«Προγραμματισμός Ι»

2- Εντολές Ελέγχου της C

Δρ. Χαράλαμπος Καρανίκας
ΠΣ Μηχανικών Πληροφορικής

ΟΙ ΕΝΤΟΛΕΣ ΕΛΕΓΧΟΥ ΤΗΣ *C*

- Έλεγχος της ροής εκτέλεσης των προγραμμάτων (**if** & **for**)
- Εντολή επιλογής. **if** (εκτέλεση κώδικα υπό συνθήκη)

  Π.χ. if (*έκφραση*) εντολή;
- Η *έκφραση* (συνθήκη ελέγχου) ~ οποιαδήποτε έγκυρη έκφραση της C
  - Έκφραση αληθής (true) ~ μη μηδενική τιμή
  - Έκφραση ψευδής (false) ~ μηδενική τιμή
- Σχεσιακοί (<,>,==) & Λογικοί τελεστές - Τμήματα κώδικα

```
#include <stdio.h>

int main(void)
{
    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);

    if(num < 0) printf("Number is negative.");
    if(num > -1) printf("Number is non-negative.");

    return 0;
}
```

ΠΑΡΑΔΕΙΓΜΑΤΑ ΤΗΣ if

```
#include <stdio.h>

int main(void)
{
        int answer;

        printf("What is 10 + 14? ");
        scanf("%d", &answer);

        if(answer == 10+14) printf("Right!");

        return 0;
}
```

```
/*Πόδια σε μέτρα ή μέτρα σε πόδια*/
#include <stdio.h>

int main(void)
{
        float num;
        int choice;

        printf("Enter value: ");
        scanf("%f", &num);

        printf("1: Feet to Meters, 2: Meters to Feet. ");
        printf("Enter choice: ");
        scanf("%d", &choice);

        if(choice == 1) printf("%f", num / 3.28);
        if(choice == 2) printf("%f", num * 3.28);

        return 0;
}
```

ΠΡΟΣΘΗΚΗ ΤΗΣ else

- Η εντολή if … else (αμφίδρομη λήψη αποφάσεων, είτε η μία είτε η άλλη):
  if (έκφρασή) εντολή1;
  else εντολή2;
- Η else προτιμότερη από δεύτερο if. (Αποδοτικότερος κώδικας)

```
/*Αρνητικός-Θετικός αριθμός*/
#include <stdio.h>

int main(void)
{
        int num;

        printf("Enter an integer: ");
        scanf("%d", &num);

        if(num < 0) printf("Number is negative.");
        else printf("Number is non-negative.");

        return 0;
}
```

```
/*Διαιρεί δύο αριθμούς*/
#include <stdio.h>

int main(void)
{
        int num1, num2;

        printf("Enter first number: ");
        scanf("%d", &num1);

        printf("Enter second number: ");
        scanf("%d", &num2);

        if(num2 == 0) printf("Cannot divide by zero.");
        else printf("Answer is: %d.", num1 / num2);

        return 0;
}
```

ΔΗΜΙΟΥΡΓΙΑ ΤΜΗΜΑΤΩΝ ΚΩΔΙΚΑ

- Τμήμα κώδικα: Δύο ή περισσότερες εντολές μαζί σε { }
- Μπορεί να χρησιμοποιηθεί όπου και μία εντολή

```
/*Βελτιωμένη έκδοση του προγρ.Πόδια σε μέτρα ή μέτρα σε πόδια*/
#include <stdio.h>
int main(void)
{
        float num;
        int choice;
        printf("1: Feet to Meters, 2: Meters to Feet. ");
        printf("Enter choice: ");
        scanf("%d", &choice);

        if(choice == 1) {
            printf("Enter number of feet: ");
            scanf("%f", &num);
            printf("Meters: %f", num / 3.28);
        }
        else {
            printf("Enter number of meters: ");
            scanf("%f", &num);
            printf("Feet: %f", num * 3.28);
        }
        return 0;
}
```

```
#include <stdio.h>

int main(void)
{
        int answer;

        printf("What is 10 + 14? ");
        scanf("%d", &answer);

        if(answer == 10+14) printf("Right!");
        else {
            printf("Sorry, you're wrong. ");
            printf("The answer is 24.");
        }
        return 0;
}
```

Ο τελεστής συνθήκης (?):

Γενική μορφή: **έκφραση ? εντολή-1 : εντολή-2**

Αν η (λογική) **έκφραση** είναι αληθής τότε αναπτύσσεται η **εντολή-1**, διαφορετικά αναπτύσσεται η **εντολή-2**.

