ΤΕΙ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΠΟΛΥΜΕΣΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΚΑΙ ΑΝΑΠΤΥΞΗΣ ΑΛΓΟΡΙΘΜΩΝ

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ
ΕΡΓΑΣΤΗΡΙΟ

Λύσεις ασκήσεων

Ι. ΞΕΖΩΝΑΚΗΣ

ΗΡΑΚΛΕΙΟ 2015

ΕΙΣΑΓΩΓΗ

Το φυλλάδιο αυτό περιέχει τις λύσεις των προτεινόμενων ασκήσεων του εργαστηρίου Δομών Δεδομένων, του Τμήματος ΕΠΠ.

ΕΡΓΑΣΤΗΡΙΟ 1

Α'. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ

1. Να γραφεί ένα πρόγραμμα, το οποίο να διαβάζει ένα ακέραιο και να τον μετατρέπει στον αντίστοιχο δυαδικό. Η μετατροπή γίνεται με τη μέθοδο των συνεχών διαιρέσεων. Ο αριθμός δηλαδή διαιρείται συνεχώς με το 2, όσο το πηλίκο είναι διάφορο του μηδενός και παίρνουμε το υπόλοιπο αυτής της ακέραιας διαίρεσης.

Τα υπόλοιπα να καταχωρούνται σε ένα πίνακα ακεραίων, Ν θέσεων. Ο δυαδικός αριθμός αποτελείται από αυτά τα υπόλοιπα, αλλά από το τέλος προς την αρχή. Να ορίσετε το Ν αρκετά μεγάλο, ώστε ο πίνακας να «χωρέσει» τον μέγιστο δυαδικό αριθμό που θα προκύψει.

```
#include <stdio.h>

#define N 15

int main( )
{
     int array[N], ak, k, j;

     for (k=0; k<N; k++)
          array[k] = 3;

     k=0;

     scanf ("%d", &ak);

     while (ak / 2 != 0)
     {
          array[k] = ak % 2;

          ak = ak / 2;

          k++;
     }
     array[k] = 1;

     for (k=0; k<N; k++)
           if (array[k] == 3)
                break;

     for (j=k-1; j>=0; j--)
           printf ("%3d%", array[j]);

    printf ("\n");
    return 0;
}
```

2. Μελετείστε το παρακάτω πρόγραμμα, στο οποίο χρησιμοποιούμε τρείς συναρτήσεις. Μέσω της fstore ( ) τοποθετούμε δεδομένα στον πίνακα array. Μέσω της fretrieve ( ) τα τυπώνουμε στην οθόνη και μέσω της fedit ( ) τα επεξεργαζόμαστε ως ένα βαθμό. Μας δίνεται η δυνατότητα να αλλάξουμε τα στοιχεία που είχαμε εισαγάγει με την fstore ( ).

```
#include <stdio.h>

#define  N  5

void fstore (int [ ]);
void fretrieve (int [ ]);
void fedit (int [ ]);


int main( )
{
    int array[N];

    printf("Balte times sto pinaka:\n");
    fstore(array);
    fretrieve(array);
    fedit(array);
    fretrieve(array);
    return 0;
}


void fstore (int a[ ]) {
    int i;

    for (i=0; i<N; i++)
        scanf("%d", &a[i]);
 }


void fretrieve (int a[ ]) {
    int i;

    for (i=0; i<N; i++)
        printf("%6d", a[i]);
    printf("\n");
}
```

```
void fedit (int a[ ]) {
    int i, q;

    for (i=0; i<N; i++)
    {
        printf ("Prev.data:%d\nEnter 1 to edit 0 to skip", a[i]);
        scanf ("%d", &q);
        if (q == 1)
        {
            printf("Enter new Value: ");
            scanf("%d", &a[i]);
        }
    }
}
```

ΕΡΓΑΣΤΗΡΙΟ 2

Α’. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 2 καλύπτονται τα παρακάτω θέματα:

- Πίνακες δύο διαστάσεων.
- Πίνακες συμβολοσειρών.
- Συμμετρικοί, τριγωνικοί πίνακες.
- Αραιοί πίνακες.

