Κεφάλαιο 3.5-3.6, 3.2:

Συναρτήσεις ΙΙ

(Διάλεξη 12)

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Ανασκόπηση Δομής Προγράμματος με Συναρτήσεις

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

1

```
void PrintMessage ();
```

Πρότυπο (Δήλωση) Συνάρτησης

(Δηλώνουν τι επιπλέον συναρτήσεις θα χρησιμοποιήσουμε χωρίς να ορίζουμε πως)

2

```
main  ( )
{
    PrintMessage ( );
    (Ζητάμε να εκτελεστεί η συνάρτηση)
}
```

Κλήση Συνάρτησης

3

```
void PrintMessage (void)
{    (Ορίζουμε ποιες εντολές θέλουμε να εκτελεστούν με την κλήση)
    printf ("A message for you:\n\n");
    printf ("Have a Nice Day!\n");
}
```

Ορισμός Συνάρτησης

Συναρτήσεις

Σήμερα θα δούμε σε περισσότερο βάθος τις συναρτήσεις.

A) Συναρτήσεις χωρίς Παραμέτρους
B) Συναρτήσεις με Παραμέτρους
- Χωρίς Επιστροφή τιμής
- Με Επιστροφή Τιμής

C) Εμβέλεια Μεταβλητών
D) Συναρτήσεις Βιβλιοθηκών

Α) Συνάρτηση Χρήστη χωρίς παραμέτρους

```
#include <stdio.h>

void display_six_stars(void)
{
        printf("******\n");
        return;
}

int main(void)
{
        display_six_stars();
        return 0;
}
```

Προσοχή

Είτε "void" με "return"

Ή "int" με "return 0"

Β) Συναρτήσεις με Μία Παράμετρο χωρίς τιμή εξόδου

Πρόβλημα:

Γράψετε μια συνάρτηση η οποία παίρνει ως τιμή εισόδου την ακτίνα ενός κύκλου (R), και στην συνέχεια εκτυπώνει το εμβαδόν του κύκλου με τον τύπο εμβαδό=3.14*R^2

Λύση:

```
void display_area(float R)
{
        int result;
        result = 3.14 * R * R;
        printf("The area is %f square meters\n", result);
}
```

Β) Συναρτήσεις με Μία Παράμετρο χωρίς τιμή εξόδου

```
void printNumber(int number)
{
        printf("%d mod 3 is %d\n",number, number%3);
}
```

Τι πρόβλημα υπάρχει με την πιο κάτω συνάρτηση?

```
void printNumber(int number)
{
        int result;
        result = 5/number;
        printf("5/%d = %d\n", number, result );
}
```

Β) Συναρτήσεις με τιμή εξόδου

Συνάρτηση παίρνει δυο παραμέτρου

```
float compute_area(float a, float b)
{
        float area;
        area = a * b;
        return area;
}
```

Επιστρέφει την τιμή της area που είναι τυπου float

```
// Μπορούσε απλούστερα να γράφει σαν:
float compute_area(float a, float b) {
        return a * b;
}
```

Β) Σημασία Επιστροφής τιμής εξόδου

Τι γίνεται ακριβώς όταν μια συνάρτηση επιστρέψει κάποια τιμή?

1) Η τιμή ανατίθεται σε κάποια μεταβλητή

2) Το πρόγραμμα συνεχίζει να εκτελείτε από εκεί και πέρα

3) Αποδεσμεύετε η μνήμη που δεσμεύθηκε όταν έγινε η κλήση της συνάρτησης. Οι μεταβλητές (της συνάρτησης) δεν υφίστανται πλέον

Ας δούμε όλη την εικόνα της εκτέλεσης …

τυπικοί παράμετροι

πρωτότυπο συνάρτησης

ορισμός συνάρτησης

πραγματικοί παράμετροι

κλήση συνάρτησης

```
#include <stdio.h>
float compute_area(float x, float y);

float compute_area(float a, float b)
{
    2       return (a * b);
}

int main()
{
        float length, width;
        float area;

        printf("Enter length and width: ");
        scanf("%f%f",&length, &width);
    1   area = compute_area(length, width);

    3   printf("The area of a rectangle with dim %f by %f m is %f sq. m\n",
                    length, width, area);
        return(0);
}
```

