9 Покажувачи

1. Да се напише програма која во низа од N цели броеви ќе го пронајде почетокот и должината на најдолгата растечка подниза.

```
#include "stdio.h"
#define MaxElem 10

void sekvenca(float x[], int n, int *pos, int *len);

void main(void)
{
float a[MaxElem];
int i,n,pos,len;

     printf("Dolzina na nizata: "); scanf("%d",&n);
     for (i=0; i<n; i++)
          scanf("%f",&a[i]);

     sekvenca(a,n,&pos,&len);

     printf("Pocetok:%d, dolzina:%d",pos,len);
}

void sekvenca(float x[], int n, int *pos, int *len)
{
     int i,poc,dolz;

     poc=0; dolz=1;
     *pos=0; *len=1;
     i=0;
     while (i< n-1)
     {
          poc=i;
          dolz=1;
          while ((x[i] < x[i+1]) && (i < n))
          {
               i=i+1;
               dolz=dolz+1;
          }
          if (dolz>*len)
          {
               *len=dolz;
               *pos=poc;
          }
```

```
            i=i+1;
      }
}
```

2. Да се напише процедура која влезната низа a_0, a_1, ..., a_{n-1} ќе ја трансформира во излезна низа b_0, b_1, ..., b_{n-1} на следниот начин:

$$b_0 = a_0 + a_{n-1}$$
$$b_1 = a_1 + a_{n-2}$$
M
$$b_{n-1} = a_{n-1} + a_0$$

На пример, влезната низа 1, 2, 3, 5, 7, треба да се трансформира во 8, 7, 6, 7, 8.

```
#include "stdio.h"
#define Max 100

int* promena(int a[], int n);
void promena(int *a);

int n;

void main(void)
{
int i;
int  a[Max];
int *bp;

      printf("Kolku elementi ima nizata: ");
      scanf("%d",&n);
      printf("Vnesi gi elementite na nizata\n");
      for (i=0; i<n; i++)
            scanf("%d",&a[i]);

      bp=promena(a,n);  // verzija 1
      promena(a);           // verzija 2


      for (i=0; i<n; i++)
            printf("b[i]=%d a[i]=%d\t",*(bp+i),a[i]);
      printf("\n");
}
// ############ VERZIJA 1 ###############
int* promena(int a[], int n)
{
      int i, b[Max];
      for (i=0; i<n; i++)
            b[i]=a[i]+a[n-i-1];
      return b;
```

```
}

// ############ VERZIJA 2 ###############
void promena(int *a)
{
     int i,j;

     for (i=0,j=n-1; i<j; i++,j--)
     {
          *(a+i)+=*(a+j);
          *(a+j)=*(a+i);
     }
     if (n%2)
          *(a+n/2) *= 2;
}
```

3. Да се напише процедура која ќе прифати несортиран вектор А од цели броеви како влезно-излезен параметар и преку истиот параметар ќе го врати векторот во иста секвенца на броевите, откако предходно ќе изврши бришење на броевите кои се дупликати. Бројот на останатите елементи е исто така излезен параметар. На пример, за даден вектор А: 15, 31, 23, 15, 75, 23, 41, 15, 31, 85 од 10 цели броеви, излезниот вектор треба да биде: 15, 31, 23, 75, 41, 85 и должина 6. Да се напише програма за тестирање на оваа процедура.

```
#include "stdio.h"
#define   MAX   100

void bezDuplikati(int a[], int n, int *nbd)
{
     int i, j, brojac;

     brojac=0;
     for (i=0; i<n; i++)
     {
          j=0;
          while ((j <= brojac) && (a[i] != a[j]))
               j++;
          if (j > brojac)
          {
               brojac++;
               a[brojac]=a[i];
          }
     }
     *nbd=brojac+1;
}

void main(void)
{
     int i,n,brElem, a[MAX];
```

```
     printf("Kolku elementi ima nizata: ");
     scanf("%d",&n);
     printf("Vnesi gi elementite na nizata\n");
     for (i=0; i<n; i++)
          scanf("%d",&a[i]);

     bezDuplikati(a,n,&brElem);
     for (i=0; i<brElem; i++)
          printf("%d",a[i]);
}
```

