Programiranje 2

Računske vježbe 1

1. Implementirati funkcije koje određuju:

- maksimalni elemenat niza,
- veći od dva cijela broja,
- maksimalni od dva stringa (leksikografski),
- "maksimalni" od dva karaktera,
- za jedan string, karakter sa najvećom ASCII vrijednošću.

```
#include <iostream>
#include <string.h>
using namespace std;

int maks(int *,int);
inline int maks(int,int);
char * maks(char *, char *);
inline char maks(char,char);
char maks(char*);

int main()
{
    int a[]={1,5,4,3,2,6}, n=6;
    cout<<maks(a,n)<<endl;
    cout<<maks(2,5)<<endl;
    cout<<maks("string","program")<<endl;
    cout<<maks('a','A')<<endl;
    cout<<maks("string");
}

int maks(int *a,int n)
{
    int m;
    m=a[0];
    for(int i=1; i<n; i++)
        if(a[i]>m)
            m=a[i];
    return m;
}
```

```
inline int maks(int a,int b)
{
    return (a>=b) ? a:b;
}

char * maks(char *a,char *b)
{
    if(strcmp(a,b)>=0) return a;
    else return b;
}

inline char maks(char a,char b)
{
    return (a>=b) ? a:b;
}

char maks(char *a)
{
    char m;
    m=a[0];
    for(int i=1; i<strlen(a); i++)
        if(a[i]>m)
            m=a[i];
    return m;
}
```

2. Date su deklaracije funkcija:

```
int f(int, char='0');
int f(char, char);
int f(double);
```

Šta će se desiti nakon sljedećih poziva funkcije:

```
R = f('0');
R = f(2.34);
R = f('c', 3);
```

Prvi poziv će izvršiti prvu funkciju. Njen prvi argument je cijeli broj koji se dobija konverzijom iz karaktera (ASCII) dok je drugi argument podrazumijevani karakter tako da će taj poziv biti isto što i f('0','0');

Kod drugog poziva će biti izvršena treća funkcija. Jedino ova funkcija ima realan broj kao argument.

Kod trećeg poziva i prva i druga funkcija mogu da odgovaraju prvom argumentu (bolje odgovara druga funkcija) ali drugi argument ne odgovara ni jednoj funkciji tako da će doći do greške u programu.

3. Dat je niz cjelobrojnih elemenata. Kreirati funkciju koja podrazumijevano obrće elemente niza "naopačke". Ako je kao argument zadat pozitivan cijeli broj, potrebno je formirati novi niz koji predstavlja elemente starog niza od prvog elementa, sa datim preskokom do kraja. Ako je zadat negativan broj, potrebno je formirati novi niz koji ide od posljednjeg elementa ka početku sa odgovarajućim zadatim preskokom.

```
#include <iostream>
using namespace std;

int uredi(int *, int, int *, int=-1);

int main()
{
    int x[10], n, nrez;
    int i, *xrez;

    cout<<"Unijeti duzinu niza: ";
    cin>>n;
    cout<<"Unijeti elemente niza: ";
    for(i=0; i<n; i++)
        cin>>x[i];

    xrez=new(int[n]);

    for(i=0; i<n; i++) cout<<x[i]<<" ";
    cout<<endl;

    nrez = uredi(x, n, xrez);
    for(i=0; i<n; i++) cout<<xrez[i]<<" ";
    cout<<endl;

    nrez = uredi(x, n, xrez, 3);
    for(i=0; i<nrez; i++) cout<<xrez[i]<<" ";
    cout<<endl;

    nrez = uredi(x, n, xrez, -2);
    for(i=0; i<nrez; i++) cout<<xrez[i]<<" ";
    cout<<endl;
}

int uredi(int *x, int n, int *xrez, int pr)
{
    int j=0;
    if(pr>0)
        for(int i=0; i<n; i+=pr) xrez[j++]=x[i];
    else
        for(int i=n-1; i>=0; i+=pr) xrez[j++]=x[i];

    return j;
}
```