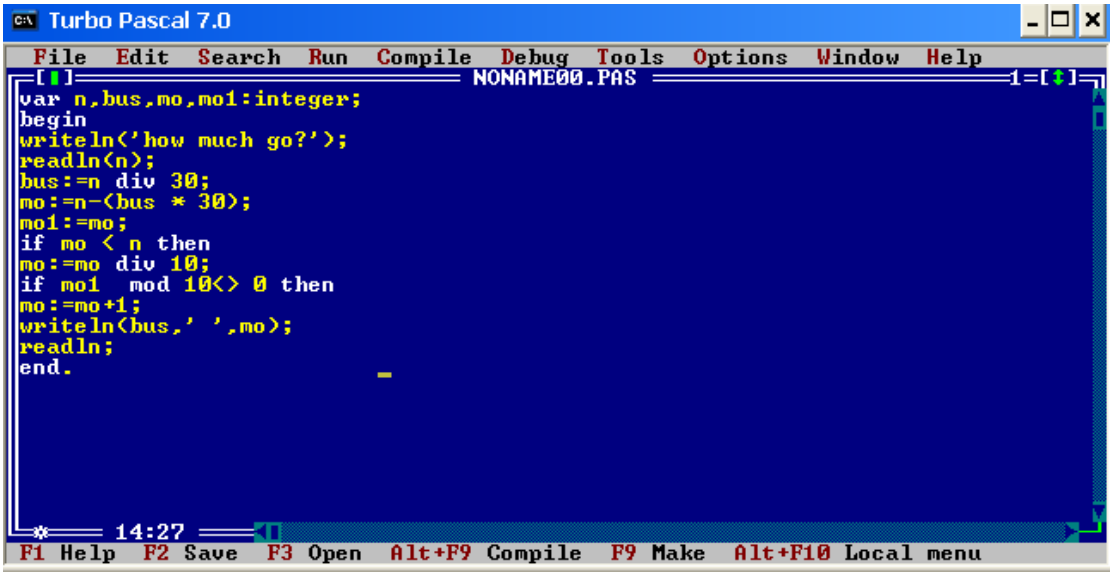
תרגיל מספר אחד:
דבר ראשון בואו נחלק את התוכנית לתת-משימות
תת-משימות:
1. דירוג מספרים לפי סדר עולה.
2. בדיקת הממוצע של המספרים החיוביים

לפיכך אני צריך ארבעה משתנים 3 משתנים לקליטת המספרים ומשתנה אחד לחישוב הממוצע.

התוכנית:

```
Program targil1;
Var num1,num2,num3:integer;
Average: real;
Begin
Writeln('enter 3 numbers');
Readln(num1,num2,num3);
If (num1>num2) and (num1>num3) then
     If num2>num3 then
     Write(num3,num2:2,num1:2)
     Else
     Write (num2,num3:2,num1:2);
If (num2>num1) and (num2>num3) then
begin
If num1>num3 then
Write(num3,num1:2,num2:2)
If num3>num1 then
Write(num1,num3:2,num2:2)
End;
If (num3>num1) and (num3>num2) then
If num2>num1 then
Write(num1,num2:2,num3:2)
Else
Write(num2,num1:2,num3:2);
End.
```

2) א.



הסבר:
הגדרנו 4 משתנים:

- n – כמות האנשים שנוסעים
- bus – כמות האוטובוסים
- Mo – כמות המוניות
- Mo1 – כמות האנשים במונית (כדי לבדוק אם צריך להזמין עוד אחת)

בהתחלה קולטים כמה אנשים נוסעים, אחרי זה מחלקים ב 30 (לא חילוק רגיל כדי שלא ישאר שארית) עושים חילוק שלם וככה יהיה בדיוק כמה אוטבוסים צריך אחרי זה בודקים כמה מוניות (כמות האנשים באוטבוס – הכמות הכללית זה הכמות שצריכה להיות במוניות) אחרי זה מחלקים ב10 שלם ורואים כמה מוניות צריך אחרי זה בודקים אם נשאר שארית אם כן אז מוסיפים עוד מונית .

חלק ב של השאלה :

```
var n,bus,mo,mo1,day,money:integer;
begin
writeln('how much go?');
readln(n);
bus:=n div 30;
mo:=n-(bus * 30);
mo1:=mo;
if mo < n then
mo:=mo div 10;
if mo1  mod 10<> 0 then
mo:=mo+1;
writeln(bus,' ',mo);
writeln('how much days?');
readln(day);
money:=(bus * 200)+(mo *100)*day;
if money >1500 then
writeln('too much money');


end.
```

הסבר:

קלטנו לכמה ימים וחישבנו כמה עולה אם יותר מ1500 אז כותבים הודעה .

3)
א)

פלט	Z	Z=y	Y	x>8	X
				false	8
	3	false	0		64
64 ,0 ,3					

ב)

פלט	Z	Z=y	Y	x>8	x
				false	3
	0		0		9
9 0 1	1	true			

ג)

פלט	Z	Y=z	Y	x>8	X
				false	3
			0		9
3 9 0					

5) פתרון:

רביע ראשון- מזווית 0 עד ל-90. רביע שני- מזווית 91 עד 180.
רביע שלישי- מזווית 181 עד 270. רביע רביעי מזווית 271 עד 360.

אחרי שהבנו את גבולות הזוויות בכל רביע ניגש עתה לתקינות הזוויות. הזוויות שיכול להיקלט הם בין 0 ל-360 ואלו בעצם יהיו גבולות המינימום והמקסימום שלנו.

תוכנית:

```
Program targil5;
Var num:integer;
Begin
Writeln('enter zavit');
Readln(num);
While num not in [0..360] then
begin
Writeln('erroe , please enter a zavit again');
Readln(num);
End;
If num in [0..90] then
Writeln('square 1');
If num in [91..180] then
Writeln('square 2');
If num in [181..270] then
Writeln('square 3');
If num in [271..360] then
Writeln('square 4');
End.
```

פרק 4: לולאות

נגיד ואנחנו רוצים לקלוט 100 ציונים של תלמידים, הרי לא נכתוב 100 פעמים readln.

אז מה נעשה? בשביל זה יש לנו את אחד מהכלים הכי חזקים בעולם התכנות לולאות או באנגלית loops.
הלולאה הראשונה שנלמד היא לולאת for.

FOR

הלולאה הזאת היא אולי הכי קלה להבנה ונתחיל אתה. ובכן המבנה התחבירי של הלולאה הולך ככה:
עבור (משתנה)=מספר עד ל-מספר אחר בצע.

בפסקל זה ייראה כך:

```
Var I :integer;
begin
      For i:=1 to 5 do
      Writeln('6');
End.
```

בואו נעקוב אחר הלולאה ונבין מה היא עושה.

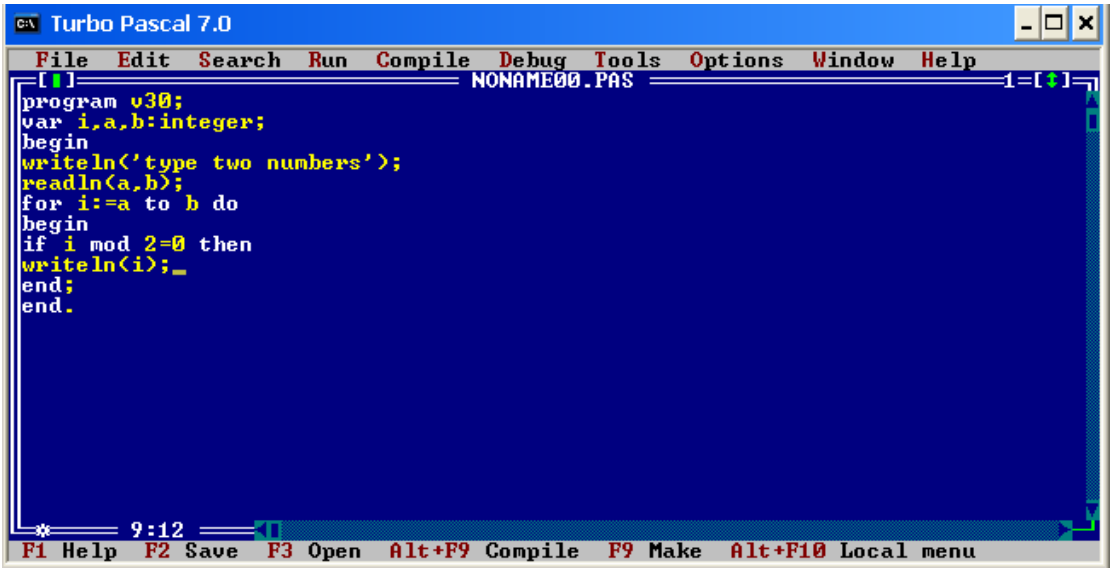
פלט	For i:=1 to 5 do
1	1<=5 (true)
2	2<=5(true)
3	3<=5(true)
4	4<=5(true)
5	5<=5 (true)
אין פלט.	6<=5 (false)

*הערה: כאשר אנו משתמשים בלולאה הנ"ל הערך של המשתנה עולה כל פעם בספרה אחת כלפי מעלה אוטומטית בלי שנגדיר זאת.(לכן היא נקראת לולאה אוטומטית)
נגיד המשתנה מושווה לאחד הערך הבא יהיה שתיים ואחריו שלוש וכך הלאה...

הסבר לקטע הקוד (על-פי טבלת המעקב):

בהתחלה השוונו את המשתנה לאחד ואמרנו מהערך 1 עד 5 בצע את ההוראה הבאה (שבמקרה הזה היא לפלוט טקסט).
1<=5 ההשוואה הזאת נכונה ולכן המהדר המשיך הלאה וכתה את הטקסט.
2<=2 ההשוואה הזאת נכונה ולכן המהדר המשיך הלא וכתב את הטקסט.
3<=5 ההשוואה הזאת נכונה ולכן המהדר המשיך הלאה וכתב את הטקסט.
4<=5 ההשוואה הזאת נכונה ולכן המהדר המשיך הלאה וכתב את הקסט.
5<=5 ההשוואה הזאת נכונה ולכן המהדר המשיך הלא וכתב את הטקסט.
6<=5 ההשוואה הזאת לא נכונה ולכן הלולאה הופסקה וכך גם התוכנית דוגמא כיוון שאין שום דבר שכתוב אחר הלולאה.

תוכנית V30



הסבר:
התוכנית קולטת שתי מספרים ופולטת את כל המספרים הזוגיים בין המספרים שנקלטו
כדי לבדוק שאם הם זוגיים, עשינו את התנאי הבא:

```
If I mod 2=0 then
```

התשובה תהייה 0 אם הם זוגיים!!

למי שלא הבין נעשה טבלת מעקב שתעזור להסביר נגיד ונקלטו 0 ו-5.

a	b	i	I mod 2=0	פלט
0	5			
0	5	1	1 mod 2=0 false	
0	5	2	2 mod 2=0 true	2
0	5	3	3 mod 2=0 false	
0	5	4	4 mod 2=0 true	4
0	5	5	5 mod 2=0 false	

הפלט הוא רק הזוגיים.

האפשרות Downto

יש אפשרות בפסקל גם לרדת למטה בלולאה בסדר יורד (לדוגמא מ-10 עד 1) ולא כמו שעשינו קודם בעזרת השימוש ב- downto.

