

## קורס תכנות

### שיעור שלישי: בקרת זרימה, לולאות

## המרה - Casting

- ממירים את הטיפוס של הביטוי

```
int num1 = 7, num2 = 2;
double num3 = 5.7, result = 0.0;

result = num1/num2;

result = (double)num1/num2;

num3 = (int)num3;
```

num1	num2	num3	result
7	2	5.7	3.5

2

## קלט של ערכי משתנים - scanf

- ברוב התוכניות נרצה לעבוד על קלט שמכניס המשתמש.
- הפונקציה scanf קולטת ערך מהמשתמש לתוך משתנה.

```
scanf( "%d %lf", &student_num, &average);
```

- המחשב ממתין לקלט מהמשתמש.
- אחרי הקשת הקלט על המשתמש להקיש Enter.
- המחשב מכניס את הקלט למשתנה שצוין.

- סימוני הקלט זהים לסימוני ההדפסה ב-printf.
- לפני שם המשתנה יש לשים את הסימן &.



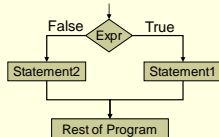
## מילים שמורות

- מילים שמורות הן מילים בשפת C כגון: if, else, int...
- אסור להשתמש בהן כשמות משתנים או פונקציות

## משפט if-else

```
if ( expression )
    statement1;
else
    statement2;
```

- אם התנאי מתקיים יתבצע משפט 1 אחרת יתבצע משפט 2



- במקרים רבים נשתמש בבולקים של משפטים בתוך if-else.

5

## מה אם יש יותר משני מצבים?

- אפשר לרשום:

```
if (condition)
    command;
else
    if (condition)
        command;
    else
        if (condition)
            command;
        else
            command;
```

- (אפשר להחליף פקודה בבולק של פקודות)

בשביל הקריאות נדאג ש-else יופיע מתחת ל-if - התואם

## בחירה בין יותר משני מצבים - דוגמה

```

if (grade >= 90) {
    printf("A\n");
} else if (grade >= 80) {
    printf("B\n");
} else if (grade >= 70) {
    printf("C\n");
} else {
    printf("Failed\n");
}
    
```

האם הציון מעל 90? תרגום לציון אמריקאי  
האם הציון מעל 80?  
האם הציון מעל 70?

## בחירה בין יותר משני מצבים - switch

- פקודת ה-switch משמשת להשוואה עם ערכים קבועים מראש
- במקרים כאלה נוחה יותר לשימוש מ-if-else מרובים

## דוגמה ל-switch: תרגום ציון אמריקאי לטווח מספרים

```

char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
    
```

## דוגמה ל-switch: תרגום ציון אמריקאי לטווח מספרים

```

char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
    
```

הביטוי שערך נבדק - הוא יכול להיות כל ביטוי מטיפוס בדיד (תו או מספר שלם).

## דוגמה ל-switch: תרגום ציון אמריקאי לטווח מספרים

```

char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
    
```

הביטוי שערך נבדק - הוא יכול להיות כל ביטוי מטיפוס בדיד (תו או מספר שלם).

הבדיקה היא אם הביטוי הנ"ל שווה לאחד מהקבועים הרשומים (לפי הסדר)

## דוגמה ל-switch: תרגום ציון אמריקאי לטווח מספרים

```

char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
    
```

הביטוי שערך נבדק - הוא יכול להיות כל ביטוי מטיפוס בדיד (תו או מספר שלם).

הבדיקה היא אם הביטוי הנ"ל שווה לאחד מהקבועים הרשומים (לפי הסדר)

אם נמצא שוויון, כל הפקודות משם ועד סוף ה-switch מבוצעות (או עד break) (אפשריים גם בלוקים של פקודות)

## דוגמה ל- switch: תרגום ציון אמריקאי לטווח מספרים

```
char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
```

הביטוי שערנו נבדק – הוא יכול להיות כל ביטוי מטיפוס בדיד (תו או מספר שלם).

הבדיקה היא אם הביטוי הנ"ל שווה לאחד מהקבועים הרשומים (לפי הסדר)

אם נמצא שוויון, כל הפקודות משם ועד סוף ה-switch מבוצעות (או עד break (אפשריים גם בלוקים של פקודות)

אם מופיעה הפקודה break קופצים לפקודה הראשונה שאחרי ה-switch

## דוגמה ל- switch: תרגום ציון אמריקאי לטווח מספרים

```
char grade;
scanf("%c", &grade);
switch (grade)
{
    case 'A':
        printf("90 - 100\n");
        break;
    case 'B':
        printf("80 - 89\n");
        break;
    case 'C':
        printf("70 - 79\n");
        break;
    default:
        printf("Failed\n");
}
```

הביטוי שערנו נבדק – הוא יכול להיות כל ביטוי מטיפוס בדיד (תו או מספר שלם).

