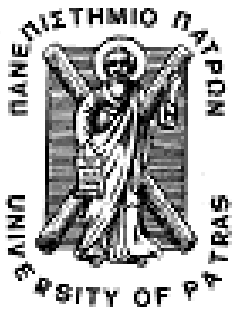
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ



Εισαγωγη στο Διαδικαστικο Προγραμματισμο (2009-2010)

Υπευθυνοι Διδασκοντες Εργαστηριου: Α. Φωκα, Κ. Σταμος

3ο ΣΕΤ ΑΣΚΗΣΕΩΝ

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/) .

- Συναρτήσεις (κεφάλαιο Functions)
- Πίνακες (κεφάλαιο Arrays)
- Συμβολοσειρές (κεφάλαιο Strings)
- Χρήση Βιβλιοθηκών (κεφάλαιο Libraries)

Άσκηση 1η

Σύμφωνα με τους κανόνες σωστής μορφοποίησης ενός κειμένου, ένα σημείο στίξης (κόμμα ή τελεία) τοποθετείται αμέσως μετά την προηγούμενη λέξη και μεσολαβεί ένα κενό πριν την επόμενη. Φτιάξτε ένα πρόγραμμα που δέχεται από το χρήστη μια ακολουθία λέξεων και να την μορφοποιεί σωστά σύμφωνα με το παραπάνω. Θεωρείστε ότι η είσοδος του χρήστη δεν θα περιέχει συνεχόμενα σημεία στίξης.
Παράδειγμα εκτέλεσης του προγράμματος (με **έντονα** η είσοδος του χρήστη):

```
Dwste mia protasi: The good ,the bad and the ugly . Starring Clint Eastwood.
The good, the bad and the ugly. Starring Clint Eastwood.
```

```
#include <stdio.h>
#define N 255

main()
{
   char string[N];
   char buffer[N];
   int i,j, words;
   char c;

   i = 0;
   printf("Dwste mia protasi: ");
   do{
        string[i++] = getchar();
   } while(string[i-1] != '\n');

   i = 0;
   while(string[i] != '\0')
   {
        if ((i+1 < N) && (string[i+1] != '\0'))
        {
          if (string[i] == ' ')
          {
            if ( (string[i+1] == ',') || (string[i+1] == '.') || (string[i+1] == ' ') )
            {
              /*This space should not be printed*/
              i++;
              continue;
            }
          }

          if (string[i] == ',' || string[i] =='.')
          {
            if (string[i+1] != ' ')
            {
              /*An additional space should be printed*/
              printf("%c ", string[i]);
              i++;
              continue;
            }
```

```
            }
        }

        printf("%c", string[i]);
        i++;
    }

}
```

Άσκηση 2η

Να κατασκευάσετε πρόγραμμα το οποίο να κάνει τη μετατροπή δεκαδικών ψηφίων (0…9) στον κώδικα Μορς και αντίστροφα.

Συγκεκριμένα ο χρήστης θα επιλέγει τον τύπο της μετατροπής που θέλει να κάνει και κατόπιν θα εισάγει είτε μία σειρά δεκαδικών ψηφίων είτε μία σειρά τελείες / παύλες και το πρόγραμμα θα εκτυπώνει τον ισοδύναμο κώδικα Μορς ή τα ισοδύναμα ψηφία αντίστοιχα.

Παράδειγμα εκτέλεσης του προγράμματος (με **έντονα** η είσοδος του χρήστη):

```
1 - Ari8mos se kwdika Morse
2 - Kwdikas Morse se ari8mo
0 - Eksodos
1
Dwse ari8mo : 2009
O isodynamos kwdikas Morse einai : ..--- ----- ----- ----.
1 - Ari8mos se kwdika Morse
2 - Kwdikas Morse se ari8mo
0 - Eksodos
2
Dwse kwdika Morse: .--------.--...---..
O isodynamos ari8mos einai : 1978
1 - Ari8mos se kwdika Morse
2 - Kwdikas Morse se ari8mo
0 - Eksodos
0
```