ת'כלס השינוי העיקרי הוא שבמקום מלמטה למעלה אנו כרגע הולכים מלמעלה למטה אז זה לא אמור לשנות הרבה. תוכנית דוגמא.

```
Program v……;
Var I:integer;
Begin
      For I:=10 downto 1 do
      Writeln(i);
End.
```

נגיד ואנחנו רוצים לעשות ממוצע של ציונים אבל אנחנו לא יודעים כמה תלמידים יש מראש
איך נעשה את זה נפרק את הבעיה הזאת ל-3 שלבים
דבר ראשון אנחנו צריכים לקלוט כמה תלמידים יש

```
Writeln('how much pupils in school?');
Readln(n);
```

דבר שני אנחנו צריכים לעשות סיכום של כל הציונים (ממוצע זה סכום כולם חלקי הכמות שלהם)
בשביל זה אנחנו צריכים משתנה לסיכום נגדיר משתנה שנקרא לו
SUM דבר ראשון נאפס אותו כי הסכום בהתחלה הוא 0 (צריך מספר כדי להתחיל) ואחרי זה נעשה סכום של כל הציונים

סיכום

```
Sum:=0;
For I:=1 to n do
Begin
Writeln('type a mark of pupil number:',i);
Readln(mark);
Sum:=sum+mark;
End;
```

ובסוף נחלק את זה במספר התלמידים

```
Avg:=sum/n;
```

עכשיו התוכנית המלאה:

```
Program v31;
Var n,I,mark:integer;avg:real;
begin
Writeln('how much puples in school?');
Readln(n);
Sum:=0;
For I:=1 to n do
Begin
Writeln('type a mark of pupil number:',i);
Readln(mark);
Sum:=sum+mark;
```

```
End;
Avg:=sum/n;
Writeln('the avg is:',avg);
End.
```

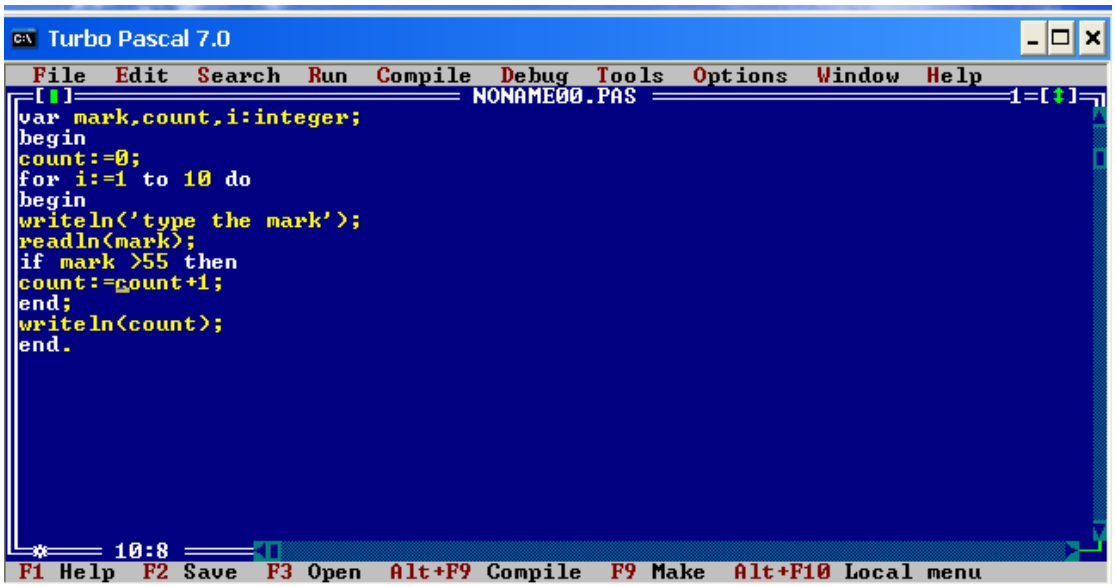
- הגדרנו את AVG כמספר ממשי, הממוצע יכול להיות גם עשרוני ומציבים בו חלוקה של שני מספרים לכן התשובה תמיד עשרונית!!!
- למשתנה סיכום לא חייב לקרוא sum אבל זה מקובל ככה.

טוב למדנו לסכם מספרים עכשיו מה קורא כאשר אנחנו רוצים לספור לדוגמא כמה ציונים גדולים מ-55 בכיתה, מה עושים?

ספירה

בשביל זה נגדיר משתנה ונקרא לו : Count ונספור. לדוגמא בתוכנית הבאה:

תוכנית V3.2



בתוכנית קלטנו 10 ציונים וכל פעם שהציון גדול מ-55 ספרנו אותו !!!

מכפלה

נגיד ואנחנו רוצים לעשות את המכפלה של 8 מספרים מה עושים?. מגדירים משתנה ונקרא לו MAH בהתחלה נציב בו 1 כדי שמתי שנכפול אותו, הוא לא יתאפס (0* מספר=0) ובפעם הראשונה אנחנו רוצים שהוא יהיה שווה למספר הראשון וזה יקרה במקרה שנגדיר שהוא אפס לפני הלולאה שבה קולטים את המספרים .

תוכנית V3.3

דוגמא קלאסית לשימוש במכפלה הוא עצרת : עצרת זה המכפלה של כל המספרים בין 1 למספר אחר (!- מסמל עצרת) לדוגמא 3! זה 1*2*3 .

```
Program v33;
Var I,mah,num:integer;
Begin
      Writeln('type a num');
      Readln(num);
      Mah:=1;
      For I:=1 to num do
            Mah=mah *i;
      Writeln(mah);
End.
```

כדי להבין טוב יותר נעשה טבלת מעקב לתוכנית הזאת נגיד ו הקלט הוא 5 :

פלט	I	Num	Mah
		5	1
	1	5	1*1
	2	5	1*2=2
	3	5	2*3=6
	4	5	6*4=24
	5	5	24*5=120
הפלט הוא 120			120

מקסימום

שימוש נוסף ללולאות הוא מציאת מינימום ומקסימום. דוגמא: מציאת הציון הגבוה ביותר בכיתה.

```
Max:=0;
For i:=1 to 10 do
Begin
Writeln('type the mark');
Readln(mark);
If mark >max then
Max :=mark;
End;
Writeln(max);
```

אנחנו מציבים 0 במשתנה מקס בהתחלה כי הציון המינימלי שיכול להיות הוא 0 אחרי זה כל פעם שאנחנו קולטים ציון אנחנו בודקים אם הוא גדול מהציון הגבוה ביותר שזהו בעצם המקסימום ,בסופו של דבר יוצב בו המספר(ציון) הגבוה ביותר.
כאשר יש מספרים שלילים ישנה בעיה כי המספר המינימלי יכול להיות קטן מאפס אז מה נציב במשתנה Max בהתחלה? יש מספר שיטות, נציג לכם אחת שטובה תמיד:

```
Writeln('type a mark');
Readln(max);
For i:=2 to 10 do
Begin
```

```
Writeln('type the mark');
Readln(mark);
If mark >max then
Max:=mark;
End;
Writeln(max);
```

אנחנו מציבים במשתנה מקס את המספר הראשון שנקלט וכך אנחנו בודקים אם היה גדול יותר.
ואנחנו מתחילם את הלולאה מ-2 כי כבר קלטנו מספר.

מינימום

מינימום זה בדיוק הפוך אנחנו קולטים בהתחלה את המספר הראשון ובודקים אם המספר הבא קטן יותר :

```
If mark <min then Min:=mark;
```

השוואה לסיכום (כל הנושאים)

שם	שימוש	הגדרה בהתחלה
Sum	כדי לעשות סכום של מספרים	לרוב מגדירים 0 כי מתחילים לסכם מ-0
Count	כדי לספור	לרוב 0 כי סופרים מ-0
Mah	כדי לעשות מכפלה	1
min	מספר מינימלי	קליטת המספר הראשון בו
max	מספר מקסימלי	קליטת המספר הראשון בו

עכשיו נציג תוכנית שמשלבת את מה שלמדנו .

תוכנית V3.3

בתוכנית נקלוט 40 משכורות של עובדים ונבדוק מה המשכורת הממוצעת המינימלית ,המקסימלית וכמה משכורות מעל 8000 .

```
begin
sum:=0;
count:=0;
writeln(' type the mascor'):
readln(mas);
min:=mas;
max:=max:
sum:=sum+mas;
if mas > 8000 then
count:=count+1;
for i:=2 to 40 do
begin
writeln(' type the mascor'):
readln(mas);
if mas >max then
max=mas;
if mas<min then
mas=min;
sum:=sum+mas;

if mas >8000 then
count:=count+1;
end;
avg=sum/40;
writeln('the min is:,min,' the max is :',max,' the avg is: ',avg);
writeln('there are ',count,'that bigger then 8000');
end.
```

Repeat

repeat זאת לולאה נוספת. המבנה שלה:

```
Reapet
הוראות
Until (תנאי)
```

בלולאה זאת אין צורך בבלוקים כי עד שלא נכתוב את התנאי, הכל בתוך הלולאה לדוגמא:

```
I:=0;
Reapet
I:=i+1;
Writeln('1');
Until (i:=5)
```

הלולאה שבדוגמא בדיוק כמו הלולאה הבאה:

```
For I:=1 to 5 do
Writeln('1');
```

אז אם כן השוני ? בלולאת זו אפשר לעשות תנאי שהוא בעצם אינו ידוע מראש למשל עד שנקלט מספר גדול מ-5. לדוגמא:

```
Sum:=0;
Reapet
Readln(num)
 Sum:=sum+num;
Until (sum <400)
```

הלולאה תתבצע עד שלא יהיה סכום שגדול מ- 400 .

*הערה: לולאה אינסופית: כאשר נעשה תנאי שאף פעם לא ייגמר אז זה יגרום לשגיאה!!! לדוגמא:

```
Reapet
I:=0
Until (i>2)
```

אף פעם זה לא יקרה!!!

לולאת While

לולאת While זוהי לולאה מאוד שימושית כיוון שניתן להפסיק אותה באמצע ויש דברים שלא ניתן לעשות בלולאה for.
כמו כן היא משמשת לשדה קליטה ושימוש ב-flag.

*הערה- כל מה שניתן לעשות בלולאה for ניתן לעשות גם בלולאה while. אך לא כל מה שניתן לעשות בלולאה while ניתן לעשות בלולאה for .

כמו בכל לולאה גם ללולאה while צריך להיות תנאי.

המבנה התחבירי (המילולי) של הלולאה while:
כל עוד משתנה (תנאי) מספר בצע.
.....
כל עוד איקס שווה ל-8 בצע.

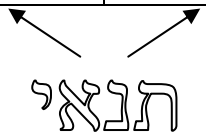
.....

בואו נראה דוגמא ללולאה מסוג while:

```
Var x,I:integer;
begin
I:=0;
X:=2;
While I<>1 do
Begin
If x=2 then
I:=1;
End;
End.
```

טבלת מעקב:

i	x	כל עוד I<>1 בצע	אם X=2 אז	i
0	2	(I=0)<>1 (true)	(x=2)=2 (true)	I=1
1	2	(I=1) <> 1 (true)	----------	----------



אם כך למדנו כיצד עובדת הלולאה while אולם בואו ננסה תרגול נוסף. אנו יודעים שהלולאה חוזרת על הפקודות שבתוכה עד שהתנאי בראש הלולאה מופר. כאשר אנו עושים לולאה while והתנאי מתקיים אז בסוף הלולאה המחשב עולה אל ראש הלולאה ובודק את התנאי. אם התנאי מתקיים אז עוד פעם הוא מיישם את הפקודות שבתוך הלולאה ועולה כלפי מעלה ובודק את התנאי וחוזר חלילה.

אם כך בואו נעשה תוכנית נוספת שתסביר את תפקודה של לולאת while כאשר אנו צריכים לעלות את משתנה הלולאה. למה אני מתכוון??? הדוגמא הבאה תסביר את זה טוב.