הבדיקה היא אם הביטוי הנ"ל שווה לאחד מהקבועים הרשומים (לפי הסדר)

אם נמצא שוויון, כל הפקודות משם ועד סוף ה-switch מבוצעות (או עד break (אפשריים גם בלוקים של פקודות)

אם מופיעה הפקודה break קופצים לפקודה הראשונה שאחרי ה-switch

default מבוצע אם לא נמצאה אף התאמה (אופציונלי)

## מבנה כללי של switch

```
switch (ביטוי בדיד)
{
    case קבוע1:
        פקודה או פקודות
        break; (אופציונלי)
    case קבוע2:
        פקודה או פקודות
        break; (אופציונלי)
    ....
    default: (אופציונלי)
        פקודה או פקודות
}
```

ביטוי בדיד – תוצאתו היא מס' שלם או תו

אם לא שמים break בין מספר cases אז אותן פקודות יבוצעו עבור מקרים שונים

default לא חייבת להיות המקרה האחרון

אסור שיפיעו שני cases בעלי אותו הערך

## מבנה כללי של switch

לא נשים break כשנרצה לעשות אותו דבר בכמה מקרים. לדוגמא:

```
switch (tav)
{
    case '0':
    case '2':
    case '4':
    case '6':
    case '8':
        printf("It's an even digit\n");
        break;
    case '1':
    case '3':
    case '5':
    case '7':
    case '9':
        printf("It's an odd digit\n");
        break;
    default:
        printf("It's not a digit\n");
}
```

השורה תבוצע אם התו הוא ספירה זוגית

השורה תבוצע אם התו הוא ספירה אי-זוגית

בכל מקרה אחר

## לולאות



## לולאות - דוגמא

```
#include <stdio.h>
int main()
{
    int i=1;
    while ( i <= 10 )
    {
        printf("%d\n", i);
        i=i+1;
    }
    return 0;
}
```

כל עוד i קטן או שווה ל-10

נדפיס את הערך של i

נעדכן את הערך של i

מה לדעתכם עושה התוכנית הבאה?

## לולאות

- מהן לולאות?
- סוגי לולאות:
  - while
  - for
  - do-while
- כיצד שוברים לולאה?
  - break
  - continue

## מהן לולאות

- לולאה:
  - קטע קוד שיכול להיות מבוצע מספר פעמים ברצף
  - קטע הקוד יבוצע כל עוד תנאי מסוים מתקיים
- שימושים בלולאות:
  - בקרת זרימה
  - מאפשרות ביצוע פעולה מסוימת מספר פעמים ברצף
- מכנה משותף של לולאות:
  - איתחול (initialization)
  - תנאי (condition)
  - שינוי המצב הנוכחי (increment / update)

## לולאת while

### דרך פעולה:

- נאתחל את המשתנים הקשורים לתנאי
- כל עוד התנאי מתקיים:
  - נבצע קבוצת פקודות
  - נבצע פקודת עדכון שתשפיע על חישוב התנאי

```
initialize stuff...
while (condition)
{
    do something ...
    increment ...
}
```

## הדפסת המספרים מ-1 ועד 10

```
#include <stdio.h>

int main()
{
    int i=1;
    while ( i <= 10 )
    {
        printf("%d\n", i);
        i=i+1;
    }
    return 0;
}
```

אתחול

תנאי

קידום

## לולאות – בדיקת ראשוניות

### מספר ראשוני:

- מספר טבעי גדול מ-1 המתחלק ללא שארית בעצמו וב-1 בלבד

```
#include <stdio.h>

int main()
{
    int num, i=2;
    scanf("%d", &num);
    while ( num % i != 0 )
        i=i+1;
    if (i != num)
        printf("%d is divided by %d\n", num, i);
    else
        printf("%d is prime\n", num);
    return 0;
}
```

האם קיים מספר בין 1 ל-חמט המחלק את חמט ללא שארית?

## לולאות אינסופיות

### אם לא נגדיר כראוי:

- את אתחול המשתנים המשפיעים על התנאי
- את התנאי עצמו כך שיהיה תלוי בערכו של משתנה אחד לפחות
- פקודה אחת (לפחות) המתבצעת כחלק מהלולאה ומשנה את ערכו של המשתנה המשפיע על התנאי

```
while ( i > 0 )
{
    printf("%d\n", i);
    i = i + 1;
}
```

```
while ( 5 > 4 )
{
    printf("%d\n", i);
    i = i + 1;
}
```

```
i = 1;
while ( i > 0 )
{
    printf("%d\n", i);
}
```