Β) Παράδειγμα Υπολογισμού Μέσου δυο ακεραίων

```
#include <stdio.h>
float AverageTwo (int num1, int num2);
main ( )
{
    float average;
    int num1 = 5, num2 = 8;
    average = AverageTwo (num1, num2);
    printf (“The average of %d and %d is %f\n”, num1, num2, average);
}

float AverageTwo (int num1, int num2)
{
    float average;

    average = (num1 + num2) / 2.0;
    return average;
}
```

C) Εμβέλεια Μεταβλητής (scope)

- Το τμήμα του προγράμματος που μπορεί μια μεταβλητή να χρησιμοποιηθεί
  - **local (τοπικές):**
    - δηλώνονται στην αρχή ενός **programming block** {..}
    - Χρησιμοποιούνται οπουδήποτε μετά τον ορισμό μέσα στο block
  - **global (σφαιρικές):** ΑΠΑΓΟΡΕΥΟΝΤΑΙ στο μάθημα
    - Δηλώνονται έξω από συναρτήσεις
    - Χρησιμοποιούνται οπουδήποτε μέσα στο πρόγραμμα

C) Είδη Μεταβλητών και Εμβέλεια

```
#include <stdio.h>

#define d 5; // ΚΑΘΟΛΙΚΗ ΣΤΑΘΕΡΑ
int c=5;    // ΚΑΘΟΛΙΚΗ ΜΕΤΑΒΛΗΤΗ (όλες οι συναρτήσεις την βλέπουν)

int sum(int x, int y) // x,y είναι τυπικές παράμετροι
{
     int c = 1;          // ΤΟΠΙΚΗ ΜΕΤΑΒΛΗΤΗ (μόνο η sum την βλέπει)
     return x + y + c;
}

int main()
{
   int a = 5, b=6, c=11; // ΤΟΠΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ (μόνο η main τις βλέπει)

   printf("%d + %d = %d", a, b, sum(a,b));

   return 0;
}
```

Το πρόγραμμα τυπώνει: 5+6=12

C) Τοπικές Μεταβλητές: Αλλαγή τιμής μεταβλητής

ΛΑΘΟΣ τρόπος να αυξήσουμε το num κατά 1

```
#include <stdio.h>
void AddOne (int);

main ()
{
    int num = 5;
    AddOne (num);
    printf (“In main: “);
    printf (“num = %d\n”, num);
}
```

num

5

```
void AddOne (int num)
{
    num++;
    printf (“In AddOne: “);
    printf (“num = %d\n”, num);
}
```

num

6

Το πρόγραμμα εκτυπώνει:

In AddOne: num=6

In main: num= 5

C) Τοπικές Μεταβλητές: Αλλαγή τιμής μεταβλητής

ΣΩΣΤΟΣ τρόπος να αυξήσουμε το num κατά 1

```
#include <stdio.h>
int AddOne (int);

main ()
{
    int num = 5;

    num = AddOne (num);
    printf ("In main: ");
    printf ("num = %d\n", num);
}
```

num

5

```
int AddOne (int num)
{
        num++;
        printf ("In AddOne: ");
        printf ("num = %d\n", num);
        return num;
}
```

num

6

Το πρόγραμμα εκτυπώνει:

In AddOne: num=6

In main: num= 6

D) Βιβλιοθήκες - Header Files

- Πολλές από τις λειτουργίες ενός προγραμματιστή είναι ήδη ορισμένες σε βιβλιοθήκες.
- Έτσι δεν χρειάζεται να γράφουμε όλη την λειτουργία ενός προγράμματος από την αρχη κάθε φορά.
- Τα header files, περιέχουν πρότυπα συναρτήσεων και δηλώσεις σταθερών και τύπων δεδομένων για τη συγκεκριμένη βιβλιοθήκη