הוטלה עלינו משימה. לבנות לולאה שתפלוט את המספרים הזוגיים בין 1 ל-100. בואו נעשה זאת ביחד בלולאת while. התוכנית:

תת-משימות:
1.אנו צריכים משתנה לולאה.
2. אנו צריכים לפלוט את המספרים הזוגיים בין 1 ל-100.

```
Program …….;
Var I:integer;
Begin
I:=2;
{ אנו נכניס את הערך 2 למשתנה הלולאה כיוון ש-1 הוא אי-זוגי אז אין טעם
.להתחיל ממנו}
While I<=100 do
{ כל עוד משתנה הלולאה קטן שווה למאה בצע}
Begin
```

עכשיו אנו נתקעים פה. איך אנו נגיע בכלל למאה אם משתנה הלולאה לא גודל ונשאר שתיים??? אם לא נשנה את משתנה הלולאה, הלולאה תהיה אינסופית וזה בפשטותו באג בתוכנית!!!. לכן אנו צריכים לעלות את משתנה הלולאה בתוך הלולאה. וזה הרעיון בלולאת while.

```
Writeln(i);
{ על מנת לקצר את פעולת הלולאה נעלה את משתנה הלולאה בשתיים כיוון שהוא
{מתחיל מהערך שתיים וכך כל הערכים הבאים יהיו זוגיים
```

```
I:=I+2;
End;
```

למדנו כיצד להשתמש בלולאה while אך ללולאה הזו יש עוד הרבה מאוד שימושים אחד מהם שנלמד עכשיו הוא דגל.
ובכן מהו בעצם דגל???:

דגל

דגל (flag)- זהו משתנה מסוג שלם שעושה את פעולתו של משתנה בוליאני. למדנו שמשתנה בוליאני הוא משתנה המכיל את הערכים אמת ושקר וכך גם פועל הדגל רק שאצל הדגל 1 זה אמת ו-0 זה שקר (אלו לא ערכים קבועים למשל אני יכול להגדיר ש-2 זה אמת ו-1 זה שקר אך נהוג להשתמש בו כך).
עכשיו בטח תשאלו את עצמכם למה זה טוב??? ובכן אם נגיד אני רוצה לבדוק האם נקלטה הספרה מאה אני לא יחכה עד שהלולאה תסתיים (הרי זה לא הגיוני להמשיך אם כבר נקלט הערך המבוקש) לכן אני אשתמש בדגל ואעצור את הלולאה.

אחרי שהבנו מהו דגל בואו נראה כיצד זה מיושם.

משימה: עליך לבנות תוכנית הקולטת מספרים חיוביים ואם המשמש קלט מספר שלילי אז לולאת שדה הקליטה נפסקת.

התוכנית:

```
Program v……;
Var flag,num:integer;
Begin
Flag:=0;
While flag<>1 do
begin
Writeln('enter a number');
Readln(num);
If num<0 then
Flag:=1;
End;
```

הדגל בהתחלה הוא 0 כיוון שהתנאי או המקרה המסויים לא התקיים (במקרה שלנו המקרה הוא הכנסת מספר שלילי) וכאשר הדגל שווה לאחד אז המקרה התקיים והלולאה נפסקת.

שדה קליטה

על מנת לקבל את הנתונים הרצויים עלינו לעשות שדה קליטה. למה אני מתכוון???, ובכן נגיד ויש לי תוכנה שמכינה את תעודת סוף השנה לתלמידים, ובמקום שהמורה יכתוב 85 הוא כתב 485 ובכן ציון כזה הוא לא הגיוני לכן יש מה שנקרא שדה קליטה על מנת למנוע טעויות קלט מהמשתמשים. שדה קליטה חייב להיות בתוך לולאה ולא בתנאי כיוון שהמשתמש יכול לעשות 2 טעויות קליטה ברצף ויותר לכן אנו זקוקים ללולאה.

הלולאה שבעזרתה אנו עושים שדה קליטה היא הלולאה while (ורק בלולאה הזו כי בלולאות אחרות זה בלתי אפשרי).

בואו נראה תוכנית דוגמא:

משימה: עליך לקלוט 3 גילאים של אנשים ולכתוב את ממוצע הגילאים שלהם.

ובכן גילו של אדם בואו נגיד לא יכול לעלות על 125 שנה ולא יכול להיות קטן מ-0 לכן זה יהיה הגיל המקסימלי והמינימלי שלנו בשדה העריכה.

בואו נראה את זה בתוכנית הדגמה:

```
Program v…….;
Var sum,I,age:integer;
Begin
Sum:=0;
For i:=1 to 3 do
begin
Writeln('enter your age');
Readln(age);
While (age<0) and (a>125) then
begin
Writeln('enter your age');
Readln(age);
End;
Sum:=sum+age;
End;
Writeln('the average of ages is ',sum/3);
End.
```

בואו נסתכל על התוכנית, ניתן לראות שהקליטה של המשתנה age היא לפני שדה הקליטה כיוון שאם אנו לא נעשה זאת לפני שדה הקליטה אז לא יהיה ערך למשתנה age לכן הקליטה של המשתנים נעשית לפני שדה הקליטה.

בואו נסתכל עכשיו על הקטע המסומן באדום. ניתן לראות שהגבול המינימלי של שדה הקליטה הוא 0 והגבול המקסימלי של שדה הקליטה הוא 125. לכן אם המשתנה יחרוג מגבולות אלו אז אנו ניכנס ללולאת שדה הקליטה ועד שלא נכתוב ערכים בין הגבול המקסימלי (125) לגבול המינימלי (0) התוכנית לא תמשיך הלאה.

בואו נראה דוגמה לשדה קליטה בפני עצמו.

```
I:=0;
While i<>8 do
Begin
Writeln('enter number');
Readln(i);
End;
```

הסבר שדה הקליטה:

כל עוד המספר שהמשתמש יכניס שונה מ-8 לולאת שדה הקליטה לא תיפסק.

תרגילים

1) כתוב תוכנית שתשרטט מלבן כאשר התוכנית תקלוט שתי נתונים 1. גובה מלבן 2. אורך המלבן.
2) כתוב תוכנית שתשרטט משולש שווה שוקיים כאשר התוכנית תקלוט נתון 1 1. גובה המשולש או 2. אורך המשולש (זה לא משנה).

פתרונות

מלבן: אנו צריכים ארבע משתנים:
1. קליטת נתון הרוחב.-זה יהיה איקס כמו במתמטיקה (משתנה מסוג שלם אינטיג'ר)
2. קליטת נתון הגובה.- זה יהיה y כמו במתמטיקה (משתנה מסוג שלם אינטיג'ר)
3. משתנה לולאה ראשון- זה יהיה I. (משתנה מסוג שלם אינטיג'ר)
4. משתנה לולאה שני- זה יהיה j. (משתנה מסוג שלם אינטיג'ר)

התוכנית עצמה:

```
Program targil1;
Var x,y,I,j:integer;
Begin
Writeln('enter width');
Readln(x);
Writeln('enter hight');
Readln(y);
For I:=1 to y do
Begin
For j:=1 to x do
Write('x');
Writeln('');
End;
End.
```

משולש הפוך: במקרה של המשולש שווה שוקיים אנו צריכים לקלוט או גובה או רוחב ושתי משתני לולאה אני מעדיף לקלוט את הרוחב (בלי שום סיבה מסויימת כי ת'כלס זה לא משנה).
x הינו הרוחב. משתני הלולאה הם I, j.

```
Program v……..;
Var x,I,j:integer;
Begin
Writeln('enter the width');
Readln(x);
For I:=x downto 1 do
begin
For j:=1 to I do
Write('*');
Writeln;
End;
End.
```

לולאה אינסופית

אם נעשה תנאי בלולאת WHILE או REAPET שהוא תנאי שאף פעם לא ישתנה ותמיד יהיה נכון אז הלולאה לא תיגמר אף פעם והתוכנית בעצם תתקע .

```
While x<>0 do
Begin
X=1;
End;
```

ה- X תמידי יהיה שונה מ-0 ולכן יש לולאה אינסופית.

לולאות מקוננות

ישנה אפשרות לעשות לולאה בתוך לולאה לדבר זה יש מספר שימושים.

דוגמא: נניח ויש לנו שתי כיתות ובכל אחת 2 תלמידים איך בודקים מה הציון הגבוה ביותר בכל כיתה ומה הציון הגבוה ביותר בבית ספר:

```
Max :=0;
maxAll:=0;
For I;=1 to 2 do
begin
For j:=1 to 2 do
Begin
Readln(mark);
If mark >max then
Max:=mark
End;
If max >maxall then
Maxall:=max;
Writeln(max)
Max:=0;
End;
Writeln(maxall)
```

כדי להבין יותר נעשה טבלת מעקב לנתונים הבאים:
כיתה 1 :ציון:80 ציון2: 50
כיתה 2: ציון: 70 ציון 100

I	J	Max	Maxall	mark	פלט
1	1	80	0	80	
1	2	80	0	50	80
2	1	70	80	70	
2	2	100	100	100	100
		100	100	100	100

הסבר:
בהתחלה נקלט המספר-80 הוא גדול מ-0 לכן הוא הוצב במשתנה מקס, אחרי זה נקלט 50 הוא קטן מהמשתנה מקס אז ממשיכים אחרי זה נגמרה הלולאה יש פלט והמשתנה מקס-הכללי יוצב 80 כי לפני זה הוא -0 , גם מאפסים את המשתנה מקס כדי שלכיתה השניה יהיה מקסימום חדש אחרי זה נקלט 70 שזה גדול מ-0 ובסוף נקלט 100 שהוא יוצב במשתנה מקס וגם בכללי כי זה הציון הגבוה ביותר בבית ספר.

*לולאה מקוננת תהייה שימושית מאוד בהמשך בפרק של מערכים רב מימדיים לכן חשוב להבין אותה!!

תרגילים לסיכום

1)כתוב תוכנית הקולטת 5 פעמים 2 מספרים וכותבת ביניהם שווה או שונה
2) כתוב תוכנית הקולטת עבור 2 מחלקות שונות משכורות של עובדים בכל מחלקה 80 עובדים אם סכום הממוצע של עובד קטן מ -400 שקל פלט מפעל מנצל ,עבוד כל מחלקה . וסכום כל המשכורות בכל מחלקה והשווה ביניהם איזה מחלקה מרוויחה יותר .
3)כתוב תוכנית הקולטת את המשכורת של 50 עובדים ומדפיסה
א. אם המשכורת מעל 100000 תודפס מרוויח יותר מידי
ב. מדפיסה את סכום כל המשכורות
4) כתוב תוכנית הקולטת מספרים עד שנקלט מספר 0 ותבדוק אם נקלט מספר זוגי או לא .
5)כתוב תוכנית הקולטת מספר שלם ובודק אם הוא ראשוני או לא (ראשוני נתחלק רק ב1 ובעצמו) ובהתאם הודעה .