לולאה אינסופית

## לולאות שלא מבוצעות

```
#include <stdio.h>

int main()
{
    int num, i=2;
    scanf("%d", &num);

    while ( num % i != 0 )
        i=i+1;

    if ( i != num)
        printf("%d is divided by %d\n", num , i);
    else
        printf("%d is prime", num);

    return 0;
}
```

אם num זוגי הלולאה לא תבצע

## מה עושה התוכנית הבאה?

```
#include <stdio.h>

int main()
{
    int i=1, result=1;

    while ( i <= 10 )
    {
        result = result * 2;
        i++;
    }

    printf("The result is %d\n", result);
    return 0;
}
```

חישוב הערך של  $2^{10}$

## לולאת while

- מבנה הלולאה
- דוגמאות
- לולאה אינסופית
- **שאלות?**

## כתיבה מקוצרת של פעולות

a += 2;	↔	a = a + 2;
b -= 3;	↔	b = b - 3;
c *= 4;	↔	c = c * 4;
d /= 5;	↔	d = d / 5;
e %= 5;	↔	e = e % 5;

## כתיבה מקוצרת של פעולות

f++; ↔ f = f + 1;

- קודם יוחזר ערך המשתנה ואח"כ נקדם את ערכו ב-1:

i = 5; j שווה ל-5.  
j = i++; i שווה ל-6.

++f; ↔ f = f + 1;

- קודם נקדם את ערך המשתנה ב-1 ואח"כ נחזיר את ערכו:

i = 5; j שווה ל-6.  
j = ++i; i שווה ל-6.

## כתיבה מקוצרת של פעולות

g--; ↔ g = g - 1;

- קודם יוחזר ערך המשתנה ואח"כ נוריד את ערכו ב-1:

i = 5; j שווה ל-5.  
j = i--; i שווה ל-4.

--g; ↔ g = g - 1;

- קודם נוריד את ערך המשתנה ב-1 ואח"כ נחזיר את ערכו:

i = 5; j שווה ל-4.  
j = --i; i שווה ל-4.

## לולאות for

```
for (פקודת עדכון; תנאי ביצוע; פקודת אתחול)
{
    פקודות
}

בדוגמא:
for ( i = 1; i<=10 ; i++)
    printf("%d\n", i);
```

- קודם מבוצעת פקודת האתחול פעם אחת.
- אז נבדק תנאי הביצוע – אם הוא מתקיים אז נכנסים ללולאה.
- בסיום פקודות הלולאה, מבוצעת פקודת העדכון
- אז שוב נבדק תנאי הביצוע ונכנסים ללולאה אם הוא מתקיים
- וחוזר חלילה, עד שהתנאי לא מתקיים.

## דוגמא:

```
#include <stdio.h>
int main()
{
    int i;
    for (i = 1; i<=10; i++)
        printf("%d\n", i);
    return 0;
}
```

וחוזר חלילה...

## מה עושה התכנית הבאה?

```
#include <stdio.h>
int main()
{
    int i,result=1;
    for ( i=1; i <= 20; i++ )
        result = result * 2;
    printf("The result is: %d\n", result);
    return 0;
}
```

התוכנית מחשבת את הערך של  $2^{20}$

## מה עושה התכנית הבאה?

```
#include <stdio.h>
int main()
{
    int i,result=1;
    for ( i=1; i <= 10; i++ )
        result = result * i;
    printf("The result is: %d\n", result);
    return 0;
}
```

התוכנית מחשבת את הערך של  $10!$

## while לעומת for

```
#include <stdio.h>
int main()
{
    int i,result=1;
    for ( i=1; i <= 10; ++i )
        result *= i;
    printf("Result is %d", result);
    return 0;
}
```

```
#include <stdio.h>
int main()
{
    int i,result=1;
    i=1;
    while ( i <= 10 )
    {
        result *= i;
        ++i;
    }
    printf("Result is %d", result);
    return 0;
}
```

כל מה שניתן לעשות בלולאת for ניתן לעשות בלולאת while ולהפך.

## while לעומת for

- לולאות for:
  - כשרוצים לבצע משהו בצורה סדרתית
  - מספר הפעמים (איטרציות) ידוע מראש
  - עבור הערכים 1 ועד n
- לולאות while:
  - כשרוצים לבצע משהו מספר לא ידוע של פעמים
  - כל עוד i קטן מ-10
  - כל עוד x זוגי

## לולאות - אתחול ועדכון

```
for ( i=0, j=4; i+j < 100; i+=5,--j )
{
    do something ...
}
```

אתחול:

- מותר לבצע מספר פקודות אתחול

עדכון:

- מותר לבצע מספר פקודות עדכון
- פקודת עדכון יכולה להיות כל פקודה המשנה את ערך משתנה התנאי