Ουσιαστικά πρόκειται για μία συμπαγής μορφή της εντολής **if-else**:

```
if (έκφραση)
    εντολή-1
else
    εντολή-2
```

Για παράδειγμα **num = (x<=5) ? 1 : 0;**

Η μεταβλητή **num** θα πάρει την τιμή **1** αν το **x** είναι μικρότερο ή ίσο του **5**, διαφορετικά θα πάρει την τιμή **0.**

ΧΡΗΣΗ ΤΟΥ ΒΡΟΧΟΥ for

- Η for μία από τις 3 εντολές δημιουργίας βρόχου της C.
- Επιτρέπει την επαναλαμβανόμενη εκτέλεση εντολών.
- Γενική μορφή:

for (αρχικοποίηση; συνθήκη-ελέγχου; αύξηση) εντολή;

  - Μεταβλητή ελέγχου. Η μεταβλητή που ελέγχει τον βρόχο
  - Ο βρόχος for μπορεί να τρέχει με «αρνητικό» βήμα.

    For (i=10; i>0; i=i-1)…
  - Επιπλέον μπορεί να αυξάνεται ή να μειώνεται με βήμα μεγαλύτερο από το 1.

    For (i=0; i<50; i=i+5)…

```
#include <stdio.h>

int main(void)
{
    int i;

        for(i=1; i<11; i=i+1)
          printf("*");

        return 0;
}
```

ΠΑΡΑΔΕΙΓΜΑΤΑ ΤΗΣ for-1

```
#include <stdio.h>

int main(void)
{
        int num;

        for(num=1; num<11; num=num+1)
                printf("%d ", num);
        printf("terminating");

        return 0;
}
```

```
#include <stdio.h>

int main(void)
{
        int num;

        /* this loop will not execute */
        for(num=11; num<11; num=num+1)
        printf("%d ", num);

        printf("terminating");

        return 0;
}
```

ΠΑΡΑΔΕΙΓΜΑΤΑ ΤΗΣ for-2

```c
#include <stdio.h>

int main(void)
{
    int answer, count;

    for(count=1; count<11; count=count+1) {
        printf("What is %d + %d? ", count, count);
        scanf("%d", &answer);

        if(answer == count+count) printf("Right! ");
        else {
          printf("Sorry, you're wrong. ");
          printf("The answer is %d. ", count+count);
        }
   }

   return 0;
}
```

```c
/*Αριθμός πρώτος ή όχι. Αριθμός πρώτος εάν διαιρείται μόνο με τον ευατό του και το 1*/
#include <stdio.h>

int main(void)
{
    int num, i, is_prime;

    printf("Enter the number to test: ");
    scanf("%d", &num);

    is_prime = 1;
    for(i=2; i<=num/2; i=i+1)
        if((num%i)==0) is_prime = 0;

    if(is_prime==1) printf("The number is prime.");
    else printf("The number is not prime.");

    return 0;
}
```

ΔΗΜΙΟΥΡΓΙΑ ΄ΕΝΘΕΤΩΝ ΒΡΟΧΩΝ

Όταν ο κορμός εντολών ενός βρόχου περιέχει έναν άλλο βρόχο(ένθετος).

/*Δέκα φορές στην οθόνη οι αριθμοί 1 έως 10. */

```
for(i=0; i<10; i++) {
        for(j=1; j<11; j++) printf("%d ", j);
       printf("\n");
}
```

```
#include <stdio.h>

int main(void)
{
  int i, j, k;
  for(i=0; i<3; i++)
    for(j=0; j<26; j++)
      for(k=0; k<2; k++) printf("%c", 'A'+j);

  return 0;
}
```

Εκτύπωση ολόκληρου του αλφάβητου τρείς φορές. Σε κάθε εκτύπωση κάθε γράμμα εκτυπώνεται δύο φορές.

ΥΠΟΚΑΤΑΣΤΑΣΗ ΤΩΝ ΤΕΛΕΣΤΩΝ ΑΥΞΗΣΗΣ & ΜΕΙΩΣΗΣ ΤΗΣ C

- for(num=0; num<some_value; num=num+1)
- Τελεστής αύξησης: i = i + 1; ~ i++;
- Τελεστής μείωσης: count = count - 1; ~ count--;
- Οι τελεστές αυτοί είναι γρηγορότεροι από τις ισοδύναμες εντολές εκχώρησης
- Όταν ο τελεστής (++ ή --) ακολουθεί το όνομα της μεταβλητής η πράξη του εκτελείται αφού χρησιμοποιηθεί η τιμή της μεταβλητής στην έκφραση.

```c
/* This will print 11 10 */
#include <stdio.h>

int main(void)
{
    int i, j;

    i = 10;
    j = i++; /* Else j = ++i; prints 11, 11*/

    printf("i and j: %d %d", i, j);

    return 0;
}
```

```c
#include <stdio.h>

int main(void)
{
    int i;

    i = 0;

    i++;
    printf("%d ", i); /* prints 1 */
    i--;
    printf("%d ", i); /* prints 0 */

    return 0;
}
```

