Προγραμματισμός σε C

Σημειώσεις

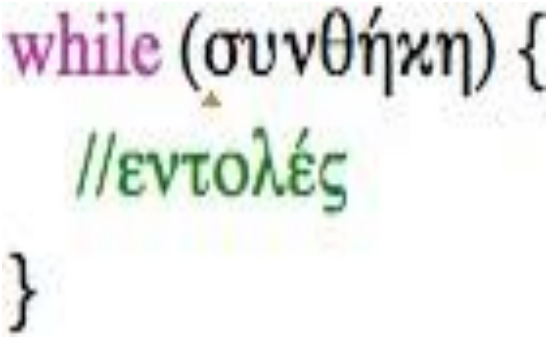
04 Επαναληπτικές Δομές

Περίληψη

Notes

1 Η επαναληπτική δομή while

Όταν λέμε επαναληπτικές δομές εννοούμε μία δομή που επαναλαμβάνει την εκτέλεσή της για όσο ισχύει (δίνει true) η αποτίμηση μιας λογικής συνθήκης. Η πρώτη από τις 3 επαναληπτικές δομές που υποστηρίζει η γλώσσα C είναι η while και η γενική της σύνταξη είναι η παρακάτω:



Η εκτέλεση των εντολών του βρόχου επαναλαμβάνεται για όσο η αποτίμηση της συνθήκης δίνει true (δηλαδή τιμή διαφορετική του 0). για παράδειγμα ο παρακάτω βρόχος

```
int i = 0;
  while (i
    < 10)
    {  i = i
    + 1;
  }
```

θα εκτελεστεί 10 φορές και μετά ο έλεγχος του προγράμματος θα περάσεις μετά την while. Όταν τερματιστεί η εκτέλεση του βρόχου η τιμή του i θα είναι 10.

Σημαντικό: οι εντολές μια while μπορεί **να μην εκτελεστούν και ποτέ** εάν την 1η φορά που θα αποτιμηθεί η συνθήκη ελέγχου δώσει false! Επίσης, πρέπει να προσέχουμε να μεταβάλλεται η αποτίμηση της συνθήκης σωστά έτσι ώστε κάποια στιγμή η αποτίμησή της να δώσει τιμή false και το πρόγραμμα να προχωρήσει παρακάτω. Αν αυτό δεν γίνει, τότε θα έχουμε έναν βρόχο που θα εκτελείται συνέχεια (**ατέρμων βρόχος**).

2 Η επαναληπτική δομή do - while

Εναλλακτικά της δομής while μπορούμε να χρησιμοποιήσουμε την δομή do-while η οποία έχει την εξής σύνταξη:

Οι εντολές της εκτελούνται για όσο η αποτίμηση της συνθήκης δίνει true. Η βασική διαφορά της με την while είναι ότι οι εντολές της εκτελούνται τουλάχιστον 1 φορά πριν αποτιμηθεί για 1η φορά η συνθήκη. Προσοχή στο ελληνικό ερωτιματικό που υπάρχει στο τέλος της και το οποίο στην ουσία τερματίζει την εντολή!

Σημαντικό: και εδώ πρέπει να προσέχουμε να μεταβάλλεται η αποτίμηση της συνθήκης σωστά έτσι ώστε κάποια στιγμή η αποτίμησή της να δώσει τιμή false και το πρόγραμμα να προχωρήσει παρακάτω. Αν αυτό δεν γίνει, τότε θα έχουμε έναν βρόχο που θα εκτελείται συνέχεια (**ατέρμων βρόχος**).

3 Βασικές χρήσεις μιας do - while

Όταν θέλουμε να εκτελεστούν επαναληπτικά κάποιες εντολές για όσο ισχύει κάποια συνθήκη, μπορούμε να χρησιμοποιήσουμε όποια επαναληπτική δομή επιθυμούμε αρκεί -φυσικά- να το κάνουμε σωστά. Παρόλα αυτά, η do-while συνίσταται για τις ακόλουθες περιπτώσεις:

I. **έλεγχος τιμής από το πληκτρολόγιο**
   Φανταστείτε το σενάριο να ζητάμε από τον χρήστη να μας δώσει κάποια τιμή από το πληκτρολόγιο και να πρέπει αυτή η τιμή να βρίσκεται μέσα σε κάποιο διάστημα τιμών [a, b]. Τότε η καταλληλότερη δομή επανάληψης για να ελέγχουμε και να ζητάμε εκ νέου την τιμή από τον χρήστη είναι η do-while όπως φαίνεται και στο παρακάτω παράδειγμα:

```
int timi;

do {
   printf("Δώσε μία τιμή στο διάστημα [3-10]:");
   scanf("%d", &timi);
} while (timi < 3 || timi >10);
```