פתרונות

(1

```
Var n1,n2,i:integer;
Begin
For I=1 to 5 do
Begin
Writeln('plz enter two numbers');
Readln(n1.n2);
If n1=n2 then
Writeln('equal')
Else
Writeln('different');
End;
End.
```

(2

```
Var i:integer;
Sum1,sum2,d1,d2:real;
Begin
Sum1:=0;sum2:=0;
For I;=1 to 80 do
Begin
Writeln('enter how much you earn');
Readln(d1);
Sum1:=sum1+d1;
End;
For I;=1 to 80 do
Begin
Writeln('enter how much you earn');
Readln(d2);
Sum2:=sum2+d2;
End;
If sum1>sum2 then
Writeln('division 1 earn more then 2')
Else
Writeln('division 2 earn more then 1');
If (sum1+sum2)\ 160 < 4000 then
Writeln('they are bad people');
End.
```

(3

```
Var mas,sum,i:integer;
Begin
Sum:=0;
For I=1 to 40 do
Begin
Writeln('how much you earn');
Readln(mas);
Sum:=sum+mas;
If mas> 1000000 then
Writeln('you earn too much');
End;
Writeln('sum:',sum);
End.
```

(4

```
Var flag,num:integer;
Begin
Flag:=0;
Writeln('enter number'):
Readln(num);
While num<>0 do
Begin
If num mod 2 = 0 then
Flag: =1;
Writeln('enter number');
Readln(num);
End;
If flag= 0 then
Writeln('no zugi')
Else
Writeln('there was zugi'):
End.
```

(5

```
Var flag,num,i:integer;
Begin
Flag:=0;
Writeln('type a number');
Readln(num);
For i:=2 to num/2 do
Begin
If num mod I >0 then
Flag:=1;
End;
If flag=0 then
Writeln(' no rishon')
else
writeln('yes rishoni');
end.
```

פרק 5: טיפוסי נתונים

בפסקל ישנם אפשרויות נוספות לשמור מידע חוץ ממשתנים, בפרק זה אנו נסביר עליהם בהרחבה.

קבועות (Const):

הוא בעצם משתנה שמגדירים לו ערך לפני תחילת התוכנית והערך הזה נשמר במהלך הרצת התוכנית.
לקבוע יש שתי שימושים
1. לייפות את התוכנית- לדוגמא אם אנחנו רוצים להציב בקבוע את המספר 100 נקרא לו איקס ובכל פעם שנרצה להשתמש במספר מאה נוכל לקרוא לאיקס. הגדרת שמות של קבועים זה בדיוק כמו הגדרת שמות של משתנים.
2. לדוגמא בכיתה כלשהי יש 40 תלמידים והרבה פעמים אנחנו משתמשים במספר ארבעים ישצרנו תוכנית ולאחר מכן גילינו שנוסף תלמיד אחד לכיתה. במקום כל פעם את המספר 40 ל-41 ניתן לשנות את הקבוע ביתר קלות.

ישנם שני סוגים של קבועים:
קבוע בעל טיפוס או קבוע בעל חסר טיפוס. כאשר מגדירים קבוע אפשר להגדיר אותו מסוג שלם או לא להגדיר אותו כלל.

```
Const
1)I=100;
2)Ii:integer=100;
```

1) משתנה ללא טיפוס.
2) משתנה בעל טיפוס integer.

*הערה: קיים הבדל בין קבועים בעלי טיפוסים לבין קבועים חסרי טיפוסים. ההבדל הוא בנוגע לשימוש כפרמטרים בפונקציות ופרוצדורות (בהמשך יוסבר בהרחבה על פונקציות ופרוצדורות). כל קבוע יכול להיות מועבר לפרוצדורה או פונקציה כפרמטר ערך אך רק קבוע בעל טיפוס יכול להיות מועבר לפי התייחסות.

קבוצות

Sets- קבוצה היא אוסף של מספרים או תווים קשורים השימוש הראשי בקבוצה הוא כדי לבדוק האם תו כלשהו נמצא בתוך הקבוצה.

```
type
      Numbers:set of 0..9;
      Var num:numbers;
Begin
End.
```

זוהי קבוצה שנכללים בה כל המספרים בין 0 עד 9 לדוגמא אם אנחנו בודקים מספר אנו יכולים לדעת אם הוא בקבוצה הזאת.

```
Begin
X:=4;
Writeln('enter number');
Readln(x)
If x in numbers then
Writeln('the number is ok');
End.
```

סקלרים

ישנה אפשרות להגדיר קבוצה שכוללת מספר ערכים לדוגמא:

```
Sex:(male,female);
```

עכשיו יש לנו משתנה שיכול להכיל רק שני ערכים.

טיפוסים (Types)

השימוש בטיפוס הוא הרבה פעמים נוח לשימוש. לטיפוסים יש מספר שימושים.
1. יצית משתנה חדש – אם ברצוננו להשתמש במשתנה שערכיו יהיו ימי השבוע ניתן יהיה לעשות זאת על ידי הגדרת משתנה חדש (Days) שיכלול את ימי השבוע.
2. הערבת ערכים של משתנים מסוג String ומערכים אל פונקציות ופרוצדורות.

```
Type x = 0..9;
Var y:x;
```

יצרנו סוג משתנה חדש שיכול להכיל רק את הספרות בין 0 ל-9 ככה אנחנו חוסכים מקום הרי אם נגדיר מסוג שלם זה יכיל את כל המספרים השלמים ככה זה מכיל רק בין 0-9
את השימוש השני נסביר בפרק של פונקציות ופרוצדורות.

סדר כתיבה :

```
Program
Const
Type
Var
Begin
End.
```

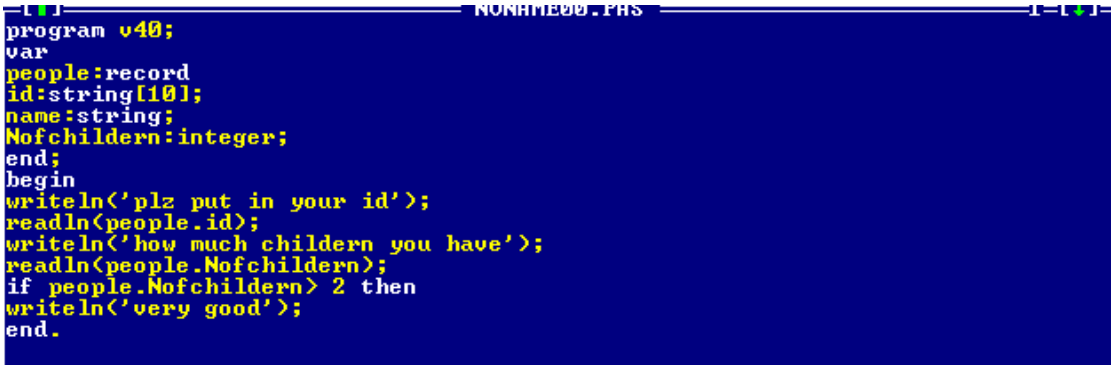
רשומות

רשומה מורכבת מצירוף של טיפוסי נתונים ,יש לה מספר יתרונות :ניתן להכניס ברשומה מספר גדול של נתונים וקל מאוד להשתמש בהם בגלל שהם מקושרים ברצף לוגי (יש להם רעיון משותף) לדוגמא רשומה של אדם (תעודות זהות)

```
People: reacord
Id:string[9];
Fname:string[10];
Lname:string[10];
Byear:string[10];
Sex:string[8];
End;
```

זוהי רשומה שבה השדות הם כל הדברים בתעודת זהות.

תוכנית V5.0



התוכנית קולטת נתונים של המשתמש לשדות של תעודת זהות ומספר ילדים ואם יש יותר מ-2 אז היא כותבת טוב מאוד .
*אפשר לעשות את זה בלי רשומה אבל זה יותר מסודר עם רשומה!!!

With

על מנת להקל על המשתמש בכתיבת שם הרשומה כל פעם אפשר לעשות כך:

```
With people
Writeln(id);
End;
```

זה שווה ל:

```
Writeln(people.id);
```

אם כותבים הרבה פעמים זה נעשה מאוד שימושי.

מערכים Arrays

מערך זה סוג נוסף של טיפוס המכיל מספר משתנים בתוכו אם אנחנו רוצים לעשות מספר משתנים בעלי אותו שם לדוגמא ציונים של 40 תלמידם עושים זאת כך:

```
Mark: array [1..40] of integer;
```

אפשר להגדיר לא רק מ-1 לדוגמא :

```
Mark: array [5..8] of string;
```

ואפשר גם תווים

```
Mark: array [a..z] of real;
```

במקרה זה יוגדרו 36 מקומות. נשים כי כי אסור לעשות מסוג עשרוני ומסוג של טקסט, למשל:

```
Mark[1.1..1.4] of real;
Mark["ad".."dd"] of Boolean;
```

כי יש אין סוף מקומות . (כמה מספרים יש בין 1.1 ל2.1????-אינסוף)

אם אנחנו רוצים לפנות לאחד מהתאים עושים את זה כך (לדוגמא תא מספר 3)

```
Write(mark[3]);
```

זה בדיוק כמו משתנה רגיל –כל תא הוא משתנה.

דוגמא למערך בעל 12 תאים:

12	11	10	9	8	7	6	5	4	3	2	1

תוכנית v3.1

```
Program v31;
Var
Student: array [1..3] of string;
Grade:array [1..3] of integer;
Max,integer,index:integer;
Begin
For i:=1 to 3
Begin
Writeln('enter your name');
Readln(student[i]);
End;
For i:=1 to 3 do
Begin
Writeln('enter your grade');
Readln(grade[i]);
End;
Max:=grade[1];
Index:=1;
For i:=2 to 3 do
If grade[i]>max then
```

```
Begin
Max:=grade[i];
Index:=I;
End;
Writeln('the student ',student[index],' with best grade
',grade[index]);
End.
```

לדוגמא: ציונים: 80,100,70 ושמות :דני,יוסי ורוני.

טבלת מעקב:

			Grade			student			Grade[i]>max
i	max	index	1	2	3	1	2	3	
1	----------	----------	דני	----	---	---	---	---	
2	----------	----------	דני	יוסי	---	---	---	---	
3	----------	----------	דני	יוסי	רוני	---	---	---	
1	----------	----------	דני	יוסי	רוני	70	---	---	
2	----------	----------	דני	יוסי	רוני	70	100	---	
3	----------	----------	דני	יוסי	רוני	70	100	80	
----------	Max=gread[1]=70	----------	דני	יוסי	רוני	70	100	80	
	70	1	דני	יוסי	רוני	70	100	80	
2	100	2	דני	יוסי	רוני	70	100	80	100>70 true
3	100	2	דני	יוסי	רוני	70	100	80	80>100 false

הפלט הסופי הוא שליוסי יש את הציון הגבוה ביותר 100.

מערך דו ממדי

מערך דו ממדי נקרא מטריצה. שזוהי בעצם טבלה המורכבת מעמודות ושורות הפועלת בשני מישורים.
כמו בצירים במתמטיקה.

דוגמא למטריצה (מערך דו-ממדי):

1	2	3
2		
3		

ובכן זהו מערך דו-ממדי של שלוש על שלוש (שלוש שורות ושלוש עמודות).

כתיבת המערך הדו-ממדי הנ"ל בפסקל תתבצע כך:

```
learn: array [1..3,1..3] of integer;
```

1..3- מספר השורות במערך.

1..3- מספר העמודות במערך.

*הערה במערך דו-ממדי קודם מצהירים על מספר השורות ורק אחר-כך על מספר העמודות
למצוא נתונים בתוך מערך דו-ממדי משום מה מבלבל הרבה מתכנתים כי לא זוכרים את הסדר של חיפוש

מציאת נתונים במערך:

עליי למצוא את הנתון במיקום הצבוע של המערך a:

1	2	3	4	5
2				
3				
4				
5				
6				
7			הנתון שצריך לקלוט	
8				
9				

ובכן מספר השורה שבו הוא נמצא זה 7 ומספר העמודה זה 4 לכן בשפת פסקל אקלוט את נתון כך:

```
X:=  a[7,4];
```

*אם לא מעניינים להשתמש במערך דו-ממדי ניתן להשתמש ברשומה חד-ממדית.

בחיים האמיתיים לא נדע איפה נמצא הנתון שעלינו לקלוט\להשתמש בו ולכן השימוש של לולאות מקוננות ומערך דו-ממדי נפוץ למדי.

כאשר אנו נשתמש בלולאות מקוננות על מנת למצוא נתונים במערך קודם כל נצטרך להצהיר על מספר השורות שאנו רוצים לחפש ואחר-כך את מספר העמודות.