4. Да се напишат процедури за линеарно и бинарно пребарување на вектори. Потоа да се напише програма за нивно тестирање.

```
#include "stdio.h"
#define Max 100

void citajNiza(int x[], int n)
{
     int i;
     for (i=0; i<n; i++)
          scanf("%d",&x[i]);
}

void pecatiNiza(int x[], int n)
{
     int i;
     for (i=0; i<n; i++)
          printf("%d ",x[i]);
     printf("\n");
}

int linearBaraj(int a[], int n, int key)
{
     int i, lb;

     lb=0;
     i=0;
     while (!lb && (i<n))
     {
          if (a[i]==key)
          {
               lb=i;
          }
          i++;
     }
     return lb;
}
```

```
int linearBaraj1(int a[], int n, int key)
{
     int i;

     i=0;
     a[n]=key; // postavuvame strazzar
     while (a[i] != key)
          i++;
     if (i==n)
          return 0;
     else
          return i;
}

int binarnoBaraj(int a[], int n, int key)
{
     int pocetok,kraj,sredina, najden;

     pocetok=0;
     kraj=n-1;
     najden=0;

     while (!najden && (pocetok <= kraj))
     {
          sredina=(pocetok+kraj)/2;
          if (key < a[sredina])
               kraj=sredina-1;
          else
               if (key > a[sredina])
                    pocetok=sredina+1;
          else // key==a[sredina]
               najden=1;
     }
     if (najden)
           return sredina;
     else
          return 0;
}

void main(void)
{
     int a[Max];
     int n,k,l;

     printf("Kolkava niza? \n"); scanf("%d",&n);

     citajNiza(a,n);

     printf("Vlezna niza  : \n");
     pecatiNiza(a,n);
```

```
     printf("Sto da baram? "); scanf("%d",&k);

     l=binarnoBaraj(a,n,k);  // nizata treba da e sortirana
     // l=linearBaraj(a,n,k);
     // l=linearBaraj1(a,n,k);

     if (!l)
          printf("Baraniot element ne e najden.\n");
     else
          printf("Baraniot element e najden na pozicija
%d\n",l);
}
```

5. Неколку различни видови на сортирање: SimpleSort, BubbleSort.

```
#include "stdio.h"

#define Max 100

void citajNiza(int x[], int n)
{
     int i;
     for (i=0; i<n; i++)
          scanf("%d",&x[i]);
}

void pecatiNiza(int x[], int n)
{
     int i;
     for (i=0; i<n; i++)
          printf("%d ",x[i]);
     printf("\n");
}

void swap(int *i,int *j)
{
     int temp;
     temp=*i;
     *i=*j;
     *j=temp;
}

void simpleSort(int a[], int n)
{
     int i,j;
     for (i=0; i<n-1; i++)
          for (j=i+1; j<n; j++)
               if (a[i]>a[j])
```

```
                    swap(&a[i],&a[j]);
}

void bubbleSort(int a[], int n)
{
     int i,j;
     for (i=0; i<n; i++)
          for (j=0; j<n-i-1; j++)
               if (a[j]>a[j+1])
                    swap(&a[j],&a[j+1]);
}

void main(void)
{
     int a[Max], b[Max];
     int n,k,l,c;

     printf("Kolkava niza? \n"); scanf("%d",&n);

     citajNiza(a,n);

     printf("Input  : \n");
     pecatiNiza(a,n);

     printf("po koj metod?\n");
     scanf("%d",&c);

     switch (c)
     {
          case 1:   simpleSort(a,n);
               break;
          case 2:   bubbleSort(a,n);
               break;
          default: printf("nema takov\n");
     }

     printf("Output : \n");
     pecatiNiza(a,n);
}
```

```
Kolkava niza? 10
73 65 52 24 83 17 35 96 41 9

============ SimpleSort ============
Input  : 73 65 52 24 83 17 35 96 41  9
Pass  1: 65 73 52 24 83 17 35 96 41  9
Pass  1: 52 73 65 24 83 17 35 96 41  9
Pass  1: 24 73 65 52 83 17 35 96 41  9
Pass  1: 17 73 65 52 83 24 35 96 41  9
Pass  1:  9 73 65 52 83 24 35 96 41 17
```

```
Pass  2:  9 65 73 52 83 24 35 96 41 17
Pass  2:  9 52 73 65 83 24 35 96 41 17
Pass  2:  9 24 73 65 83 52 35 96 41 17
Pass  2:  9 17 73 65 83 52 35 96 41 24
Pass  3:  9 17 65 73 83 52 35 96 41 24
Pass  3:  9 17 52 73 83 65 35 96 41 24
Pass  3:  9 17 35 73 83 65 52 96 41 24
Pass  3:  9 17 24 73 83 65 52 96 41 35
Pass  4:  9 17 24 65 83 73 52 96 41 35
Pass  4:  9 17 24 52 83 73 65 96 41 35
Pass  4:  9 17 24 41 83 73 65 96 52 35
Pass  4:  9 17 24 35 83 73 65 96 52 41
Pass  5:  9 17 24 35 73 83 65 96 52 41
Pass  5:  9 17 24 35 65 83 73 96 52 41
Pass  5:  9 17 24 35 52 83 73 96 65 41
Pass  5:  9 17 24 35 41 83 73 96 65 52
Pass  6:  9 17 24 35 41 73 83 96 65 52
Pass  6:  9 17 24 35 41 65 83 96 73 52
Pass  6:  9 17 24 35 41 52 83 96 73 65
Pass  7:  9 17 24 35 41 52 73 96 83 65
Pass  7:  9 17 24 35 41 52 65 96 83 73
Pass  8:  9 17 24 35 41 52 65 83 96 73
Pass  8:  9 17 24 35 41 52 65 73 96 83
Pass  9:  9 17 24 35 41 52 65 73 83 96
Output :  9 17 24 35 41 52 65 73 83 96

============= BubbleSort ==============
Input  : 73 65 52 24 83 17 35 96 41  9
Pass  1: 65 73 52 24 83 17 35 96 41  9
Pass  1: 65 52 73 24 83 17 35 96 41  9
Pass  1: 65 52 24 73 83 17 35 96 41  9
Pass  1: 65 52 24 73 17 83 35 96 41  9
Pass  1: 65 52 24 73 17 35 83 96 41  9
Pass  1: 65 52 24 73 17 35 83 41 96  9
Pass  1: 65 52 24 73 17 35 83 41  9 96
Pass  2: 52 65 24 73 17 35 83 41  9 96
Pass  2: 52 24 65 73 17 35 83 41  9 96
Pass  2: 52 24 65 17 73 35 83 41  9 96
Pass  2: 52 24 65 17 35 73 83 41  9 96
Pass  2: 52 24 65 17 35 73 41 83  9 96
Pass  2: 52 24 65 17 35 73 41  9 83 96
Pass  3: 24 52 65 17 35 73 41  9 83 96
Pass  3: 24 52 17 65 35 73 41  9 83 96
Pass  3: 24 52 17 35 65 73 41  9 83 96
Pass  3: 24 52 17 35 65 41 73  9 83 96
Pass  3: 24 52 17 35 65 41  9 73 83 96
Pass  4: 24 17 52 35 65 41  9 73 83 96
Pass  4: 24 17 35 52 65 41  9 73 83 96
Pass  4: 24 17 35 52 41 65  9 73 83 96
Pass  4: 24 17 35 52 41  9 65 73 83 96
Pass  5: 17 24 35 52 41  9 65 73 83 96
Pass  5: 17 24 35 41 52  9 65 73 83 96
Pass  5: 17 24 35 41  9 52 65 73 83 96
Pass  6: 17 24 35  9 41 52 65 73 83 96
Pass  7: 17 24  9 35 41 52 65 73 83 96
Pass  8: 17  9 24 35 41 52 65 73 83 96
Pass  9:  9 17 24 35 41 52 65 73 83 96
```

```
Output :  9 17 24 35 41 52 65 73 83 96
```