II. διαχείριση Μενού Επιλογών

Φανταστείτε το σενάριο να θέλουμε να εμφανίζουμε ένα Μενού Επιλογών στον χρήστη και να θέλουμε το πρόγραμμά μας να επιστρέφει σε αυτό κάθε φορά που τελειώνει κάποια εργασία με βάση την επιλογή από το μενού που έκανε ο χρήστης. Τότε και πάλι χρησιμοποιούμε την δομή do-while σε συνδυασμό με μία switch:

```
do {
    printf("1. Κατάθεση\n");
    printf("2. Ανάληψη\n");
    printf("3. Έλεγχος υπολοίπου\n");
    printf("0. Έξοδος\n\n");
    printf("Δώσε την επιλογή σου:");
    scanf("%d", &epilogi);

    switch (epilogi) {
        case 1:
            //εντολες
            break;
        case 2:
            //εντολες
            break;
        case 3:
            //εντολες
            break;
        case 0:
            //εντολες
            break;
        default:
            break;
    }

} while (epilogi != 0);
```

4 Η επαναληπτική δομή for

Η επαναληπτική δομή for έχει ως σύνταξη την ακόλουθη:

```
for (αρχικήΤιμή; συνθήκη; ανανέωση) {
  //εντολές
}
```

Εκτελείται ως εξής:

- αρχικά και για 1 και μόνο φορά γίνεται η αρχικοποίηση της μεταβλητής που θα χρησιμοποιεί η for πχ i = 0;
- στη συνέχεια αποτιμάται η συνθήκη ελέγχου
- εάν η αποτίμηση δώσει αποτέλεσμα true τότε εκτελούνται οι εντολές της for
- στο τέλος εκτέλεσης των εντολών εκτελείται η ανανέωση της τιμής της μεταβλητής της for
- και στη συνέχεια αποτιμάται εν νέου η συνθήκη ελέγχου. Εάν δώσει και πάλι true εκτελούνται για ακόμη μία φορά οι εντολές της for. Η διαδικασία συνεχίζεται για όσο η αποτίμηση δίνει true.

Σημαντικό: και οι εντολές της for **μπορεί να μην εκτελεστούν καμία φορά**. Επίσης, **τα 3 μέρη της χωρίζονται πάντα με 2 ελληνικά ερωτηματικά ;** ακόμη και όταν οι ενδιάμεσες εκφράσεις παραλείπονται για κάποιον λόγο. Για παράδειγμα, μία for θα μπορούσε να γραφτεί ως εξής:

```
i = 0;

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

εφόσον το i έχει αρχικοποιηθεί πιο πριν.

Φυσικά, στο σημείο της ανανέωσης μπορούμε να έχουμε αύξηση ή και μείωση με οποιονδήποτε ρυθμό, για παράδειγμα:

```
for ( i = 0; i < 10; i += 5) {
   printf("%d\n", i);
}
```

```
for ( i = 0; i < 10; i *= 2) {
   printf("%d\n", i);
}
```

```
for ( i = 1000; i > 10; i /= 5) {
   printf("%d\n", i);
}
```

```
for ( i = 1000; i > 10; i -= 3) {
   printf("%d\n", i);
}
```

5 Βασικές χρήσεις της for

Η επαναληπτική δομή for χρησιμοποιείται κυρίως στις ακόλουθες 2 περιπτώσεις:

I. **όταν γνωρίζουμε τον αριθμό των επαναλήψεων που πρέπει να γίνουν**

II. **όταν θέλουμε να έχουμε πρόσβαση σε στοιχεία ενός πίνακα.**

6 Break και continue

Την **εντολή break** την έχουμε δει στην switch και η χρήση της είναι να διακόπτει την εκτέλεση της συγκεκριμένης εντολής και ο έλεγχος του προγράμματος να πηγαίνει στην επόμενη εντολή μετά την switch. Όταν η break χρησιμοποιείται σε μία επαναληπτική εντολή και εκτελεστεί, τότε διακόπτεται η εκτέλεση της επαναληπτικής δομής και ο έλεγχος πηγαίνει στην αμέσως επόμενη εντολή μετά από αυτή. Για παράδειγμα :