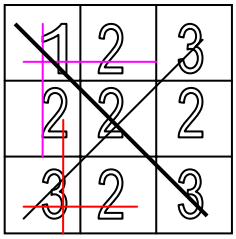
* יש אפשרות במקום לעשות מערך דו ממדי לעשות רשומה שיש במערך חד ממדי לדוגמא :

```
A:array[1..3,1..4] of integer;
:זה אותו דבר כמו
A:array[1..3] :record
B:array[1..4] of integer
End;
```

אלכסונים במערכים דו-מימדיים

הגדרת מערך לדוגמא: a:array [1..3,1..3] Of integer;

1. שימושים שונים באלכסוני המערך:



אלכסון משני:

```
For I: =1 to n do
Sum: =sum+a (I, n-I+1)
```

אלכסון ראשי:

```
for I:=1 to n
sums:=sum+a(I,I)
```

2. מתחת לראשי:

```
For I:=2 to n do          I=y
For j:=1 to I-1 do        J=x
    Sum:=sum+a(I,j)
```

3. מעל המשני:

```
for I:=1 to n-1 do
for j:=1 to n-I do
sum:=sum+a[I,j]
```

מתחת לראשי כולל

```
for I:=1 to n do
for j:=1 to n-1+1 do
sum:=sum+a[I,j]
```

מעל המשני כולל:

```
for I:=1 to n do
for j:=1 to n-1+1 do
sun:=sum+a[I,j]
```

מעל הראשי כולל:

```
for I:=1 to n-1 do
for j:=1 to n do
sum :=sum+a[I,j]
```

מתחת למשני:

```
for I:=2 to n do
for j:=n-I+2 to n do
```

sum:=sum+a[I,j]

כולל הכל:
for I:=1 to n do
for j:=n-1+1 to n do
sum:=sum+a[I,j]

String

הצהרה על משתנה מסוג string נעשית כך:

```
Var
A:string;
```

אם אנו רוצים לקלוט נתונים המורכבים מתווים עלינו להשתמש במשתנה מסוג String.
דוגמא לקלט:
אינטרנט.

ובכן קלטנו את המילה אינטרנט המורכבת מ-7 תווים. ניתן לראות שבעצם הסוג משתנה שלנו מסוג String במקרה הנ"ל הוא בעצם מערך המורכב מ-7 תווים.

ולכן ניתן להגיד בביטחון מלא שמשתנה מסוג string הוא בעצם מערך חד-מימדי.

אם כך איך ניגשים למקום ב-string. ובכן זה מאוד פשוט נגיד ויש לי משתנה מסוג string וקוראים לו a והקלט הוא מה שקלטנו לפני זה (אינטרנט) נכתוב עכשיו פקודה שתכתוב את המקום ה-5 במערך. הפקודה בפסקל תיראה כך.

```
Writeln(a[5]);
```

המשתנה String שלנו מוגבל במספר התווים שהוא יכול להכיל (כמו כל משתנה אחר) אולם כמות התווים שהוא יכול להכיל לעומת שפות תיכנות אחרות הוא ממש קטן אם אינני טועה כמות התווים שהוא יכול להכיל אינה עולה על 255 תווים.

נגיד ואני עושה תוכנית בפסקל ועלי לקלוט איזשהו משתנה שכתיבת אורך התווים המקסימלי שלו היא 15 תווים אז אין לי צורך להשתמש בכל ה-255 תווים (כי זה סתם בזבוז מקום), לפיכך ניתן להגדיר את מספר התווים שהמשתנה של מחרוזת (מחרוזת-מורכבת מספר כלשהו של תווים הגדולים מאחד) יתייחס אליהם. עושים זאת כך:

```
  A:string[7];
```

המשתנה יהיה מערך חד-מימדי בעל 7 תאים.

*הערה אם מספר התווים הנקלט כמו במקרה הנ"ל יעלה על 7 תווים אז המשתנה לא יתייחס אליהם.

דוגמא לקלט: אינטרנט זה טוב

המשתנה יכיל רק את 7 התווים הראשונים שהם: אינטרנ.
*המקום ה-0 מערך הוא בעצם הגודל שלו לכן מתחילם מאחד.

תוכנית :

```
var x:string;
i:integer;
begin
writeln('type a str');
readln(x);
for I:=1 to length(x) do
begin
if x[i]<>' ' then
x[i]:=chr(ord(x[i])+32);
end;
writeln(x);
readln;

end.
```

התוכנה קולטת סטרינג. דבר ראשון יש לולאה מ-1 עד סוף המחרוזת

סוף המחרוזת – כדי לבדוק מה הוא סוף המחרוזת יש פונקציה המקבלת את המחרוזת ומחזירה את מספר התווים שמכילה המחרוזת
(שם המחרוזת)LENGTH
כל פעם בודקים אם התו שונה מרווח אם כן אז עושים את הדבר הבאה.

מהי טבלת ה-ASCII

זוהי רשימת כל התווים שיש על המקלדת ולכל תו יש ציון מספרי יש בין 0-255, הם מסודרים לפי סדר לוגי.
ASCII ראשי תיבות של American Standard Code for Information Interchange
הוא קוד לייצוג של תווים (ספרות, אותיות האלפבית, סימני פיסוק ועוד) בזכרון מחשב ובקובצי מחשב.

CHR: זוהי פונקציה המקבלת מספר ומחזירה את התו לפי טבלת ה-ASCII.
ORD: הפונקציה מקבלת תו והופכת אותו למספר לפי טבלת ה-ASCII.

בתוכנית בעצם לוקחים את התו הופכים אותו למספר ומוספים לו 32 והופכים את התוצאה לתו –דבר זה בעצם הופך תו של אות גדולה באנגלית לאות קטנה כי ההפרש בין גדולה לקטנה הוא 32.

בתוכנית זו זה הופך רק תווים גדולים לקטים אם רוצים ההפך צריך לעשות פחות 32 .

*בגלל שזה לולאה אז זה עובר על כל התווים .

טבלת ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	∑
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	ø
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

השוואת מחרוזות

```
If str1=str2 then
….
```

אם מחרוזת 1 שווה ל 2 אז .....

פונקציות בפסקל לטיפול במחרוזות:

העתקה חלק ממחרוזת

```
Str1:=copy(str2,I,num);
```

Str1 זה המחרוזת שבה נציב חלק מהמחרוזת המקורית. Str2 זוהי המחרוזת המקורית.
I המקום שממנו נתחיל להעתיק ו-Num מספר התווים שמעתיקים.

דוגמא :

1	2	3	4	5	6	7	8	9	10	11
H	e	l	l	o		w	o	r	l	D

```
Str2:='hello world';
Write(copy(str2,7,5));
```

הפלט יהיה : World

מציאת תת מחרוזת בתוך מחרוזת :

```
I:=pos(st1,st2);
```

I בוא יוצב הפעם **הראשונה** שמופיע מחרוזת 1 ב -2

אם זה לא נמצא יוחזר 0 !!!

חיבור שתי מחרוזות

אם אנחנו רוצים להוסיף למחרוזת עוד מחרוזת לפי או אחרי

יש שתי אפשריות או כך:

```
St3:=concat(st2,st3):
```

המחרוזת השלישית תהייה שווה ל1 פלוס השניה כמו בדוגמא הבאה:

```
St:=concat('Hello',' World');
```

יש עוד אפשרות שהיא פשוטה יותר ונוחה יותר :

```
St:='hello'+' world'
```

הפיכת תו מאות קטנה לגדולה

הפונקציה מקבלת כפרמטר את התווים מ-a עד z. ומחזירה את אותו התו בצורת האותיות הגדולה.
דוגמא:

```
C:=upcase('a');
```

C (המשתנה) בעצם שווה לפרמטר A.

*כדי לעשות את זה למחרוזת צריך לעבור תו תו.

פרוצדורות בפסקל לטיפול במחרוזות

הפרוצדורה Delete:

הפרוצדורה Delete מוחקת תווים (או בשפת סטאלין 'מעלימה תווים'). על מנת למחוק תווים עלינו לציין את נקודת ההתחלה שממנה אנו רוצים למחוק, את מספר התווים שאנו רוצים למחוק וכמובן את המחרוזת שעליה אנו רוצים לעשות זאת.

דוגמא:

```
St:='hello world';
Delete(st,4,3);
```

הערך החדש של המחרוזת st יהיה:

```
St='helorld'
```

st-המחרוזת שהפרוצדורה תפעל עליה.
4- התו במחרוזת שממנו הפרוצדורה מתחילה למחוק.
3-מספר התווים שהפרוצדורה מוחקת.

הפרוצדורה Insert:

הפרוצדורה מכניסה תווים מתת-מחרוזת אחת למחרוזת אחרת. על מנת להוסיף תווים ממחרוזת אחת לאחת עלינו לציין את המחרוזת שאליה אנו מוסיפים את התווים, את המחרוזת שממנה אנו מוסיפים את התווים ואת המקום שממנו אנו מתחילים להעתיק את התווים.

דוגמא:

```
Insert(s1,s2,4);
```

הסבר:
הפרוצדורה מכניסה את מחרוזות S1 למחרוזת S2 החל מהתו ה-4.

הפרוצדורה Str:

הפרוצדורה מקבלת את אותו ערך מספרי וממירה אותו למחרוזת כלומר אם יש לי ערך מספרי שלם נסוג אינטיג'ר אז במקום שהערך יושם בתוך משתנה מספרי הוא יושם בתוך מחרוזת

```
String(value,string);
```

value- הערך השלם או הממשי שאנו רוצים להמיר.
string- המחרוזת שבה יאוחסן הערך המספרי.

הפרוצדורה Val:

הפוך מהפרוצדורה שלמדנו מקודם. הפרוצדורה val מעבירה את הערך שבמשתנה מחרוזת אל משתנה שהוא ערך מספרי כלשהו.

מבנה הפרוצדורה:

```
Var(st,var,code);
```

st- המחרוזת שמכילה את הערך המספרי.
var- המשתנה המספרי (שיכול להיות שלם או ממשי).
code- כאשר הקוד 0 אז ההמרה הצליחה אם הקוד שונה מ-0 אז הוא מכיל מספר שמצביע על מיקום התו (הראשון) שממנו ההמרה לא הצליחה.

דוגמא למחרוזת: '1ט7ו'.

בהתחלה הקוד יחזיר 1 (האות 'ו') כיוון שזהו התו הראשון שממנו ההמרה לא הצליחה. לאחר מכן אם התו ו יהפוך למספר אז הערך המספרי הבא של הקוד יהיה 3 (האות 'ט'). אם האות ט תהפוך להיות לערך מספרי אז הערך הבא של משתנה הקוד יהיה 0 כיוון שאין שום תווים השונים מתווים מספריים ולכן ההמרה הצליחה.

*הערה- אם אני משתמש בפרוצדורה כלשהי לטיפול במחרוזות בלולאה אז יכול להיות שמספר התווים בפרוצדורה ישתנה אולם עדיין הלולאה תמשיך בלי בעיות כיוון שהכוח של הפרוצדורה היא שברגע שהיא משתנה אז זה לא משפיע על הפקודות שנעשים עליה להבדיל מפונקציה. אולם עדיין זה נחשב תכנות לא תקני לכן אנו לא ממליצים לכם להשתמש בדבר הזה.

*הערה : כמו שראיתם יש פקודות שהם עצמאיות (פרוצדורות) ויש פקודות שהם לא עצמאיות (הכוונה לפקודות שנכתבות בפקודות אחרות כדוגמאת writeln).
לכן ההבדל המהותי בין פונקציה לפרוצדורה שפרוצדורה היא עצמאית ופונקציה היא לא (הבדלים נוספים ושימושים נלמד לעומק בפרק פונקציות ופרוצדורות).

