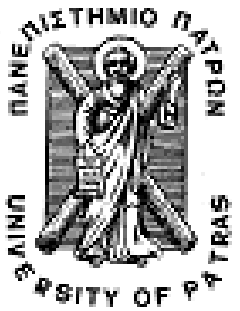
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ



ΕΙΣΑΓΩΓΗ ΣΤΟ ΔΙΑΔΙΚΑΣΤΙΚΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2010-2011)

ΥΠΕΥΘΥΝΟΙ ΔΙΔΑΣΚΟΝΤΕΣ ΕΡΓΑΣΤΗΡΙΟΥ: Α. ΦΩΚΑ, Κ. ΣΤΑΜΟΣ

2ο ΣΕΤ ΑΣΚΗΣΕΩΝ

ΕΝΔΕΙΚΤΙΚΕΣ ΛΥΣΕΙΣ

Οι ασκήσεις αυτού του φυλλαδίου καλύπτουν τα παρακάτω θέματα και δίνονται ενδεικτικά οι αντίστοιχες ενότητες στο βιβλίο The GNU C Programming Tutorial που μπορείτε συμβουλευτείτε (http://crasseux.com/books/ctutorial/) .

- Εντολές ελέγχου (κεφάλαιο Decisions)
- Εντολές επανάληψης (κεφάλαιο Loops)

Άσκηση 1η

Γράψτε ένα πρόγραμμα που να υπολογίζει τον αριθμό των ψηφίων ενός ακέραιου αριθμού (τύπου long int) και το άθροισμά τους.
Π.χ. Το 1087 έχει 4 ψηφία και έχουν άθροισμα 16.

```
#include <stdio.h>

main()
{
        long int x, sum, count;

        printf("Dwse enan arithmo: ");
        scanf("%ld", &x);

        sum = count = 0;

        while(x > 0) {
                count++;
                sum += x % 10;
                x = x / 10;
                printf("x = %d, count = %d, sum = %d\n", x, count, sum);
        }

        printf("O arithmos exei %d pshfia kai to athroisma tous einai %d", count, sum);
}
```

Άσκηση 2η

Δεδομένου (από τον χρήστη) ενός αριθμού N>3, να υπολογιστούν οι παρακάτω παραστάσεις:

P1 = 1 + 2 + 3 + ... + N (άθροισμα όλων των αριθμών από 1 έως N)
P2 = 1 + 3 + 5 + ... + N (άθροισμα όλων των περιττών αριθμών από 1 έως N)
P3 = 1/1 * 1/2 * 1/3 * ... * 1/N
P4 = $1^0 + 2^1 + 3^2 + \ldots + N^{N-1}$

Σημείωση: Θα πρέπει να κάνετε έλεγχο για λαθεμένη είσοδο από τον χρήστη και σε αυτή την περίπτωση να ξαναζητάτε το N. Χρησιμοποιείστε την εντολή do ... while. Για τον υπολογισμό της ύψωσης ενός αριθμού σε δύναμη ΔΕΝ πρέπει να χρησιμοποιήσετε την συνάρτηση power() αλλά να υπολογιστεί με μια εντολή επανάληψης.

```
#include <stdio.h>
main()
{
 int i,j,N;
 int P1, P2, P4, term;
 float P3;

 do
 {
    printf("Dwse arithmo N>3: ");
    scanf("%d", &N);
 }while (N<=3);

 P1 = P2 = P4 = 0;
 P3 = 0.0;

 for (i=1; i<=N; i++)
 {
     P1 += i;

     if (i%2 != 0)
        P2 += i;

     P3 += 1.0 / i;

     term = 1;

     for (j=0; j<i-1; j++) /*ypswsh se dynamh*/
         term *= i;

     P4 += term;
 }
  printf("P1 = %d\nP2 = %d\nP3 = %.2f\nP4 = %d\n", P1, P2, P3, P4);
}
```

Άσκηση 3η

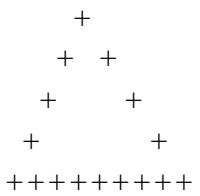
Γράψτε ένα πρόγραμμα που υπολογίζει όλους τους πρώτους αριθμούς που είναι μικρότεροι από μια τιμή που ορίζεται από μια σταθερά LIMIT. Προσπαθήστε να λύσετε το πρόβλημα με όσο το δυνατόν λιγότερους υπολογισμούς.