D) Κοινά Header Files

header file	**Περιέχει πρότυπα συναρτήσεων για:**
<stdio.h>	είσοδο/ έξοδο
<math.h>	μαθηματικά
<stdlib.h>	μετατροπές αριθμών σε κείμενο (και αντίστροφα, ανάθεση μνήμης (memory allocation), τυχαίους αριθμούς, και αλλα.
<time.h>	χειρισμός ώρας και ημερομηνίας
<ctype.h>	συναρτήσεις που ελέγχουν χαρακτήρες για κάποιες ιδιότητες και μπορούν να μετατρέψουν μικρά σε κεφαλαία
΄<string.h>	χειρισμός strings

D1) Βιβλιοθήκη math.h

- **double sqrt (double x);**
  - Επιστρέφει την τετραγωνική ρίζα του x
- **double pow (double x, double y)**
  - Το x υψώνεται στην δύναμη y
    - pow (3.0, 2.0) is 9.0
    - pow (8.0, 1.0 / 3) is 2.0
- **double sin (double x)**
  - Ημίτονο του x (το x είναι σε ακτίνια)
- **Όλες οι μαθηματικές συναρτήσεις έχουν double ως ορίσματα, και επιστρέφουν double**

D2) Βιβλιοθήκη stdlib.h

- void exit (int x);
  - Προκαλεί τερματισμό του προγράμματος
- void srand (unsigned int x);
  - Αρχικοποιεί την γεννήτρια τυχαίων αριθμών με ένα ακέραιο (unsigned), π.χ.
    - srand (200);
- int rand (void);
  - Επιστρέφει ένα ψευδοτυχαίο ακέραιο (unsigned int) 0 και 4294967295
  - num = rand( );

D2) Βιβλιοθήκη stdlib.h: Συνάρτηση rand()

- Καθώς η rand( ) επιστρέφει unsigned integers σε ένα συγκεκριμένο εύρος τιμών, πρέπει να επεξεργαστούμε την τιμή που επιστρέφεται για να ταιριάξει στις ανάγκες μας.
- Αν θέλουμε τυχαίους αριθμούς από 0 μέχρι το 5
  - num = rand ( ) % 6
- Αν θέλουμε από το 1 μέχρι το 6
  - num = 1 + (rand( ) % 6);
- Από το 5 μέχρι το 20
  - num = 5 + (rand ( ) % 16);

D2) Βιβλιοθήκη stdlib.h: Συνάρτηση rand()&srand()

- Η γεννήτρια ψευδοτυχαίων αριθμών, χρειάζεται ένα unsigned int ως είσοδο (seed)
- Παρόλο που αυτά που παράγει φαίνονται τυχαίοι αριθμοί, αν ξαναχρησιμοποιήσουμε το ίδιο seed, θα πάρουμε την ίδια ακολουθία τυχαίων αριθμών
- Για να πάρουμε διαφορετική ακολουθία τυχαίων αριθμών, πρέπει να δίνουμε διαφορετικό seed

D2) Βιβλιοθήκη stdlib.h: Συνάρτηση rand() & srand()

```
#include <stdio.h>
#include <stdlib.h>
#define SEED 67

main ( )
{
    int i, num;
    srand (SEED);
    for (i = 0; i < 5; i++)
    {
        num = rand ( );
        num = num % 6;
        printf ("%d\n", num);
    }
}
```

Χρησιμοποιώ το 67 ως seed (σπόρο) συνεπώς οι ίδιοι αριθμοί θα παράγονται κάθε φορά που τρέχω το πρόγραμμα

Αρχικοποίηση

Το πρόγραμμα μας παράγει 5 τυχαίους αριθμούς από το μηδέν έως το πέντε

D3) Βιβλιοθήκη time.h

- Μια χρήσιμη συνάρτηση στη βιβλιοθήκη time είναι η time( )
- Επιστέφει την ώρα της ημέρας ως δευτερόλεπτα
- Εφόσον ο αριθμός είναι διαφορετικός κάθε φορά που καλείται, μπορεί να χρησιμοποιηθεί ως seed σε μία γεννήτρια τυχαίων αριθμών
- Π.χ.
  - **srand (time (NULL))** ;