Β’. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ

1. Να γραφεί ένα πρόγραμμα, στο οποίο να δηλώσετε ένα δισδιάστατο πίνακα ακεραίων, τον **grades[ ][ ]**. Ο πίνακας να αποτελείται το πολύ από τόσες γραμμές, όσοι είναι οι μαθητές μιας τάξης και από 5 στήλες, όσος είναι ο μέγιστος αριθμός των τεστ, στα οποία έχουν υποβληθεί οι μαθητές

Να δώσετε τιμές στον πίνακα grades και στη συνέχεια να δημιουργήσετε μια συνάρτηση, την **better( )**, η οποία να βρίσκει και να επιστρέφει στη main( ) τον μεγαλύτερο από τους βαθμούς αυτούς, ο οποίος και να εμφανίζεται στην οθόνη. Ο πίνακας grades δεν είναι απαραίτητο να γεμίσει ολόκληρος ούτε ως προς τον αριθμό των μαθητών ούτε ως προς το πλήθος των τέστ. Όταν καλείται η συνάρτηση better( ), αυτή να ενημερώνεται για το πόσες γραμμές και πόσες στήλες του πίνακα να ερευνήσει

Η υλοποίηση γίνεται με δύο τρόπους. Για τον πρώτο τρόπο δείτε τα σχόλια / 1 */ και για τον δεύτερο τα σχόλια /* 2 */*

```
#include <stdio.h>

#define  STUDENTS 15
#define  TESTS 5

int better (int [ ] [TESTS], int, int);          /* 1 */
int better (int *, int);                         /* 2 */

int main( ) {
    int grades [STUDENTS][TESTS];
```

```
    int highest, i, j;
    int num_students = 5;
    int num_tests = 4;
    for (i=0; i< num_students; i++)
        for (j=0; j< num_tests; j++)
            scanf ("%d", &grades[i][j]);

    highest = better (grades, num_students, num_tests);   /*1 */
    highest = better (grades[0], num_students*num_tests);   /* 2 */
    printf("O upsiloteros vathmos einai %d.\n", highest);
    return 0;
}

int better (int a[ ][TESTS], int row, int col) {                    /* 1 */
    int i, j;
    int highest = a[0][0];

    for (i=0; i<row; i++)
        for (j=0; j<col; j++)
            if (a[i][j] > highest)
                highest=a[i][j];
    return highest;
}
int better (int *a, int elem) {                                     /* 2 */
    int i, j;
    int highest = *a;

    for (i=0; i<elem; i++)
        if (*(a + i) > highest)
            highest = *(a + i);
    return highest;
}
```


2. Στη main( ) ενός προγράμματος να δηλώσετε ένα δισδιάστατο πίνακα χαρακτήρων, RxC, τον **students**. Στον πίνακα αυτόν να καταχωρήσετε R ονόματα φοιτητών (συμβολοσειρές) από το πληκτρολόγιο, Στη συνέχεια:

- Να εμφανίσετε στην οθόνη τα ονόματα με τη σειρά που τα καταχωρήσατε στον πίνακα.
- Να κατατάξετε τα ονόματα στον πίνακα αλφαβητικά και να τα εμφανίσετε ξανά στην οθόνη.

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

#define R 5
#define C 20

int main ( ) {
   char ch;
   int i,j;
   char students[R][C];
   char copy[C];

   for (i=0;i<R;i++) {
      printf ("Dwse Onoma\n");
      scanf ("%s",students[ i ] );
   }

   printf ("Ta onomata opws ta dwsate einai\n");

   for (i=0;i<R;i++) {
         printf ("O foiththns %d onomazetai " , i+1);
         puts (students[ i ] );
    }

   printf ("As ftiaxoume ton pinaka alphabitika\n");

   for (i=0; i<R-1 ; ++i) {
        for (j=i+1; j<R ; ++j) {
             if (strcmp(students[ i ], students[ j ] )>0) {
                  strcpy (copy, students[ i ] );
                  strcpy (students[ i ], students[ j ] );
                  strcpy (students[ j ], copy); }
      }
   }

  printf ("O pinakas alphabitika\n");