ΕΠΕΚΤΑΣΗ ΤΩΝ ΔΥΝΑΤΟΤΗΤΩΝ ΤΗΣ printf

- Η C υποστηρίζει μία σειρά από ειδικούς (escape) χαρακτήρες:

Κωδικός	Σημασία
\b	Backspace
\f	Form feed (αλλαγή σελίδας)
\n	Newline (αλλαγή γραμμής)
\r	Carriage return (επαναφορά δρομέα)
\t	Horizontal tab (οριζόντιος στηλοθέτης)
\”	Double quote (εισαγωγικό)
\’	Single quote (απόστροφος)
\0	Null (Κενό)
\\	Backslash
\v	Vertical tab (κατακόρυφος στηλοθέτης)
\a	Alert (προειδοποιητικό σήμα)
\?	Question mark (ερωτηματικό)
\N	Τιμή εκφρασμένη στο οκταδικό σύστημα
\xN	Τιμή εκφρασμένη στο δεκαεξαδικό σύστημα

ΠΑΡΑΔΕΙΓΜΑΤΑ ΤΗΣ printf

```c
#include <stdio.h>

int main(void)
{
    printf("This is line one.\n");
    printf("This is line two.\n");
    printf("This is line three.");

    return 0;
}
```

```c
#include <stdio.h>

int main(void)
{
    printf("\a");

    return 0;
}
```

```c
#include <stdio.h>

int main(void)
{
    printf("one\ntwo\nthree\nfour");

    return 0;
}
```

ΧΡΗΣΗ ΤΩΝ ΣΧΕΣΙΑΚΩΝ & ΛΟΓΙΚΩΝ ΤΕΛΕΣΤΩΝ ΤΗΣ C-1

- Σχεσιακοί τελεστές της C, συγκρίνουν δύο τιμές και επιστρέφουν (true/false):

Τελεστής	Ενέργεια
>	Μεγαλύτερο από
>=	Μεγαλύτερο από ή ίσο με
<	Μικρότερο από
<=	Μικρότερο από ή ίσο με
==	Ίσο
!=	Άνισο (διάφορο)

- Λογικοί τελεστές της C, συνενώνουν αποτελέσματα (true/false):

Τελεστής	Ενέργεια
&&	AND
||	OR
!	NOT

Example:

p=0, q=1→ p&&q=0, p||q=1, !p=1

ΧΡΗΣΗ ΤΩΝ ΣΧΕΣΙΑΚΩΝ & ΛΟΓΙΚΩΝ ΤΕΛΕΣΤΩΝ ΤΗΣ C-2

- Οι σχεσιακοί και οι λογικοί τελεστές έχουν χαμηλότερη προτεραιότητα από τους αριθμητικούς τελεστές.

  10 + count>a +12 →(10+count)>(a+12)

- Προτεραιότητα σχεσιακών & λογικών τελεστών:

  Υψηλότερη !

  > >= < <=

  ++ !=

  &&

  Χαμηλότερη ||

Examples:

- if (a>0 && b>0) printf(“both are positive”);
- If (count !=0)… → if (count)…
- If (count ==0)… →if (!count)…

ΠΑΡΑΔΕΙΓΜΑΤΑ ΛΟΓΙΚΩΝ & ΣΧΕΣΙΑΚΩΝ ΤΕΛΕΣΤΩΝ

```c
#include <stdio.h>

int main(void)
{
  int i, j;

  printf("Enter first number: ");
  scanf("%d", &i);
  printf("Enter second number: ");
  scanf("%d", &j);

  printf("i < j %d\n", i < j);
  printf("i <= j %d\n", i <= j);
  printf("i == j %d\n", i == j);
  printf("i > j %d\n", i > j);
  printf("i >= j %d\n", i >= j);

  printf("i && j %d\n", i && j);
  printf("i || j %d\n", i || j);
  printf("!i !j %d %d\n", !i, !j);

  return 0;
}
```

```c
#include <stdio.h>

int xor(int a, int b);

int main(void)
{
  int p, q;

  printf("enter P (0 or 1): ");
  scanf("%d", &p);
  printf("enter Q (0 or 1): ");
  scanf("%d", &q);
  printf("P AND Q: %d\n", p && q);
  printf("P OR Q: %d\n", p || q);
  printf("P XOR Q: %d\n", xor(p, q));

  return 0;
}

int xor(int a, int b)
{
  return (a || b) && !(a && b);
}
```