## לולאות מקוננות

• לולאות יכולות להיות חלק מלולאות אחרות:

```
for ( i=0; i < 10; ++i )
{
    for ( j=i; j < 10; ++j )
        printf("%*");

    printf("\n");
}
```

```
*****
*****
*****
*****
*****
****
***
**
*
```

## מה עושה התכנית הבאה?

```
#include <stdio.h>
```

```
int main()
{
    int i=0,j=0;

    for ( i=1; i <= 10; i++ )
    {
        for ( j=1; j <= 10; j++ )
            printf("%d ", i*j);

        printf("\n");
    }

    return 0;
}
```

התוכנית מדפיסה את לוח הכפל

```
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```

## סידור הפלט

```
#include <stdio.h>
```

```
int main()
{
    int i=0,j=0;

    for ( i=1; i <= 10; i++ )
    {
        for ( j=1; j <= 10; j++ )
            printf("%4d", i*j);

        printf("\n");
    }

    return 0;
}
```

התוכנית מדפיסה את לוח הכפל

```
cx "E:\worktable\C\trt\Debug\main.exe"
1  2  3  4  5  6  7  8  9 10
2  4  6  8 10 12 14 16 18 20
3  6  9 12 15 18 21 24 27 30
4  8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
Press any key to continue.
```

## לולאות do-while

```
initialize ...
do
{
    do something ...
    increment ...
}
while ( condition );
```

ההבדל בין do-while ל-while:

- גוף הלולאה מבוצע לפחות פעם אחת (אפילו אם התנאי לא מתקיים לעולם)
- התנאי נבדק רק לאחר ביצוע גוף הלולאה

## לולאות do-while

```
#include <stdio.h>
```

```
int main()
{
    int i=1;

    do
    {
        printf("%d\n", i);
        i++;
    }
    while ( i < 10 );

    return 0;
}
```

## לולאות do-while

```
#include <stdio.h>

int main()
{
    int i=1;

    do
    {
        printf("%d\n", i);
        i++;
    }
    while (i < 10);

    return 0;
}
```

```
#include <stdio.h>

int main()
{
    int i=1;

    while (i < 10)
    {
        printf("%d\n", i);
        i++;
    }

    return 0;
}
```

## לולאות do-while

```
#include <stdio.h>

int main()
{
    int i=11;

    do
    {
        printf("%d\n", i);
        i++;
    }
    while (i < 10);

    return 0;
}
```

תנאי הלולאה לא מתקיים

פלט: "11"

```
#include <stdio.h>

int main()
{
    int i=11;

    while (i < 10)
    {
        printf("%d\n", i);
        i++;
    }

    return 0;
}
```

אין פלט

## לולאת do-while

```
#include <stdio.h>

int main()
{
    int grade = 0;

    do
    {
        printf("Enter a grade (0-100)\n");
        scanf("%d", &grade);
    }
    while ((grade < 0) || (grade > 100));

    printf("You entered a legal grade now\n");
    return 0;
}
```

שימוש נפוץ:

- קבלת קלט מהמשתמש שוב ושוב עד שיוכנס קלט חוקי

## הפקודה break בלולאות

הפקודה break:

- מפסיקה את ביצוע הלולאה הנוכחית
- התוכנית תמשיך לרוץ החל מהפקודה הראשונה שאחרי הלולאה
- משמשת למקרים בהם רוצים לסיים את הלולאה באמצע
- לפני סיום האיטרציה הנוכחית

## דוגמא – חישוב ממוצע של עד 10 ציונים

- דרישות מהתכנית:
  - לקלוט עד 10 ציונים מהמשתמש (או עד שמוכנס מספר שלילי)
  - להדפיס את הממוצע

## דוגמא – חישוב ממוצע של עד 10 ציונים

```
#include <stdio.h>

int main()
{
    int newNum = 0, num_elements = 0;
    double avg = 0.0;

    do
    {
        scanf("%d", &newNum);
        if (newNum < 0)
            break;

        avg += newNum;
        ++num_elements;
    }
    while (num_elements < 10);

    avg /= num_elements;
    printf("The averages is %lf\n", avg);
    return 0;
}
```



## הפקודה continue

**continue:**

- מפסיקה את האיטרציה הנוכחית של הלולאה
- התוכנית תמשיך לרוץ מתחילת האיטרציה הבאה
- במקרה של for נמשיך מפעולת הקידום.

## דוגמא – חישוב ממוצע של 10 ציונים

נתעלם ממספרים שליליים

```
#include <stdio.h>

int main()
{
    int newNum = 0, num_elements = 0;
    double avg = 0.0;

    do
    {
        scanf("%d", &newNum);
        if (newNum < 0)
            continue;
        avg += newNum;
        ++num_elements;
    }
    while ( num_elements < 10 );

    avg /= num_elements;
    printf("The averages is %lf\n", avg);
    return 0;
}
```

שאלות?