   for (i=0; i<R; i++) {
         printf ("O mathitis %d onomazetai ", i+1 );
         puts (students[ i ] );
   }
   system ("pause");
}
```

ΕΡΓΑΣΤΗΡΙΟ 3

Α'. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 3 καλύπτονται τα παρακάτω θέματα:

- Δομές (structures): Περιγραφή, πεδία δομής, δηλώσεις και δεδομένα στις δομές.
- Φωλιασμένες δομές.

Β'. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ

Δομές: Περιγραφή, πεδία δομής. Δηλώσεις και δεδομένα στις δομές.

1. Τι θα εμφανιστεί στην οθόνη μετά την εκτέλεση του πιο κάτω προγράμματος και γιατί.

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

struct complex { float re, im; };      // Περιγραφή νέου τύπου

int main( )
{
    struct complex z1, z2;             // Δήλωσεις μεταβλητών
    printf("sizeof = %d \n", sizeof (struct complex));
    printf("give z1.re -> ");
    scanf( "%f", &z1.re );             // Διάβασμα
    z1.im = 55;
    z2 = z1;                           // Επιτρεπτή εκχώρηση τιμής
    printf("z2 = %f + i * %f \n", z2.re, z2.im );
    system("pause");
    return 0;
}
```

Θεωρώντας ότι πληκτολογήσαμε το 5, θα εμφανίσει:

```
sizeof = 8
give z1.re -> 5
z2 = 5.000000 + i * 55.000000
```

2. Σε συνέχεια της παραπάνω άσκησης να γράψετε ένα πρόγραμμα, στο οποίο να διαβάζετε από το πληκτρολόγιο τιμές για δύο μιγαδικούς αριθμούς. Θα διαβάζετε επίσης ένα χαρακτήρα. Αν ο χαρακτήρας είναι το + θα κάνετε πρόσθεση των μιγαδικών αριθμών, αν είναι – θα κάνετε αφαίρεση, αν είναι * θα κάνετε πολλαπλασιασμό και αν είναι / θα κάνετε διαίρεση. Για οποιονδήποτε άλλο χαρακτήρα θα γράφεται στην οθόνη «Λάθος», θα ζητείται καινούργιος χαρακτήρας και θα επαναλαμβάνεται όλη η διαδικασία. Το αποτέλεσμα της πράξης να γράφεται στην οθόνη.

Επεξηγήσεις - υπενθυμίσεις:

Έστω δύο μιγαδικοί αριθμοί, οι z1=a+jb και z2=c+jd. Τότε:

- w1 = z1+z2 = (a+c) + j (b+d)
- w2 = z1-z2 = (a-c) + j (b-d)
- w3 = z1*z2 = (ac-bd) + j (bc+ad)
- w4 = z1/z2 = $(ac+bd)/(c^2+d^2)$ + j $(bc-ad)/(c^2+d^2)$

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

struct complex { float re;  float im; };

int main( )
{
        struct complex z1, z2, z3;
        char ch;

        printf ("Dwste pragmatiko meros 1ou ari8mou ");
        scanf ( "%f", &z1.re );
        printf ("Dwste fantasiko meros 1ou ari8mou ");
        scanf ( "%f", &z1.im );
        printf ("Dwste pragmatiko meros 2ou ari8mou ");
        scanf ( "%f", &z2.re );
        printf ("Dwste fantasiko meros 2ou ari8mou ");
        scanf ( "%f", &z2.im );
        printf ("Dwste xarakthra");
        ch = getche( );
        switch (ch)
        {
            case '+' :
                    z3.re = z1.re + z2.re;
                    z3.im = z1.im + z2.im;
                    break;
            case '-' :
                     z3.re = z1.re - z2.re;
                     z3.im = z1.im - z2.im;
                     break;
```

