Τα βήματα εκτέλεσης της do-while

1. Εκτελείται η ομάδα εντολών που υπάρχει ανάμεσα στα άγκιστρα { }

2. Γίνεται έλεγχος της τιμής της συνθήκης (η οποία είναι συνήθως μία σχεσιακή έκφραση)
   - Αν η συνθήκη είναι ψευδής (false) τότε ο do-while βρόχος τερματίζεται και η εκτέλεση του προγράμματος συνεχίζει με την πρώτη εντολή που υπάρχει μετά το άγκιστρο κλεισίματος της do-while εντολής
   - Αν η συνθήκη είναι αληθής (true) τότε επανεκτελείται η ομάδα εντολών που υπάρχει ανάμεσα στα άγκιστρα { }
   - Το βήμα αυτό επαναλαμβάνεται μέχρι η τιμή της συνθήκης να γίνει ψευδής

Παρατηρήσεις

- Ο βρόχος **do-while** χρησιμοποιείται πολύ λιγότερο από τους **for** και **while** βρόχους

- Όποιο πρόβλημα λύνεται με χρήση του βρόχου **do-while** θα μπορούσε να επιλυθεί και με χρήση βρόχων **while** ή **for**

Κάθε **do-while** βρόχος εκτελείται τουλάχιστον μία φορά, ακόμα κι αν η **συνθήκη** του βρόχου είναι ψευδής

Ο βρόχος **do-while** **πρέπει να τελειώνει** με το ελληνικό ερωτηματικό (;)

Παράδειγμα

- Ποια είναι η έξοδος του παρακάτω προγράμματος ???

```
#include <stdio.h>
int main()
{
        int i = 5;
        do
        {
                printf("text\n");
                i += 5;
        } while(i > 10);

        return 0;
}
```

Έξοδος: text

Η εντολή goto

- Η εντολή goto χρησιμοποιείται με σκοπό να μεταφέρει την εκτέλεση του προγράμματος σε κάποια άλλη εντολή μέσα στην ίδια συνάρτηση, με την προϋπόθεση ότι η εντολή έχει μία ετικέτα

- Γενική σύνταξη της εντολής goto:

```
goto location;
```

- Όταν εκτελείται η εντολή goto η εκτέλεση του προγράμματος μεταβαίνει άμεσα στην εντολή που ακολουθεί τη θέση που έχει δηλωθεί με το όνομα location

- Η θέση με το όνομα location πρέπει να είναι **μοναδική** μέσα στη συνάρτηση και προσδιορίζεται γράφοντας **το όνομά της** και **άνω κάτω τελεία** :

Παρατηρήσεις (ΙΙ)

- Ίσως, ορισμένες φορές, η χρήση της goto να οδηγεί και σε απλούστερο κώδικα (π.χ. στην άμεση έξοδο από έναν «αρκετά ένθετο» for βρόχο, όπως τον παρακάτω)

```
for(i = 0; i < 10; i++)
        for(j = 0; j < 20; j++)
                for(k = 0; k < 30; k++)
                {
                        if(συνθήκη)
                                goto NEXT;
                }
NEXT:
...
```

Παιχνίδι «βρες τον αριθμό!»

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main() {
    int i, random_number, users_guess, attempts=3;
    srand(time(NULL));
    random_number = rand() % 10 + 1;
    printf("You have %d attempts!\n",attempts);
    for(i=1; i<=attempts; i++) {
        printf("Enter your guess (1-10): ");
        scanf("%d",&users_guess);
        if ((users_guess<1) || (users_guess>10) ){
            i--;
            continue;
        }
        if (users_guess == random_number) {
            printf("BINGO!!!");
            break;
        }
        else {
            if (users_guess < random_number) {
                printf("Try a larger number. You still have %d attempts\n",attempts-i);
            }
            else {
                printf("Try a smaller number. You still have %d attempts\n",attempts-i);
            }
        }
    }
    return 0;
}
```

Άσκηση 6.34
(επέκταση και για δεκαδικούς αριθμούς)

```c
#include <stdio.h>
#include <math.h>
int main() {
    float fnum;
    int num, sum, dig;
    sum = dig = 0;
    printf("Enter number: ");
    scanf("%f", &fnum);
    if(fnum < 0)
        fnum = -fnum; /* Απλά, να τον κάνουμε θετικό. */
    else if(fnum == 0)
        dig = 1; /* Ελέγχουμε την περίπτωση που ο χρήστης εισήγαγε το 0. */

    while((fnum - floor(fnum)) > 0) {
        fnum *= 10;
    }
    num = fnum;
    while(num > 0)  {
        sum += num % 10;
        num = num / 10;
        dig++;
    }
    printf("%d digits and their sum is %d\n", dig, sum);
    return 0;
}
```

Άσκηση 6.37
(βρόχος while με χρήση break)

```c
#include <stdio.h>
int main() {
    int i, num, sum;

    printf("Enter number: ");
    scanf("%d", &num);
    if(num <= 0) {
        printf("Error: The number should be positive\n");
        return 0;
    }
    sum = 0;
    i = 1;
    while(1) {
        sum += i*i;
        if(sum >= num)
            break;
        i++;
    }
    printf("The last number is = %d\n", i-1); /* Η τιμή i-1 αποτελεί την
    return 0;
}
```

Άσκηση 6.37
(βρόχος for χωρίς χρήση break)

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

int main() {
    int i, n, M, sum;

    printf("Enter number: ");
    scanf("%d", &M);
    if(M <= 0) {
        printf("Error: The number should be positive\n");
        return 0;
    }
    sum = 0;
    for (i=1; (sum+=pow(i,2))<=M; i++);
    n = --i;
    printf("The last number is = %d\n", n);
    return 0;
}
```