תרגילים לסיכום

1) כתוב תוכנית שתהייה בה רשומה שיש בה פרטים על עובד(משכורת,שם,תעודת זהות) התוכנית תקלוט את זה ל 100 אנשים ותבדוק משכורת ממוצעת .

2) כתוב תוכנית הקולטת תו ומחרוזת ובודקת אם יש את התו במחרוזת

3) כתוב תוכנית הקולטת מחרוזת הכוללת משפט וכותב את מילה בשורה שונה

4)כתוב תוכנית הקולט מחרוזת ובודקת אם יש בין כל מילה רווח אחד או לא אם יש יותר מידי או פחות מידי היא מסדרת ומדפיסה בסוף את המחרוזת המתוקנת .

5) כתובת תוכנית הקולטת 2 מחרוזות ובודק איזה מחרוזת ארוכה יותר.

6) כתוב תוכנית הקולטת מספר עשרוני והתוכנית פולט את הערך השלם בשורה נפרדת והערך העשרוני בשורה נפרד וכותב אם הם שווים.

פתרונות

(1

```
Var
Man :record
Name:string;
Mas:integer;
Id:string;
End;
I,sum:integer;
Begin
Sum:=0;
For i:=1 to 100 do
Begin
With man do
Writeln('type name,id and mas');
Readln(name,id,mas);
End with
Sum:=sum+man.mas;
End;
Writeln(sum/100);
End.
```

פתרון ל-2:

```
Var st,letter:string;
Begin
Writeln('enter mahrozet');
Readln(st);
Writeln('enter letter');
Readln(letter);
If pos(letter,st)]<>0 then
Writeln('the letter in the mahrozet')
Else
Writeln(' the letter not in the mahrozet');
End.
```

(3

```
var i:integer;st:string;
begin
readln(st);
for i:=1 to length(st) do
begin
if st[i] <> ' ' then
write(st[i])
else
writeln(' ');
end;
end.
```

פתרון ל-4:

```
Var st,stlite:string;
Count,pbegin,num:integer;
Begin
Count:=0;
Writeln('enter mahrozet');
Readln(st);
Pbegin:=pos(' ',st);
While pos(' ',st)<>do
begin
Count:=count+1;
Num:=pos(' ',st);
Delete (st,num,1);
End;
Stlite:=Delete(st,pbegin+1,count-1);
Writeln(stlite);
End.
```

(5

```
Var st1,st2:integer
L1,l2:integer;
Begin
Readln(st1,st2);
L1:=length(st1);
L2:=length(st2);
If l1 > l2 then
begin
Writeln('st1 >st2')
Else
Writeln('st2>st1');
End.
```

(6

```
Var st:string;
Num:integer;
begin
writeln('enter mahrozet');
readln(st);
num:=pos('.',st);
writeln('the value of the number that bigger then 1 is ',
copy(st,1,num-1));
writeln('the value of the number that lower then 1 is
',copy(st,num+1,length(st)));
end.
```

פרק 6: פונקציות ופרצדורות

פונקציות

כבר ראינו פונקציות בעבר כמו הפונקציות לערך מוחלט ונוספות . אנו יכולים ליצור פונקציות משלנו לדוגמא ליצור פונקציה שמקבלת שתי מספרים ומחזירה את הסכום .
בשביל מה פונקציות ? הן עוזרות לחלק את התוכנית למשימות שונות לדוגמא תוכנית יכולה להכיל הרבה תתי משימות, אנו יכולים ליצור לכל תת משימה פונקציה ואחרי זה להשתמש בה בתוכנית הראשית שלנו.
עד עכשיו ראינו תוכניות ככה:

```
Program dogma;
Var ….
Begin
זה התוכנית הראשית
End.
```

כדי להגדיר פונקציה, צריך להגדיר אותה אחרי ההצהרה על הגדרת משתנים ולפני התוכנית הראשית .

```
Function name(x:integer):integer;
Var y:integer;
Begin
…
Name:=x+y;
End;
```

name- זה שם הפונקציה.
(x:integer)-מה שבין הסוגרים זה משתנים שהפונקציה מקבלת לדוגמא כאן זה איקס
אפשר יותר מ-אחד ובין כל משתנה פסיק .
כדי להבין יותר טוב את הפרמטרים(המשתנים בין הסוגרים) נקח לדוגמא את הפונקציה לערך מוחלט שהיא מקבלת מספר ומחזירה את הערך המוחלט ,אותו מספר זה בעצם הפרמטר של הפונקציה .
integer: - נקודתיים ואחרי זה הסוג , זה בעצם הסוג שהיא מחזירה כאן לדוגמא זה שלם
אחרי זה יש משתנים של אותה פונקציה ,במשתנים אלו אפשר להשתמש רק באותה פונקציה והם לא קיימים בתוכנית הראשית.
אחרי זה כותבים את הפונקציה ובסוף כותבים את הדבר הבא :
Name:=x+y; - שם הפונקציה נקודתיים שווה למה שהיא מחזירה לדוגמא כאן היא מחזירה סכום של שני המספרים. נראה דוגמא כדי להבין טוב יותר.

תוכנית V60

```
Program v60;
Var I,n1,n2:integer;

Function sum(n1,n2:integer):integer;
Begin
Sum:=n1+n2;
(* סוף פונקציה סכום *)
End;

Begin
(*  ראשי *)
For i:=1 to 5 do
```

```
Begin
Writeln('type two numbers');
Readln(n1,n2);
Writeln(sum(n1,n2));
End;
End.
```

הסבר : בתוכנית יצרנו פונקציה המקבלת שתי מספרים ומחזירה את הסכום. אחרי זה בתוכנית הראשית עשינו לולאה שכל פעם קולטת 2 מספרים במשך 5 פעמים והיא בעזרת הפונקציה כל פעם כותבת את הסכום שלהם .

משתנים מקומיים

בכל פונקציה או פרוצדורה אפשר לעשות משתנים שאפשר להשתמש בהם רק באותה פונקציה או פרוצדורה לשימוש ***פנימי בלבד*** .

משתנה ערך לעומת משתנה יחס

יש שני סוגים של פרמטרים (מה שמעברים לפונקציה או הפרוצדורה): פרמטר יחס ופרמטר ערך **שגרה** זה בעצם תת תוכנית, שם כולל לפונקציה ופרוצדורה .

פרמטר ערך:

כאשר משתנה מועבר כפרמטר ערך ,נוצר העתק של אותו משתנה וכאשר מתבצעים עליו פעולות שונות בשגרה הוא אינו משתנה מחוץ לאותה שיגרה . כאשר לערך משתנה זה משפיע רק על העתק הזמני

לדוגמא :

```
Function x(y:integer):integer
Begin
Y:=y+1;
X:=y;
End;
Begin
Z:=5;
Writeln(x(z));
End.
```

(y:integer)- זה פרמט ערך , כדי להגדיר פרמטר ערך לא כותבים לפני הגדרתו שום דבר בניגוד ליחס (ראה בהמשך) שכתובים לפניו את המילה var.
מעבירים בתוכנית את המשתנה זאד שערכו בעצם הוא 5 והפונקציה מגדילה את הערך הזמני של המשתנה זאד באחד לכן אם נשתמש בו אחרי זה שוב ערכו יהיה 5 ולא שש .

פרמטר יחס :

כאשר משתנה מעובר לשגרה כפרמטר יחס ,אם ערכו משתנה בתוך השגרה אז ערכו נשאר כפי שהוא השתנה ולא חוזר כמו פרמטר ערך לערכו המקורי .
דרך ההגדרה היא ככה :

```
Var y:integer;
```

ההבדל בהגדרה בין יחס לערך הוא :

```
Function dogma(x:integer;var y:integer):integer;
…
```

האיקס זהו פרמטר ערך והואי זה פרמטר יחס .!!!

תוכנית סיכום לפונקציה

תוכנית V61

```
Program v61;
Var a,b,c,d,s:integer;

Function f(var x:integer;y:integer);
Begin
X:=x+1;
Y:=y+1;
f:=x+y;
End;

Begin
A:=2;b:=4;c:=5;d:=10;s:=0;
S:=f(a,b);
Writeln(s,a,b);
S:=f(b,c);
Writeln(s,b,c);
S:=f(c,d);
Writeln(s,c,d);
S:=f(d,a);-
Writeln(s,d,a);
Writeln(a,b,c,d);
End.
```

כדי להבין מה התוכנית עושה וההבדל בין שתי סוגי הפרמטרים נעשה תוכנית מעקב :(לא נכלול את מה שבתוך הפונקציה)

a	b	c	d	s	פלט
2	4	5	10	0	
2+1=3	4	5	10	8	834
3	4+1=5	5	10	11	1155
3	5	5+1=6	10	12	12610
3	5	6	10+1=11	15	15113
3	5	6	11	15	35611

כפי שראינו המשתנה שעובר לפרמט שהוא ואר השתנה כל פעם והמשתנה השני לא השתנה כלל!!!

פרוצדורה

פרוצדורה היא בעצם קטע תוכנית. לא כמו פונקציה ,הפרוצדורה לא מחזירה בעזרת שמה היא מחזירה אך ורק אם פרמטרים יחס .

ההגדרה שלה נעשית ככה :

```
Procedure name(a:integer):
Var x:integer;
Begin
…
End;
```

name -שם הפרוצדורה
:(a:integer)- הפרמטרים
*הפרוצדורה לא מחזירה דרך שמה לכן אין נקודותיים סוג
ואחרי זה יש משתנים מקומיים ומה שהיא מבצעת.
* אין בסוף את מה שהיא מחזירה כמו בפונקציה כי היא לא מחזירה דרכה.

פונקציה לעומת פרצדורה:

יש כמה הבדלים כמו שכבר ראיתם :

1) אופן ההגדרה : פונקציה מוגדרת אם החזרה ופרוצדורה לא.
2) אופן ההחזרה : פרוצדורה מחזירה בעזרת פרמטרים שהם יחס (הם משתנים וככה זה מעובר לתוכנית הראשית) כמו לדוגמא הפרוצדורה המובנת לטיפול במחרוזות
Str(num,str); המשתנה STR הוא כאן מסוג ואר והוא משתנה בהתאם אחרי פעולת הפרוצדורה יהיה בוא מחרוזות . ולעומת זאת בפונקציה אפשר גם כך וגם בעזרת שמה (בכל מקרה חייב להחזיר משהו לפחות בעזרת שמה)
3) בתוך פרוצדורה אפשר לכתוב כל פקודה כמו WRITELN אבל בפונקציה אי אפשר לכתוב פקודות קלט ופלט כי היא מבצעת חישובים ולא פולט וקולטת דברים

העברת פרמטרים ללא הגדרה סוג

ניתן להעביר פרמטרים ללא הגדרה הסוג שלם . הם חייבים להיות מסוג יחס (var) כמו בדוגמא הבאה:

```
Procedure example (var x);
….
```

הפרמטר כאן ללא טיפוס.