```
            case '*' :
                    z3.re = z1.re*z2.re - z1.im*z2.im;
                    z3.im = z1.im*z2.re + z1.re*z2.im;
                    break;
            case '/' :
                    z3.re = (z1.re*z2.re+z1.im*z2.im) / (z2.re*z2.re +
                            z2.im*z2.im);
                    z3.im = (z1.im*z2.re-z1.re*z2.im)/(z2.re*z2.re +
                            z2.im*z2.im);
                    break;
        }
        printf("Apotelesma:\n");
        printf("z3 = %f + i * %f \n", z3.re, z3.im );
        system("pause");
        return 0;
}
```

ΕΡΓΑΣΤΗΡΙΟ 4

Α’. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 4 καλύπτονται τα παρακάτω θέματα:

- Πίνακες δομών.
- Δομές ως παράμετροι και ως τιμή επιστροφής συναρτήσεων.
- Δείκτες σε δομές.

Β’. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ

Πίνακες δομών.

1. Πληκτρολογήστε και εκτελέστε το πιο κάτω πρόγραμμα και παρατηρείστε τι κάνει:

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

#define N 100

struct student { int am; char name[30]; };

int main( )
{
      struct student pin[N];                  // Πίνακας δομών
      int i;

      for (i=0; i<N; i++)                     // Διάβασμα
      {
          printf ("Dwste AM toy spoudasth %d -> ", i );
          scanf ("%d", &pin[i].am );          // Προσέξτε το &
          printf ("Dwste onoma -> ");
          gets (pin[i].name);
      }
      for (i=0; i<N; i++)                     // Εκτύπωση
      {
          printf ("Spoudasths %2d, AM=%4d, Onoma=%s \n",
                        i, pin[i].am, pin[i].name );
      }
      system("pause");
      return 0;
}
```

2. Σε ένα πρόγραμμα να δηλώσετε ένα πίνακα Ν δομών για υπαλλήλους. Για κάθε υπάλληλο να τηρούνται σε μιά δομή τα εξής στοιχεία: όνομα, επίθετο, ηλικία, έτη υπηρεσίας και μισθός. Αφού γεμίσετε τον πίνακα, το πρόγραμμά σας να εμφανίζει το όνομα και το επίθετο των εξής υπαλλήλων:

- Αυτού που παίρνει τον μεγαλύτερο μισθό
- Του μεγαλύτερου σε ηλικία
- Αυτού που έχει τα περισσότερα χρόνια υπηρεσίας.

Αν αυτά τα κριτήρια τα πληρούν πολλοί υπάλληλοι, το πρόγραμμα να εμφανίζει τα στοιχεία μόνο του πρώτου που θα συναντήσει στον πίνακα.

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

#define N 5

struct person {
        char onoma[20];
        char epith[30];
        int  age;
        int  years;
        float salary;  };

int main( )
{
        struct person pinax[N];
        int k, max_age, max_years, pma=0, pmy=0, pms=0;
        float max_salary;
        char temp[10];

        printf ("Gemisma pinaka\n");
        for (k=0; k<N; k++)  {
                printf ("Onoma %dou ypallhlou ", k+1);
                gets (pinax[k].onoma);
                printf ("Epi8eto %dou ypallhlou ", k+1);
                gets (pinax[k].epith);
                printf ("Hlikia %dou ypallhlou ", k+1);
                gets (temp);
```

```
            pinax[k].age = atoi(temp);
            printf ("Eth yphresias %dou ypallhlou ", k+1);
            gets (temp);
            pinax[k].years = atoi(temp);
            printf ("Mis8os %dou ypallhlou ", k+1);
            gets (temp);
            pinax[k].salary = atof(temp);
            printf ("*******************************\n");
        }
        max_age = pinax[0].age;
        max_years = pinax[0].years;
        max_salary = pinax[0].salary;

        for (k=0; k<N; k++)  {
            if (pinax[k].age > max_age) {
                max_age = pinax[k].age;
                pma = k;}
            if (pinax[k].years > max_years) {
                max_years = pinax[k].years;
                pmy = k;}
            if (pinax[k].salary > max_salary) {
                max_salary = pinax[k].salary;
                pms = k;}
        }