```
for ( i = 0; i < 10; i ++) {

    if (i == 5)
        break;

    printf("%d\n", i);
}
```

η εντολή printf() θα εκτελεστεί για i 0, 1, 2, 3, 4 και **όταν γίνει 5 η συνθήκη της if δίνει true και εκτελείται η break με συνέπεια να τερματιστεί η for** και το πρόγραμμα να πάει παρακάτω. Η έξοδος θα ήταν η ακόλουθη:

```
0
1
2
3
4
Program ended with exit code: 0
```

Η **εντολή continue** στην ίδια περίπτωση θα προκαλούσε “αγνόηση” των υπόλοιπων εντολών (της printf() στην ουσία) της συγκεκριμένης επανάληψης της for (για i=5 δηλαδή) και ο έλεγχος θα εκτελούσε την επόμενη επανάληψη (για i=6) χωρίς να τερματίσει την εντολή for!. Εάν στην θέση της break βάλουμε την εντολή continue στον παραπάνω κώδικα, θα είχαμε τα εξής αποτελέσματα:

```
for ( i = 0; i < 10; i ++) {

    if (i == 5)
        continue;

    printf("%d\n", i);
}
```

Έξοδος ???

```
for ( i = 0; i < 10; i ++) {

   if (i == 5)
      continue;

   printf("%d\n", i);
}
```

Έξοδος

```
0
1
2
3
4
6
7
8
9
Program ended with exit code: 0
```

δηλαδή θα εκτυπωνόνταν όλα τα i εκτός από το 5.

Άσκηση 01 - άθροισμα τετραγώνων

Να γραφεί πρόγραμμα το οποίο να ζητάει από τον χρήστη έναν θετικό ακέραιο αριθμό και θα υπολογίζεικ θα εκτυπώνει το άθροισμα των τετραγώνων των Ν πρώτων ακεραίων

$S = 1^2 + 2^2 + 3^2 + \ldots + N^2$

Άσκηση 01 - άθροισμα τετραγώνων

Να γραφεί πρόγραμμα το οποίο να ζητάει από τον χρήστη έναν θετικό ακέραιο αριθμό και θα υπολογίζεικ θα εκτυπώνει το άθροισμα των τετραγώνων των Ν πρώτων ακεραίων

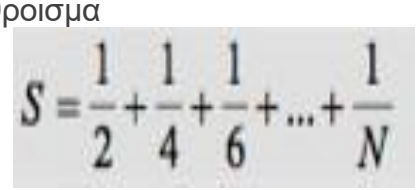
$S = 1^2 + 2^2 + 3^2 + \ldots + N^2$

Λύση

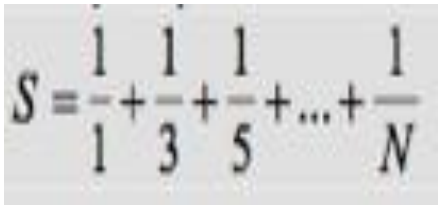
```
#include <stdio.h>
#include <math.h>

int main( ) {


    int i, S;
    int N;

    do {
       printf("N:");
       scanf("%d", &N);
    } while (N < 1);

    S = 0;

    for (i = 1; i <= N; i++) {
       S += pow(i, 2.0);
    }

    printf("S = %d\n\n", S);

    return 0;
}
```

Άσκηση 02 - Άθροισμα κλασμάτων

Να γραφεί πρόγραμμα στο οποίο ο χρήστης να δίνει έναν θετικό ακέραιο αριθμό Ν και το πρόγραμμα να υπολογίζει το άθροισμα



εάν το Ν είναι άρτιος ή το άθροισμα



εάν το Ν είναι περιττός.

Άσκηση 04 - Μενού τράπεζας

Ένας πελάτης τράπεζας έχει καταθέσεις που ανέρχονται στο ποσό των 100.000 ευρώ. Να γράψετε πρόγραμμα το οποίο:

α) θα εμφανίζει στον χρήστη ένα **Μενού Επιλογών με επιλογές 1.Ανάληψη, 2.Κατάθεση, 0.Έξοδος**

β) ο χρήστης θα δίνει την επιλογή του

γ) ανάλογα με την επιλογή ο χρήστης θα δίνει το ποσό της συναλλαγής του

δ) μετά τη συναλλαγή το πρόγραμμα θα εμφανίζει το αποτέλεσμα της και το τελικό υπόλοιπο του λογαριασμού

ε) το πρόγραμμα **πάντα θα επιστρέφει στο Μενού Επιλογών** μέχρι ο χρήστης να δώσει 0 και να τερματίσει το πρόγραμμα