Σημείωση: Πρώτος αριθμός λέγεται ο αριθμός που διαιρείται ακριβώς μόνο με τον εαυτό του και το 1.

```
#include <stdio.h>
#define LIMIT 1000
main()
{
        int cnt = 0, j, k;
        for (k = 2; k < LIMIT; ++k)
        {
                j = 2;
                while ( k % j != 0)
                        ++j;
                if ( j == k)
                {
                        ++cnt; /* a prime has been found */
                        if ( cnt % 6 == 1 ) /* change line every 6 primes */
                                printf( "\n");
                        printf( "%12d", k);
                }
        }
        printf("\n\nthere are %d prime numbers less than %d \n\n", cnt, LIMIT);
}
```

Άσκηση 4η

Γράψτε ένα πρόγραμμα το οποίο να ζητά από τον χρήστη να εισάγει ένα περιττό αριθμό N, N > 3, (να γίνεται έλεγχος εγκυρότητας της τιμής του N) και στην συνέχεια να σχεδιάζει στο τερματικό, με χρήση μόνο των χαρακτήρων +, <space> και <newline> ένα τρίγωνο της μορφής που φαίνεται στο παρακάτω σχήμα (που αντιστοιχεί στην περίπτωση N=5).



```
#include <stdio.h>

main()
{
 int i,j,N,nspaces1,nspaces2;

 do
 {
    printf("Dwse arithmo N>3, peritto: ");
    scanf("%d", &N);
 }while (N<=3 || N%2==0);

 nspaces1 = N-1;
 nspaces2 = 1;

 /*1h grammh */
 for (j=0; j<nspaces1; j++)
         printf(" ");

 printf("+\n");

  /*grammes 2h ews N-1 */
 for (i=1; i<N-1; i++)
 {
     nspaces1--;

     for (j=0; j<nspaces1; j++)
         printf(" ");

     printf("+");

     for (j=0; j<nspaces2; j++)
         printf(" ");

     printf("+\n");

     nspaces2 += 2;
  }

  /*grammh N*/
  for (i=0; i < (N-1)*2 + 1; i++)
      printf("+");

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

Άσκηση 5η

Λόγω της αύξησης της τιμής της βενζίνης, ένας οδηγός χρειάζεται ένα πρόγραμμα το οποίο θα υπολογίζει τη μέση κατανάλωση βενζίνης του αυτοκινήτου του, δηλ. πόσα χιλιόμετρα έκανε ανά λίτρο βενζίνης. Το πρόγραμμα θα δέχεται από το χρήστη για κάθε γέμισμα που έκανε τον αριθμό λίτρων βενζίνης που έβαλε και τα χιλιόμετρα που έκανε. Η διαδικασία θα συνεχίζεται μέχρι ο χρήστης να βάλει την τιμή –1 για αριθμό λίτρων. Για κάθε γέμισμα του αυτοκινήτου θα υπολογίζεται πόσα χιλιόμετρα έκανε ανά λίτρο βενζίνης (υπολογίζεται ως αριθμός_χιλιομέτρων / αριθμός_λίτρων_βενζίνης) και στο τέλος (αφού έχει εισάγει ο χρήστης την τιμή –1) θα εμφανίζεται η συνολική μέση κατανάλωση του αυτοκινήτου, δηλ. το άθροισμα της μέσης κατανάλωσης για κάθε γέμισμα δια το πλήθος γεμισμάτων που εισήγαγε ο χρήστης.

Επιτρεπτές τιμές: Τα λίτρα πρέπει να είναι από 1 έως 70, τα χιλιόμετρα πρέπει να είναι από 1 έως 800.

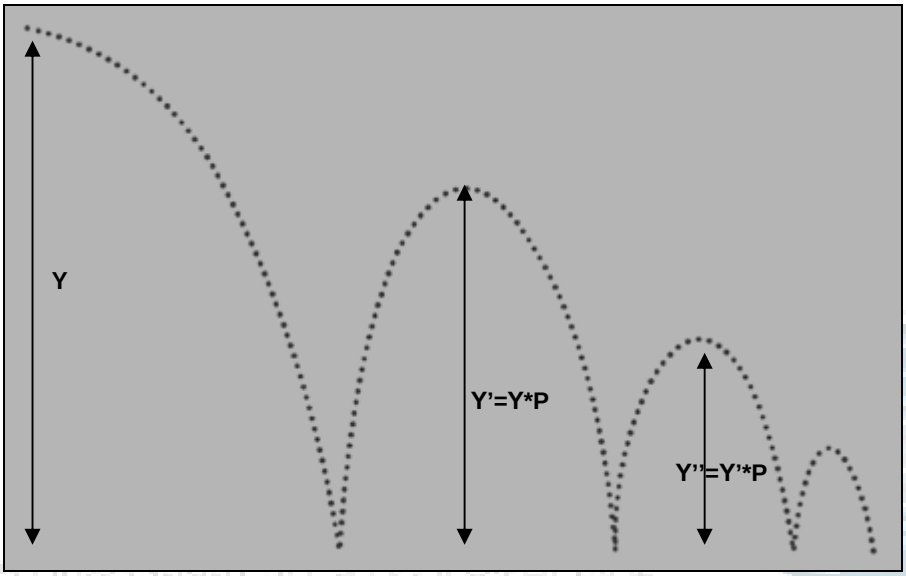
Σημείωση: Η εισαγωγή των τιμών να γίνεται με τη βοήθεια κατάλληλων βοηθητικών μηνυμάτων που θα εμφανίζονται στην οθόνη όταν το πρόγραμμα αναμένει είσοδο από τον χρήστη: Π.χ.