        printf ("Ton megalytero mis8o pairnei o: %s %s\n",
        pinax[pms].onoma, pinax[pms].epith);
        printf ("O megalyteros se hlikia einai o: %s %s\n",
        pinax[pma].onoma, pinax[pma].epith);
        printf("Ta perissotera xronia yphresias exei o: %s %s\n",
        pinax[pmy].onoma, pinax[pmy].epith);
        system("pause");
        return 0;
}
```

ΕΡΓΑΣΤΗΡΙΟ 5

Α’. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 5 καλύπτονται τα παρακάτω θέματα:

- Δυναμική δέσμευση μνήμης.
- Συναρτήσεις malloc, calloc, realloc.
- Πίνακες δεικτών.
- Δυναμικοί πίνακες.

Β’. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ

1. Να γραφεί ένα πρόγραμμα στο οποίο θα διαβάζετε ένα ακέραιο από το πληκτρολόγιο, τον num. Στη συνέχεια, με την χρήση της calloc( ) να δεσμεύσετε χώρο για να τοποθετήσετε num ακεραίους. Να εμφανίσετε τα περιεχόμενα του χώρου που δεσμεύτηκε στην οθόνη. Στη συνέχεια να διαβάσετε num ακέραιους, τους οποίους να τοποθετήσετε στον χώρο που δεσμεύτηκε και να εμφανίσετε ξανά τα περιεχόμενα του χώρου αυτού στην οθόνη.

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

int main()
{
      unsigned num;
      int *ptr, k;

      scanf ("%d", &num);
      ptr = (int*) calloc (num, sizeof(int));
      for (k=0; k<num; k++)
            printf ("%5d\n", *(ptr+k));
      printf ("%\n");
      for (k=0; k<num; k++)
            scanf ("%d", ptr+k);
      printf ("%\n");
      for (k=0; k<num; k++)
            printf ("%5d\n", *(ptr+k));
      system ("pause");
      return 1;
}
```

2. Να διαβάσετε μια συμβολοσειρά από το πληκτρολόγιο, την pin. Να δεσμεύσετε με την malloc( ) όσο χώρο χρειάζεστε για την αποθήκευση της συμβολοσειράς αυτής. Στη συνέχεια να διαβάσετε μια άλλη συμβολοσειρά, την mat, την οποία να αποθηκεύσετε στον χώρο της pin, μεγαλώνοντάς τον ή μικραίνοντάς τον με την χρήση της realloc( ). Στη συνέχεια να γράψετε στην οθόνη την pin.

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>

int main( )
{
      char pin[80], mat[80], *mes;

      puts ("Dwste prwth symboloseira");
      gets (pin);
      mes = (char *) malloc (strlen (pin)+1);
      strcpy (mes, pin);
      puts ("Dwste deyterh symboloseira");
      gets (mat);
      mes = (char *) realloc (mes, (strlen(mat)+1));
      strcpy (mes,mat);
      puts (mes);
      system ("pause");
      return 1;
}
```

ΕΡΓΑΣΤΗΡΙΟ 6

Α’. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 6 καλύπτονται τα παρακάτω θέματα:

- Στοίβες**.**
- Ουρές**.**

Β’. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ:

Στοίβες:

1. Να γραφεί ένα πρόγραμμα, στο οποίο να χρησιμοποιείτε μια στοίβα θετικών ακεραίων, την οποία να υλοποιήσετε με την χρήση πίνακα ακεραίων Ν θέσεων. Να γράψετε τις συναρτήσεις ώθησης (push) και ανάκλησης (pop) από την στοίβα, στις οποίες να γίνεται έλεγχος πλήρους και κενής στοίβας αντίστοιχα. Το πρόγραμμα να διαβάζει ένα χαρακτήρα από το πληκτρολόγιο, τον ch και:

- Εάν ο χαρακτήρας είναι το U να διαβάζετε ένα ακέραιο και να γίνεται ώθηση του ακεραίου στην στοίβα.
- Εάν ο χαρακτήρας είναι το Ο να γίνεται ανάκληση ακεραίου από την στοίβα, ο οποίος να εμφανίζεται στην οθόνη.
- Εάν ο χαρακτήρας είναι το Ε, να τελειώνει η όλη διαδικασία και να εμφανίζονται στην οθόνη τα περιεχόμενα της στοίβας.
- Εάν ο χαρακτήρας δεν είναι κανείς από τους παραπάνω, να γράφεται στην οθόνη «ΛΑΘΟΣ ΧΑΡΑΚΤΗΡΑΣ» και να διαβάζεται νέος χαρακτήρας από το πληκτρολόγιο.

Αφού τελειώσει η όλη διαδικασία, να γράφονται στην οθόνη οι ακέραιοι, οι οποίοι εξακολουθούν να υπάρχουν στην στοίβα


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

#define N 30

int stack[N];
int top;

void  push (int);
int  pop ( );
void display ( );
```

```
int main ( )
{
      char ch;
      int ak;

      ch = getche ( );
      while (ch != 'E')
      {
            if (ch =='U')
            {
                  scanf ("%d", &ak);
                  push (ak);
            }

            if (ch == 'O')
            {
                  ak = pop ( );
                  if (ak > 0)
                        printf ("ANAKLH8HKE O %d\n", ak);
            }
            if (ch != 'U' && ch != 'O')
                  puts (" LA8OS XARAKTHRAS");
            ch = getche ( );
      }
      display ( );
      system ("pause");
      return 1;
}

void push (int elem)
{
      if (top >= N-1)
            puts ("PLHRHS STOIBA. ADYNATH W8HSH");
      stack[top++] = elem;
}

int pop ( )
{
      if (top == 0)
      {
            puts ("ADEIA STOIBA. ADYNATH ANAKLHSH");
            return -1;
      }
      return (stack[--top]);
}

void display ( )
{
      int k;

      for (k=top-1; k>=0; k--)
            printf ("%d\n", stack[k]);
}
```