העברת מחרוזות ומערכים כפרמטרים

כדי להעביר מחרוזות ומערכים חייבים להגדיר אותם מסוג טיפוס חדש אחרת תיהיה שגיאה . לדוגמא :

```
Program example;
Var s:string[10]; a:array[1..3] of integer;
```

כדי להעביר אותם חייב להגדיר אותם כטיפוס לפני הואר.

```
Program example;
Type
```

```
S=string[10];
A=array[1..10] of integer;
Var s1:s;a1:a;
```

עכשיו אפשר להעביר אותם כפרמטרים. לדוגמא יצרנו פרוצדורה :

```
T(s,a);
```

<u>תוכנית סיכום לפונקציות ופרוצדורות</u>

* יש אפשרות לזמן שגרה מתוך שגרה אחרת כמו בתוך התוכנית הראשית (חייב שהשגרה שמשתמשים תהייה קודמת לשגרה השניה) לדוגמא: זימון פרוצדורה בתוך פרוצדורה אחרת.

```
var num,num2:integer;
action:char;
procedure kelet(var num,num2:integer);
begin
writeln('enter two numbers');
readln(num,num2);
end;
function pluse(num,num2:integer):integer;
begin
pluse:=num+num2;
end;
function minose(num,num2:integer):integer;
begin
minose:=num-num2;
end;
procedure helky(num,num2:integer);
begin
if num2=0 then
writeln('error')
else
writeln(num/num2);
 nd;
 unction kefel(num,num2:integer):longint;
 egin
kefel:=num*num2;
 nd;
 rocedure act(var action:char);
 egin
writeln('choose an action');
readln(action);
hile (action<>'+') and (action<>'-') and (action<>'*') and (action<>'/') do
 egin
writeln('choose an action');
readln(action);
 nd;
if action='-' then
writeln(minose(num,num2));
if action='/' then
helky(num,num2);
if action='+' then
writeln(pluse(num,num2));
  action='*' then
writeln(kefel(num,num2));
end;

begin
kelet(num,num2);
act(action);
end.
```

תתי-משימות:

1. קליטת מספרים.
2. קליטת פעולה חשבונית
3. פליטת תוצאה או פלט בהתאם.

פרק 7: נספחים

שימוש בקבצים חיצונים

בפסקל אפשר להשתמש בקבצים חיצונים שנבנו והם חלק מהתוכנה ישנם הרבה קבצים ואפשר ללמוד הרבה עליהם בעזרה של התוכנה בספר זה אנו נסביר חלק מהם ,דבר זה יכול להעשיר מאוד את התוכנות שלכם לכן כדי מאוד ללמוד נושאים אלו .

איך משתמשים בקובץ חיצוני?

לדוגמא יש את הקובץ של מדפסת PRINTER. ככה משתמשים בו :

```
Program d;
Uses printer;
Begin
…
End.
```

שימוש במדפסת

כדי להדפיס עושים את הדבר הבא :

```
Program x;
Uses printer;
Begin
Writeln(Lst,'abs');
End.
```

כאן אנו במקום לכתוב על המסך אנו כותבים להתקן המדפסת . אפשר להדפיס כמובן גם כל דבר אחר.

ספריית CRT

ספרייה זו מכילה כל מיני דברים שאפשר לעשות במסך הטקסט כמו צביעת טקסט וצביעת הרקע.
כדי להשתמש בה עושים את הדבר הבא :

```
Uses crt;
```

Textcolor

פקודה זו משמשת כדי לצבוע את הטקסט כדי שהתוכנה תהייה יפה יותר וצבעונית.
דגש: אחרי שמשנים את הצבע הוא לא יחזור לשחור עד שלא נחזיר אותו עד סוף התוכנית. דוגמא:

```
Program x;
Uses crt;
Begin
Textcolor(1);
Writeln('hello world');
End.
```

הפלט כאן יהיה בצבע כחול.ישנם צבעים רבים ,יש גם צבעים שם מהבהבים נסו לכתוב מספרים שונים ותראו צבעים שונים.

פקודה דומה עבור צבע הרקע היא Textbackground.

GOTOXY

למסך יש כמו למפה לכל מקום יש שם מוגדר כמו בגרף. הפינה השמאלית העליונה היא 1:1 שזה בעצם x 1 ו-y 1, והפינה הימנית התחתונה היא 80:25 .

דבר זה שימושי מאוד לדוגמא אם אנחנו רוצים לכתוב משהו בעצם אנחנו צריכים להזיז את הסמן(המקום שיבוא כותבים) לאמצע המסך:

```
Program x;
Uses crt;
Begin
Gotoxy(40,12);
Write('hello');
```

התוצאה תהייה שבאמצע המסך ייכתב המילה .

Delay

כאשר נשתמש בפקודה הזאת התוכנית תעצור

```
Delay(ms);
```

Ms - כמה מילישניות נרצה שהתוכנית תחכה עד שתעבור לפקודה הבאה.

Sound

פקודה זו יוצרת קול לדוגמא התוכנית הבאה:

```
Program d;
Uses crt;
Begin
Sound(25):
Delay(100);
Nosound;
End.
```

התוכנית משמיע קול בגובה של 25 מחכה 100 מילישניות ואחרי זה מפסיק להשמיע את הצליל בעזרת הפרוצדורה NOSOUND.

READKEY

פקודה זו מחכה עד שנקלוט תו אחד בניגוד ל READLN שהיא מחכה עד שנלחץ אנטר. לדוגמא :

```
Program d;
Uses crt;
Var c:char;
Begin
 C:=readkey;
If c:='2' then
Writeln('you press 2');
End.
```

התוכנית תקלוט תו ואחרי זה אם התו 2 היא תכתוב שנלחץ 2 .

Keypressed

זוהי בעצם פקודה בולאנית שאומרת אם נלחץ באותו רגע כפתור או לא . לדוגמא הקטע תוכנית הבא:

```
While not keypressed do
Begin
Delay(100);
Write('*'):
End.
```

כל עוד לא לחצנו על כפתור ייכתב כוכבים ועד שלא נלחץ התוכנית לא תפסיק (כל פעם נחכה 100 מילישניות)

תוכנית סיכום

```
program dugma;
uses crt;
var c:char;
begin
clrscr;
textcolor(2);
gotoxy(1,25);
writeln('           ^   ');
writeln('         | ',#15,' |');
writeln('         |   | ');
writeln('        #     #');
writeln('        #  P  #');
writeln('        #  a  #');
writeln('        #  s  #');
writeln('        #  c  #');
writeln('        #  a  #');
writeln('        #  l  #');
writeln('       #       #');
writeln('      # ####### #');
writeln('    #           #');
delay(900);
writeln('         |   | ');
delay(100);
writeln('         |   | ');
delay(100);
writeln('         |   | ');
delay(100);
writeln('         |   | ');
delay(100);
writeln('');
delay(100);
writeln('');
delay(100);
writeln('');
delay(100);
writeln('');
delay(100);
writeln('');
delay(100);
```

```
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('');
writeln('');
writeln('');
delay(50);
writeln('Press any key...');
c:=readkey;
end.
```

הסבר :

בתוכנית אנחנו עושים הדמיה של טיסה של טיל בהתחלה אנחנו מוחקים את המסך אחרי זה מעבירים לנקודה ההתחלתית של המסך , מציירים את הטייל ,מחקים קצת ואחרי זה עושים עשן ...ובסוף עושים קליטת תו כדי שנוכל לראות את התמונה בסוף .

שגיאות

בפסקל יש כלים רבים למציאת שגיאות אנו נסביר מספר כלים כדי לעזור עליכם למצוא שגיאות :

מעבר שלב -שלב

יש אפשרות לעבור שורה אחר שורה ולראות איך התוכנית פועלת. אם תלחצו על כפתור F8 אז תראו שהתוכנה עוברת שלב –שלב.

הוספת נקודת צפייה

אם אתם רואים שיש נקודה מסוימת שיש בה בעיה ניתן להוסיף צפייה בעזרת הכפתורים CTRL-F7

DEBUG

כפי שאתם יכולים לראות בשורת המשימות יש את מילה זו תחתיה יש כלים רבים שעוזרים פתירת בעיות

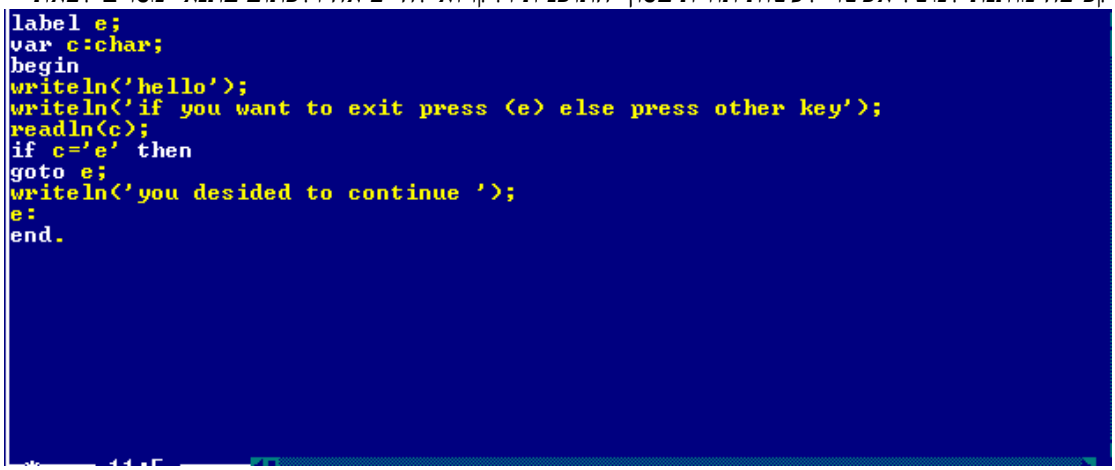
HELP

בעזרת של התוכנית יש הסברים רבים על השגיאות וככך אפשר להבין באמת במה הטעות .

כלים נוספים

GOTO

יש אפשרות להגדיר תווית שהיא מעין נקודה בתוכנית שאפשר לעבור עליה במקרה כלשהו זה בעצם קפיצה מותנת למשל אפשר לעשות תווית בסוף התוכנית ולקרוא לה יציאה ולכתוב בתנאי מסוים לצאת



Include קבצי

עורך הטורבו פסקל אינו יכול להכיל יותר מכמות טקסט מסוימת (אני לא יודע בקשר לגרסאות אחרות אבל בטורבו פסקל 6 זה 62 קילובייט).
ניתן לחלק את התוכנית לקבצי (include) כלומר קובץ ראשי וקבצים משניים נלווים.
*הערה- את הקובץ הראשי שומרים בטופס נפרד וכך גם את הקבצים הנלווים.

דוגמא לשימוש:

קובץ ראשי:

```
Program main;
(*$I try.inc*)
begin
call1;
end.
```

הערה-הסיומת של קבצים משניים היא אך ורק inc.

קובץ משני (Try.inc):

```
procedure call1;
begin
writeln('new stuff');
end;
```

הסבר ראשי:
בקובץ הראשי אחרי ה-program כתבנו את השורה הבאה:

(*$I try.inc*)

שזוהי הערה למהדר שאומרת שכוללים את הקובץ הזה בתוכנית.

אופן השימוש: יש לפתוח סוגריים עגולות עם הסימן * ולאחר מכן כותבים את סימן הדולר $ ואז I גדולה (חייב שהיא תהיה גדולה!). לאחר מכן כותבים את הקובץ שאליו פונים כולל הסיומת inc ואז כותבים את הסימן * וסוגרים את הסוגריים. ואז במקרה שלנו קראנו אל הפרוצדורה בתוך הקובץ המשני שיצרנו שנקראת call1.

הסבר קובץ משני:
כותבים אך ורק שגרה (פונקציה או פרוצדורה) ובתוכה אנו מכניסים את הפקודות.

*הערה-לא צריך לכתוב את הפקודות הראשיות Begin, end כיוון שהם נמצאות בקובץ הראשי במקרה שלנו. (למרות שיש אפשרות לקחת את הפקודות הראשיות ולשים בתוך הקובץ המשני).

פרק 8: תרגילי סיכום לספר

1) יש לפתח פונקציה אשר מקבלת שלוש אותיות ומחזירה כפלט מספר האותיות אשר שוות לאות הראשונה .
א) כתוב את זה כאלגוריתם
ב) תרגם את האלגוריתם לפסקל
ג) כתוב תוכנית הקוראת לפונקציה (לפי רצונך)

2) כתוב תוכנית הקולט מערך דו ממדי (מספרים מערך בגודל 100 על 100) ונותנת כפלט
א) סכום האלכסון המשני
ב)סכום איברים מתחת לשורה הראשונה
ג) סכום מתחת לאלכסון הראשי כולל .