Υπόδειξη: Εσωτερικά στο πρόγραμμά σας μπορείτε να αποθηκεύετε σε ένα δυσδιάστατο πίνακα την αναπαράσταση των ψηφίων σε κώδικα Μορς (http://en.wikipedia.org/wiki/Morse_code).

```
#include <stdio.h>
#define N 255

printMenu() {
   printf("\n1 - Ari8mos se kwdika Morse\n");
   printf("2 - Kwdikas Morse se ari8mo\n");
   printf("0 - Eksodos\n");
}

main()
{
   int MorseTable[10][5] =
   {{'-','-','-','-','-'},
    {'.','-','-','-','-'},
    {'.','.','-','-','-'},
    {'.','.','.','-','-'},
    {'.','.','.','.','-'},
    {'.','.','.','.','.'},
    {'-','.','.','.','.'},
    {'-','-','.','.','.'},
    {'-','-','-','.','.'},
    {'-','-','-','-','.'}};
```

```
int i,j,k,found,numdigit,count = 0;
int num[N];
char dotdash,dummy,digit;
char morse[N];
char choice = '1';

while (choice != '0') {
   printMenu();
   scanf("%c",&choice);

   /*A dummy read to clean the stdin*/
   scanf("%c", &dummy);
   if (choice == '1') {
      printf("Dwse psifia: ");
      i = 0;
      do {
         scanf("%c", &digit);
         if (digit == '0') {
            num[i++] = 0;
         } else if (digit == '1') {
            num[i++] = 1;
         } else if (digit == '2') {
            num[i++] = 2;
         } else if (digit == '3') {
            num[i++] = 3;
         } else if (digit == '4') {
            num[i++] = 4;
         } else if (digit == '5') {
            num[i++] = 5;
         } else if (digit == '6') {
            num[i++] = 6;
         } else if (digit == '7') {
            num[i++] = 7;
         } else if (digit == '8') {
            num[i++] = 8;
         } else if (digit == '9') {
            num[i++] = 9;
         }
      } while (digit != '\n');
      count = i;

      printf("\nO isodynamos kwdikas Morse einai: ");
      for (i=0; i<count; i++) {
         for (j=0; j<5; j++) {
            printf("%c",MorseTable[num[i]][j]);
         }
         printf(" ");
      }
   }
   else if (choice == '2') {
      i = 0;
      printf("Dwse kwdika Morse : ");
      do {
         scanf("%c",&dotdash);
         morse[i++] = dotdash;
      } while(morse[i-1] != '\n');

      printf("\nO isodynamos ari8mos einai: ");
      i = 0;
      while (morse[i] != '\0') {
         /* Traverse all ten digits*/
         k = 0;
         found = 0;
         while (k<10 && !found) {
            /* Check every five chars to match a digit*/
            j = 0;
            found = 1;
            while (j<5 && found) {
               if (morse[i+j] != MorseTable[k][j]) {
                  found = 0;
               }
               j++;
            }

            /* If found is 1 at this point, it means we found our digit at row k*/
            if (found) {
```

```
                    printf("%d",k);
                }
                k++;
            }

            /*Move five chars forward*/
            i += 5;
        }
    }
}
```

Άσκηση 3η

Γράψτε ένα πρόγραμμα το οποίο να παράγει N=1000 τυχαίους θετικούς ακεραίους αριθμούς μεταξύ [1, K] , όπου ακέραιος K > 1, και να κατασκευάζει το ιστόγραμμά τους. Η συχνότητα εμφάνισης κάθε αριθμού να αποθηκεύεται σε έναν πίνακα. Ένα παράδειγμα φαίνεται στο παρακάτω σχήμα για K=9 :

```
1 | *****
2 | ***
3 | ****
4 | **
5 | *******
6 | *****
7 | ***
8 | ****
9 | ***
```

Κάθε αστέρι αντιστοιχεί σε 5 εμφανίσεις του αριθμού. Π.χ. αν ένας αριθμός έχει συχνότητα 18 τότε θα εμφανιστούν 18 / 5 = 3 αστέρια.

Για την παραγωγή τυχαίων αριθμών χρησιμοποιήστε την γεννήτρια τυχαίων αριθμών rand() της γλώσσας C όπως στο παρακάτω παράδειγμα για την παραγωγή N τυχαίων αριθμών στο πεδίο τιμών [1,9].

Παράδειγμα:

```
#include <stdlib,.h>
#include <time.h>

/* γεννήτρια παραγωγής τυχαίων αριθμών με βάση την συνάρτηση srand() */
srand((int)time(NULL));

/* συνάρτηση για την παραγωγή τυχαίου αριθμού μέσα σε καθορισμένο πεδίο
   τιμών. */
int RandomInteger(int low, int high)
{
   int k;
   double d;

   d = (double) rand() /  ((double)RAND_MAX + 1);
   k = (int) (d * (high - low + 1));

   return (low + k);
```

```
}

main()
{
	for (i = 0; i < N; i++){
		number = RandomInteger(1, 9);
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define N 1000
#define K 9
int RandomInteger(int low,int high)
{
	int l;
	double d;
	d=(double)rand()/((double)RAND_MAX+1);
	l=(int)((high-low+1)*d);
	return(low+l);
}

main()
{
	int A[K],i,m,number;

	srand((int)time(NULL));

	for(i=0;i<K;i++)
		A[i]=0;

	for(i=1;i<=N;i++)
	{
		number=RandomInteger(1,K);
		A[number-1]++;
	}
	for(i=0;i<K;i++)
	{
		m=A[i]/5;
		printf("%d",i+1);
		printf("|");
		while(m>0)
		{
		printf("*");
		m--;
		}
		printf("\n", A[i]);
	}
}
```


Άσκηση 4η

Γράψτε μια συνάρτηση που να υπολογίζει το άθροισμα δύο δυαδικών αριθμών. Η συνάρτηση να δέχεται ως όρισμα τρεις πίνακες ακεραίων (A,B,C), οι δύο (A,B) αποτελούν την είσοδο και έχουν μέγεθος N και ο τρίτος (C) είναι το αποτέλεσμα και έχει μέγεθος N+1. Θεωρείστε ότι οι πίνακες A και B αποτελούνται μόνο από “μηδέν” και “ένα” (δεν απαιτείται έλεγχος από εσάς).