2. Να γραφεί ένα πρόγραμμα, στο οποίο να υλοποιήσετε μια ουρά αναμονής θετικών ακεραίων, με την χρήση πίνακα ακεραίων Ν θέσεων. Να γράψετε τις συναρτήσεις ώθησης (qinsert) και ανάκλησης (qremove) από την ουρά, στις οποίες να γίνεται έλεγχος πλήρους και κενής ουράς αντίστοιχα. Το πρόγραμμα να διαβάζει ένα χαρακτήρα από το πληκτρολόγιο, τον ch και:

- Εάν ο χαρακτήρας είναι το U να διαβάζετε ένα ακέραιο και να γίνεται ώθηση του ακεραίου στην ουρά.
- Εάν ο χαρακτήρας είναι το O να γίνεται ανάκληση ακεραίου από την ουρά, ο οποίος να εμφανίζεται στην οθόνη.
- Εάν ο χαρακτήρας είναι το E, να τελειώνει η όλη διαδικασία και να εμφανίζονται στην οθόνη τα περιεχόμενα της ουράς.
- Εάν ο χαρακτήρας δεν είναι κανείς από τους παραπάνω, να γράφεται στην οθόνη «ΛΑΘΟΣ ΧΑΡΑΚΤΗΡΑΣ» και να διαβάζεται νέος χαρακτήρας από το πληκτρολόγιο.

Αφού τελειώσει η όλη διαδικασία, να γράφονται στην οθόνη οι ακέραιοι, οι οποίοι εξακολουθούν να υπάρχουν στην ουρά.

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

#define N 7

int queue [N];
int ip, rp;

void qinsert (int);
int qremove ( );
void display ( );

int main (void)
{
     int ak, apot;
     char ch;

     ch = getche ( );
     while (ch != 'E')
     {
          if (ch =='U')
          {
               scanf ("%d", &ak);
               qinsert (ak);
          }
          if (ch == 'O')
          {
               apot = qremove ( );
               if (apot > 0)
                  printf ("ANAKLH8HKE O %d\n", apot);
          }
          if (ch != 'U' && ch != 'O')
               puts (" LA8OS XARAKTHRAS");
          ch = getche();
     }
```

```
        display ( );
        system ("pause");
        return 1;
}

void qinsert (int elem)
{
        if (ip == N)
                puts ("OYRA PLHRHS. DEN EGINE EISAGWGH");
        else
                queue [ip++] = elem;
}

int qremove ( )
{
        int data;

        if (ip == rp)
        {
                puts ("OYRA KENH");
                return -1;
        }
        else
        {
                data = queue[rp++];
                if (ip == rp)
                        ip = rp = 0;
                return data;
        }
}

void display()
{
        int k;