6. Да се напише процедура којашто влезната матрица $A_{m\times n}$ ќе ја преуреди така да колоните и се во растечки редослед на максималниот елемент на колоната.
   Пример:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 1 \\ 8 & 6 & 6 & 3 & 3 \\ 0 & 3 & 9 & 2 & 7 \\ 8 & 6 & 9 & 4 & 7 \end{bmatrix} \Rightarrow \begin{bmatrix} 4 & 2 & 1 & 1 & 3 \\ 3 & 6 & 3 & 8 & 6 \\ 2 & 3 & 7 & 0 & 9 \\ 4 & 6 & 7 & 8 & 9 \end{bmatrix}$$

```
#include "stdio.h"

#define Mx 100

void main(void)
{

int m,n,i,j,k;
float a[Mx][Mx], max[Mx], t;

     printf("Vnesi gi dimenziite na matricata\n");
     scanf("%d %d",&m, &n);
     printf("Vnesi ja matricata...\n");
     for (i=0; i<m; i++)
           for (j=0; j<n; j++)
           {
                printf("a[%d,%d]=",i,j);
                scanf("%f",&a[i][j]);
           }

    for (j=0; j<n; j++)
     {
          max[j]=a[0][j];
          for (i=1; i<m; i++)
               if (max[j]<a[i][j])
                    max[j]=a[i][j];
     }

     for (i=0; i<n; i++)
           for (j=i+1; j<n; j++)
                if (max[i] > max[j])
                {
                     t=max[i]; max[i]=max[j]; max[j]=t;
                     for (k=0; k<m; k++)
                     {
                          t=a[k][i];
                          a[k][i]=a[k][j];
                          a[k][j]=t;
                     }
```

```
                    }

      for (j=0; j<n; j++)
      {
           printf("%5f\t",max[j]);
      }

      for (i=0; i<m; i++)
       {
            printf("\n");
            for (j=0; j<n; j++)
                 printf("%5.2f\t",a[i][j]);
       }
}
```