3) נתונה הפרוצדורה הבא :

```
Procedure x(var x,y,z:integer;a,d:integer);
Var a,I :integer;
Begin
For i:=1 to d do
Begin
A=a+1;
X=a+y
Y=a+b;
Writeln(y);
End;
```

א) תקן את השגיאות
ב) נתונה התוכנית הבאה שמשתמשת בפרוצדורה. מה לא נכון? (תקן את השגיאות)

```
Program xxx;
…' הפרצדורה X ' …
Var x,a,b:integer;
Begin
X(x,a,20,0,b);
X(2,3,3,3,3);
Writeln(x,a,b):
```

ג) תרגם את התוכנית לאלגוריתם (המתוקנת (במקרה שמשהו לא נכון תקן ואם אפשר תמציא))
ד) מה הפלט של התוכנית? עקוב בעזרת טבלת מעקב.

4) כתוב תוכנית הקולט מחרוזות ומספר מסוג סדיר ,התוכנית תקלו מחרוזות עד שלא יקלט מספר שלילי התוכנית תיתן כפלט :
א) המחרוזת כפולה
ב) המחרוזת הפוכה
ג) כל תו לפי המקום הסידורי
ד) כל אות לפי המספר שניקלט * המקום הסידורי
ה) המחרוזת ללא מספרים
דוגמא :
המחרוזת : ,"שלום 22 זה מספר זוגי " המספר :2

מוכפלת : "שלום,22 זה מספר זוגי שלום 22 זה מספר זוגי " הפוכה : "יגוז רפסמ הז 22 סולש " כל אות לפי מספר סידורי : "שללווומממם ...

5) חברת שירים החליטה לצנזר את השירים שלה ,הם לקחו אותך כדי לבנות להם תוכנית .
תפקידך ליצור תוכנית הקולטת מערך של 10 מילים לצנזור ושירים למחרוזת עד שלא נקלטת מחרוזת ריקה . התוכנית תצנזר את השירים ותפלוט כמה פעמים הופיע כל מילה אסור מבין המערך של המילים לצנזור . הפלט כלומר השירים החדשים וכמות הפעמים שהופיע מילה אסורה

6) כתוב תת תוכנית הקולטת 2 מערכים בעלי 4 מקומות של מספרים בתחום 100-1ויוצרת מערך חדש שהוא שילוב של המספרים הגבוהים מבניהם (מחזירה את המערך החדש)

דוגמא :
מערך א

44	90	60	20

מערך ב

80	33	90	12

מערך ג(השילוב)

80	90	90	20

א) תאר במילים את התת תוכנית
ב) בחר משתנים עיקריים ,הגדר את טיפוסים ותאר את תפקידם (בעברית)
ג) חברת ייצור רובוטים החליטה לשלב 4 רובוטים קיימים (ליצור רובוט חדש שבו יש את התכונות הטובות ביותר מבין ה 4) תפקידך ליצור תוכנית הקולטת 4 מערכים של 4 מקומות בתחום המספרים בין 1 ל 100 (כל מערך כזה מתאר 4 תכנות של הרובוט לפי אחוזים בין 1 ל 100) ולקלוט מערך של שמות התכונות(שמות התוכנות בהתאם לתכונות והם שווים בין 4 הרובוטים)
1) כתוב את התוכנית בעזרת התת תוכנית שיצרת
2) פלט התכנות של הרובוט החדש (שם +אחוזים)
3) חלק את התוכנית לתת משימות

7) כתוב תוכנית הקולטת לתוך מערך דו ממדי מספרים עד שנקלט מספר דו ספרתי
תוכנית ממיינת את המספרים(לפי סדר עולה)-בעזרת פרוצדורה
התוכנית תספור כמה מספרים חד ספרתים יש ותציב אותם במערך חד ממדי-בעזרת פרוצדורה
אחרי כן יש ליצור מערך חדש שבו יש את כל המספרים בסדר הפוך של כל המעריך ביחד
הפלט הוא כל המערכים
חלק את התוכנית לתת משימות

8) כתוב תוכנית הקולטת קוד מוצר למחירה פומבית ו הצעות מחירים ובאופן אקראי (אבל הגיוני כאילו (ההצעה הגבוה ביותר
לעצור את המחירה ולהכריז על המנצח
העזר במערך*
חלק את התוכנית ל4 תתי משימות ומעלה**

9) בעזרת "A" כתוב תוכנית התקלוט מערך של תווים התוכנית תבדוק כמה רצפים יש של התו
פרוצדורה בדוק
במקרה שהוא "A" אם כן תיתן פלט בהתאם (אסור שתספור רצף של "C" "B" "A" אם יש רצף כזה "C" "B" "A" כלול ברצף של
"C" "B" "A" ורצפים של A התוכנית תיתן וכפלט כמה רציפים יש של
חלק ל פרוצדורות ופונקציה אחת ומעלה**

10) כתוב אלגוריתם למכירת כרטיסים לאתר החרמון , התוכנית תקלוט את כמות הכרטיסים שיש באתר וגיל האדם שרוצה לקנות כרטיס (התוכנית תפסיק לעבוד כאשר נגמר מלאי הכרטיסים) ותיתן כפלט :
א: המחיר שהוא צריך לשלם לפי המחירון .
ב: במקרה שהוא הזמין מראש תנתן לו הנחה אקראי (חשוב שלא תהיה מעל המחיר שהוא צריך לשלם)
ג: כמה אנשים הזמינו כרטיסים .
ד. מה הרווח היומי(במקרה של הנחה יותר מ 20 אחוז אז יש הפסד כלומר זה מופחת מהרווח)
*זכור לבדוק תקינות קלט ולכתוב תיעוד

<u>מחירות:</u>
-מתחת לגיל 5 100 שח
-מ5 עד גיל 16 120.5 שח
-מ16 עד גיל 20 (במקרה שהוא סטודנט) 30 שח
-מ16 עד 20 (במקרה שהוא לא סטודנט) 12
-מעל גיל 20 400 שח

11) כתוב פרוצדורה המקבלת מספר A ותו בודד B וקולטת תווים עד שלא נקלט התו השווה ל B או המשתמש קלט יותר פעמים מ A ומחזירה כמה ניסיונות הוא ניחש אם בכלל .כתוב תוכנית הקולט למערך חד ממדי 10 תווים ומספר הניסיונות לניחוש התווים , העזר בפרוצדורה שעשית לפני זה , כל פעם שהמשתמש ניחש בפעם אחת בניחוש תו הבא הוא יקבל בניסיון אחד יותר . התוכנית נותן כפלט:
א: את ממוצע ההצלחות כלומר כמה פעמים בממוצע לקח לו לנחש.
ב: תו+בכמה ניסיונות הוא ניחש אותו
ג: אם הוא ניחש הכל פלט כל הכבוד
ד: חלק את התוכנית לעוד תת משימות

12)כתוב תוכנית הקולטת ציונים של 50 כיתות לתוך מערך דו ממדי (בכל כיתה 50 תלמידים). התוכנית תקלוט בנוסף ציונים של 10 תלמידים מצטיינים מבית ספר אחר.
א: תוכנית תבדוק כמה תלמידים בבית ספר עברו את התלמידים המצטיינים ותתן את זה כפלט .
ב: התכנית תבדוק האם יש כיתה שעברה את ממוצע התלמידים המצטיינים (כלומר הממוצע שלה). ותתן כפלט אם כן את הכיתה והממוצע של התלמידים בה שעברו את ה90.
ג: כמה תלמידים שמספרם הסידורי בכיתה מתחלק ב 5 ללא שארית נכשלו
ד: את ממוצע הציונים של התלמיד הראשון בכיתה הראשונה השני בכיתה שניה וכך אלה
*בדוק תקינות קלט בעזרת מסננת קלט (כולל שהתלמידים המצטיינים לא קיבלו פחות מ 80)
*חלק את התוכית לתת משימות ככל האפשר

13) נתון מערך בעל 20 מקומות מסוג שלם
א: הצב בו מספרים אקראיים בין 100 ל 150 לא כולל 150
ב: בדוק כמה רצפים של 111 יש במערך ותן כפלט את האורך המקסימלי ואת מיקומה
ג: במקרה שלא היה אף רצף שנה את המספרים ובדוק שוב (בעזרת פרוצדורה)
ד: במקרה שלא היה אף פעם את המספר 111 בנה פונקציה שמחברת את כל המספרים .
** סעיף ג נמצא בתוך ד (כלומר אם אין את המספר אז גם אין רצף) אז תעשה שהם לא יתנגשו כלומר אם יש את ד אין את ג.

סוף דבר

אנו מקווים שלמדתם מהספר ובזאת אנו מקווים לפתוח אבן דרך בכתיבת מדריכים בנושא פסקל. היינו מוסיפים יותר אך גילינו מאוחר מדי על התחרות המתקיימת וזה מנע מאתנו להסביר על עוד נושאים כגון קבצים גרפיקה ועוד... מטרתנו הייתה ליצור ספר שיסביר בהיקפיות על הנושאים השונים בשפת התכנות כיוון שבהרבה ספרים ניתן לראות שפרטים שאנו שמנו במדריך מושמטים בקלות בספרים (אפילו אנו קיצצנו בדברים בגלל דוחק הזמן אז דמיינו כמה חומר עוד חסר לכם עד לאן שהגעתם בספרים אחרים). אנחנו ניסינו להעביר בקלות את החומר הנלמד למתכנתים גם בהנחה שזאת שפת התכנות הראשונה שלמדתם. יצאנו מנקודת הנחה לאחר סיום המדריך שהוא מובן לכל אחד שינסה ללמוד ממנו גם אם אתה לא מבין במחשבים. בחרנו לכתוב את המדריך הנ"ל בנושא פסקל כי לפי דעתינו המדריכים שמסופקים ברחבי האינטרנט בנושא פסקל בשפה העברית אינם מספקים אם זה בגלל עומק הדברים או בגלל חוסר האינפורמציה הרב שמחסירים בלמידה. אנחנו אולי לא הגענו לנושאים מתקדמים עקב דוחק הזמן אך אנו מבטיחים שאת החומר שלמדתם במדריך אתם תדעו באופן מקיף ויסודי. אנחנו מבטיחים לכם בזאת שאם תהייה דרישה מאנשים להמשכת המדריך אנו נעשה גם חלק ב' שיכלול נושאים מתקדמים יותר. אולי בטח שמעתם מאנשים אחרים תגובות כגון: למה ללמוד פסקל??? או זו שפה שעברה מהעולם ודברים כגון אלו. ובכן יש המון תשובות אך לפי דעתנו התשובה היא למה לא?? פסקל זאת השפה הכי קלה שדרכה תוכל לרכוש ביתר קלו והבנה את הכלים שמשתמשים בשפת התכנות כיוון שזוהי שפה שמטרתה הייתה לימוד. נכון שאולי זאת לא שפה חזקה כמו שפות אחרות: C , c++, assembly

כמו כן דרך הלימוד בפסקל (אם אתה כבר מתכנת בשפות אחרות) זאת השפה האידיאלית שדרכה תוכל לחזק את הכלים וההבנה שרכשת בשפות אחרות. כך שזה לא משנה אם אתה מתחיל או כבר מתכנת קטן אנו מליצים בחום ללמוד פסקל. גם אם זה לא ביסודיות ובהיקפיות של כלל השפה.

להערות או פניות נא לפנות לאמייל:

יגאל: virtual888@gmail.com

יוחאי: vbyohai@gmail.com