```
Gemisma 1 - Dwse litra benzinhs: 50
Gemisma 1 - Dwse xiliometra: 630
Mesi Katanalwsh - Gemisma 1 : 12.60 km/lt
Gemisma 1 - Dwse litra benzinhs: 50
Gemisma 1 - Dwse xiliometra: 723
Mesi Katanalwsh - Gemisma 1 : 14.46 km/lt
Gemisma 1 - Dwse litra benzinhs: -1
Synolikh mesi katanalosi : 13.53 km/lt
```

```
#include <stdio.h>
main()
{
	int lt, km, sum_lt, sum_km, count = 0;
	float kat, total_kat, sum_kat = 0.0;

	do {
	     do {
	          printf("Gemisma %d - Dwse litra benzinhs: ", count+1);
	          scanf("%d", &lt);
	     } while((lt < 1 || lt > 70) && lt != -1);

	     if (lt != -1)
	     {
	            do {
	                 printf("Gemisma %d - Dwse xiliometra: ", count+1);
	                 scanf("%d", &km);
	            } while(km < 1 || km > 800);

	            count++;

	            kat = (float)km / lt;
	            sum_kat +=kat;

	            printf("Mesi Katanalwsh - Gemisma %d : %.2f\n", count, kat);
	     }
	} while(lt != -1);

	total_kat = sum_kat / count;
	printf("Synolikh mesi katanalosi : %.2f\n", total_kat);
}
```

Άσκηση 6η

Ένα μπαλάκι του τένις αφήνεται να πέσει από ύψος Υ και ξεκινά να αναπηδά. Λόγω της ενέργειας που απορροφάται εξαιτίας της ελαστικότητας κατά την πρόσκρουσή στο έδαφος, μετά από κάθε αναπήδηση το μπαλάκι φτάνει σε ένα ποσοστό P (%) του ύψους που είχε φτάσει στην προηγούμενη αναπήδηση (το ύψος εξαρτάται από το υλικό του γηπέδου). Να δημιουργηθεί πρόγραμμα το οποίο θα δέχεται ως είσοδο το αρχικό ύψος Υ και το ποσοστό P και θα υπολογίζει (και θα εμφανίζει) το ύψος της κάθε αναπήδησης και το πλήθος των αναπηδήσεων θεωρώντας ότι σταματά όταν το ύψος αναπήδησης γίνει μικρότερο του 10% του αρχικού ύψους Υ. (*Σε οποιαδήποτε μεταβλητή χρησιμοποιήσετε πραγματικές τιμές να ορίσετε την ακρίβεια στα 2 δεκαδικά ψηφία).




```
#include <stdio.h>
main()
{
      int Y,EP;
      float K, P;

      printf("Dose ypsos ap' opou afises to mpalaki ");
      scanf("%d", &Y);

      printf("\n Dose pososto anapidisis (se dekadiko) ");
      scanf("%f", &P);

      K=Y;
      EP=0;

      do{

          K=P*K;
          EP++;
          printf("\n Anapidise sta %.2f", K);
     }while(K>=0.1*Y);

     printf("\n Telika ekane %d anapidiseis\n", EP);
}
```