        for (k=ip-1; k>=rp; k--)
                printf ("%d\n", queue[k]);
}
```

ΕΡΓΑΣΤΗΡΙΟ 7

Α’. ΠΕΡΙΕΧΟΜΕΝΑ ΜΑΘΗΜΑΤΟΣ

Στο εργαστήριο 7 καλύπτονται τα παρακάτω θέματα:

- Απλά συνδεδεμένες λίστες. Δημιουργία.
- Λειτουργίες στις απλά συνδεδεμένες λίστες: αναζήτηση στοιχείου, εισαγωγή κόμβου, διαγραφή κόμβου, μετακίνηση κόμβου, αντιστροφή λίστας, συνένωση λιστών κλπ.

Β’. ΑΣΚΗΣΕΙΣ ΓΙΑ ΕΚΤΕΛΕΣΗ – ΕΠΙΔΕΙΞΗ / ΕΠΕΞΗΓΗΣΕΙΣ

1. Το πρόγραμμα που ακολουθεί ζητεί να δημιουργηθεί μια απλά συνδεδεμένη λίστα. Τα στοιχεία της λίστας είναι του είδους node και περιέχουν το όνομα ενός ατόμου, τον αριθμό μητρώου του και το οφειλόμενο σε αυτόν ποσόν. Στοιχεία εισάγονται στην λίστα μέχρι να δοθεί αριθμός μητρώου μηδέν. Το πρόγραμμα δεν προβλέπει το ενδεχόμενο να δοθεί ίδιος αριθμός μητρώου για δύο άτομα. Συνιστάται να το βελτιώσετε, ώστε να αποκλείεται ένα τέτοιο ενδεχόμενο.
Μετά την καταχώρηση των στοιχείων, το πρόγραμμα, με τη χρήση της συνάρτησης display( ), εμφανίζει στην οθόνη τα ονόματα, τους αριθμούς μητρώου των ατόμων που έχουν καταχωρηθεί στην λίστα και το ποσόν που οφείλεται σε καθένα.
Μελετείστε το πρόγραμμα και αποθηκεύστε το, προκειμένου να εκτελέσετε και τις επόμενες ασκήσεις χρησιμοποιώντας το, χωρίς να ξαναδημιουργείτε την λίστα από την αρχή.


```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>

struct node
{
      char name [30];
      int   am;
      float poson;
      struct node *next;
};

void display (struct node *);

int main (void)
{
      struct node *head, *curr, *ptr;
      char nol[8];
      int num;
```

```
    /* Δημιουργία – Εισαγωγή στοιχείων στη λίστα */

    head = (struct node *) malloc(sizeof (struct node));
    curr = ptr = head;
    curr->next = NULL;
    printf ("DWSTE ARI8MO MHTRWOY ");
    gets (nol);
    num = atoi (nol);

    while (num > 0)
    {
        curr->am = num;
        printf ("DWSTE ONOMA ");
        gets (curr->name);
        printf ("DWSTE POSON ");
        gets (nol);
        curr->poson = atof (nol);
        curr->next = (struct node *) malloc(sizeof (struct node));
        curr = curr->next;
        curr->next = NULL;
        printf ("DWSTE EPOMENO ARI8MO MHTRWOY ");
        gets (nol);
        num = atoi (nol);
    }
    while (ptr->next != curr)
        ptr = ptr->next;
    ptr->next = NULL;
    free (curr);
    curr = NULL;
    display (head);
    system ("pause");
    return 1;
}

void display (struct node *head)
{
    struct node *curr;

    /* Εμφάνιση των στοιχείων της λίστας στην οθόνη */

    printf("\nEMFANISH TWN STOIXEIWN THS LISTAS\n");
    curr = head;
    while (curr != NULL)
    {
        printf ("ONOMA %s\t ARI8MOS MHTRWOY %5d\t POSON
                %10.2f\n", curr->name, curr->am, curr->poson);
        curr = curr->next;
    }
}
```

2. Να γραφεί μία συνάρτηση, την ofeiles, η οποία να αθροίζει τα ποσά που οφείλονται σε όλα τα άτομα της λίστας της άσκησης 1 και να υπολογίζει τον μέσο όρο των οφειλόμενων ποσών στα άτομα της λίστας. Η main( ) να γράφει στην οθόνη το συνολικό οφειλόμενο ποσόν και τον μέσο όρο των οφειλομένων ποσών.

```
float ofeiles (struct node *head, float *mo) {
        struct node *curr;
        int num=0;
        float sum=0;

        curr = head;
        while (curr != NULL) {
                num++;
                sum += curr->poson;
                curr = curr->next;
        }
        *mo = sum / num;
        return sum;
}
```

Στην main( ):

```
float mo, tot_poson;
.....................................
tot_poson = ofeiles (head, &mo);
printf ("\nSYNOLIKO OFEILOMENO POSON: %8.2f\n", tot_poson);
printf ("\nMESOS OROS OFEILOMENWN POSWN ANA ATOMO: %8.2f\n",
          mo);
```