Η main() παίρνει τιμές για τους δυαδικούς Α, Β από τον χρήστη, καλεί την συνάρτηση για να εκτελεστεί η πρόσθεση και τέλος εμφανίζει το αποτέλεσμα στην οθόνη.

Η είσοδος των δυαδικών αριθμών από το πληκτρολόγιο γίνεται δίνοντας ο χρήστης όλα τα ψηφία του αριθμού και στο τέλος πατάει enter. ΔΕΝ δίνει ένα-ένα τα ψηφία.

Εκτός από τη συνάρτηση για την πρόσθεση μπορούν να υλοποιηθούν και άλλες βοηθητικές συναρτήσεις αν χρειάζεται.

Η στοιχειώδης πρόσθεση μεταξύ δυαδικών ψηφίων έχει ως εξής:
0 + 0 = 0
1 + 0 = 1
0 + 1 = 1
1 + 1 = 0 και 1 κρατούμενο για ανώτερη τάξη.

```
#include <stdio.h>
#include <string.h>
#define N 41

void convertToArray(int array[N], char s[N])
{
     int j,i;
     int l = strlen(s);

     for(i = l-1; i >=0; i--)
     {
          if (s[l-1-i] == '1')
             array[i] = 1;
          else
             array[i] = 0;
     }

     for(j = N-1; j >= strlen(s); j--)
          array[j] = -1;
}

void printBinary(int a[], int n)
{
     int i;

     for (i=n-1; i >= 0; i--)
          if (a[i] != -1)
             printf("%d", a[i]);

     printf("\n");
}

int addbin(int a, int b, int C[N+1], int pos)
{
     int flag=0;

     if (a == 1 && b == 1)
     {
          C[pos] = 0;
```

```
          flag++;
     }
     else
          C[pos] = a + b;

     return (flag);
}

void add(int A[N], int B[N], int C[N+1])
{
     int i, j, flag;
     flag = 0;
     i = 0;

     while(A[i] != -1 && B[i] != -1 && i < N)
     {
          if (flag > 0)
          {
               flag = addbin(A[i], B[i], C, i);
               addbin(1, C[i], C, i);
          }
          else
               flag = addbin(A[i], B[i], C, i);

          printf("%d: %d + %d = %d ¦ %d\n", i, A[i], B[i], C[i], flag);
          i++;
     }

     while(i < N)
     {
          if (A[i] == -1 && B[i] == -1)
             if (flag > 0)
                flag = addbin(1, 0, C, i);
             else
                C[i] = -1;
             else if (A[i] == -1)
                flag = addbin(B[i], flag, C, i);
             else
                flag = addbin(A[i], flag, C, i);

          printf("%d: %d + %d = %d - %d\n", i, A[i], B[i], C[i], flag);
          i++;
     }

     if (flag > 0)
          C[i] = 1;
     else
          C[i] = -1;
}

main()
{
     char n1[N+1], n2[N+1];
     int A[N], B[N], C[N+1];
     int i;

     printf("1os dyadikos arithmos megistou megethous %d: ", N);
     scanf("%s", &n1);

     printf("2os dyadikos arithmos megistou megethous %d: ", N);
     scanf("%s", &n2);
```

```
    convertToArray(A, n1);
    printBinary(A, N);
    convertToArray(B, n2);
    printBinary(B, N);
    printf("---------------\n");

    add(A, B, C);

    printBinary(C, N+1);
}
```

Άσκηση 5η

Γράψτε ένα πρόγραμμα που να χρησιμοποιεί μια συνάρτηση gcd(),

```
int gcd(int num1, int num2)
```

που να υπολογίζει τον μέγιστο κοινό διαιρέτη δύο ακεραίων (Greatest Common Divisor – gcd) αριθμών. Δείξτε μια αναδρομική και μια μη αναδρομική λύση.

```
#include <stdio.h>

int gcd_loop(int num1, int num2)
{
    int dividor;

    if ( num1>num2 )
        dividor = num2;
    else
        dividor = num1;

     while(dividor>0 && !(num1%dividor==0 && num2%dividor==0))
         dividor--;

    return dividor;
}

int gcd_recursive(int num1, int num2)
{
    if (num1 == 0)
         return(num2);

    if (num2 == 0)
         return(num1);

    return(gcd_recursive(num2, num1 % num2));
}

int main()
{
    int num1, num2;

    printf("Dwse duo ari8mous: ");
    scanf("%d %d", &num1, &num2);

    if (num1 == 0 && num2 == 0)
```

```
          exit(-1);

     if (num1 < 0)
          num1 = -num1;
     if (num2 < 0)
          num2 = -num2;

     printf("\nGCD: %d\n", gcd_loop(num1, num2));
     printf("\nGCD: %d\n", gcd_recursive(num1, num2));
}
```