18. PROLASCI KROZ NIZ I ISPIS NIZA

Niz je izvedeni tip podatka koji predstavlja strukturu u kojoj su svi elementi istog tipa. Tip niza proizilazi iz tipa iz tipa njegovih elemenata. Indeks niza je promenljiva ili izraz koji se nalazi između zagrada [ ]. Početni element niza ima indeks 0 (nula).

Prolazak kroz jednodimenzionalni niz i ispis elemenata niza se postiže korišćenjem ciklusa: for, while ili do-while. Kontrolna, brojačka promenljiva ciklusa polazi od vrednosti 0 i ciklus se ponavlja dok se ne stigne do poslednjeg elementa niza. Ispis vrednosti se postiže korišćenjem funkcije printf() iz standardne biblioteke za rad sa ulazom i izlazom.

Primer C programa koji ispisuje elemente deklarisanog niza od 5 elemenata sa vrednostima: 10, 20, 30, 40, 50 for ciklusom:

```
#include <stdio.h>
int main()
{
  // deklarisanje i inicijalizacija niza
  int arr[] = {10, 20, 30, 40, 50};
  int i=0;
  // ispis elemenata niza
  for(i; i<5; i++)
  {
     printf("%d ", arr[i]);
  }
  return 0;
}
```

Rezultat izvršavanja:
10 20 30 40 50

Primer C programa koji ispisuje elemente deklarisanog niza od 5 elemenata sa vrednostima: 10, 20, 30, 40, 50 while ciklusom:

```
#include <stdio.h>
int main()
{
  int i=0;
  // deklarisanje i inicijalizacija niza
  int arr[] = {10, 20, 30, 40, 50};
  // ispis elemenata niza
  while(i<5)
  {
     printf("%d ", arr[i]);
     i++;
  }
  return 0;
}
```

Rezultat izvršavanja:
10 20 30 40 50

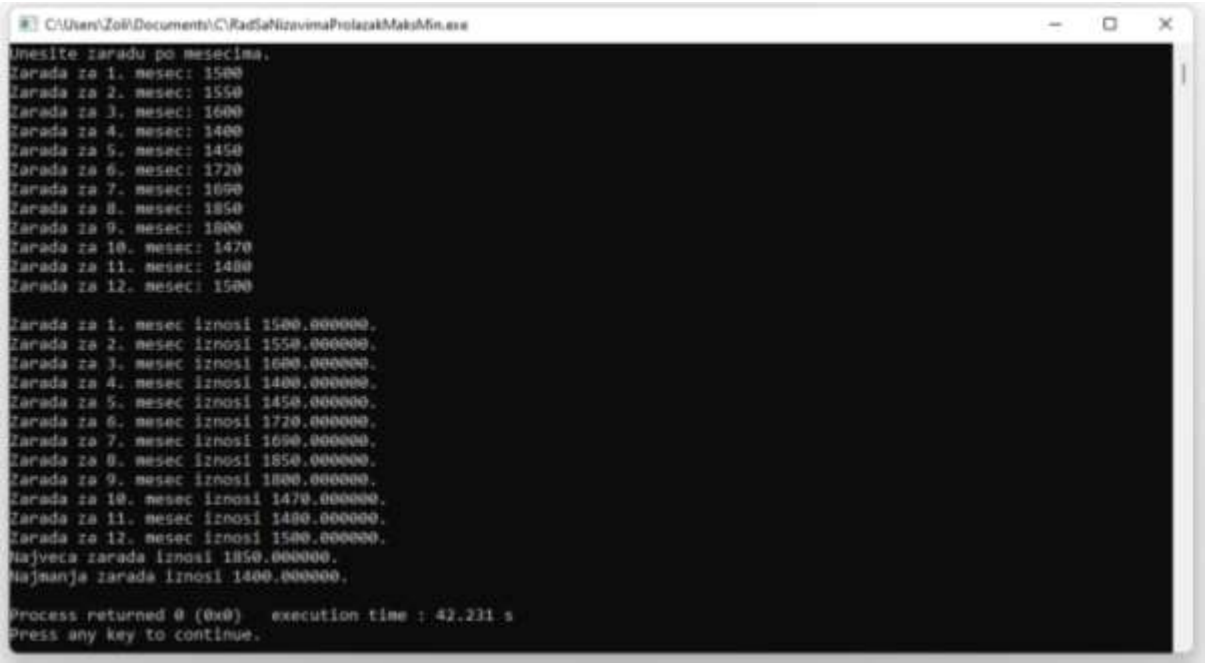
Određivanje najveće i najmanje vrednost niza vrši se prolazcima kroz niz ciklusom. Prvo se deklariše niz, bez inicijalizacije, zatim se for ciklusom dodele vrednosti svim elementima niza. Kada su sve vrednosti niza u memoriji, maksimalna vrednost se određuje tako što se uvodi promenljiva (npr. **maks**) koja će memorisati najveću vrednost u nizu. Ova promenljiva dobija vrednost prvog elementa niza (sa indeksom 0), pa se ciklusom prolazi kroz niz od prvog do poslednjeg elementa i vrši poređenje promenljive sa elementima niza, pa ukoliko je element niza veća vrednost od vrednosti

promenljive, vrši se dodela vrednosti promenljivoj preko vrednosti tekućeg elementa niza. Kada se završi prolazak kroz niz, promenljiva maks će sadržati vrednost najvećeg elementa niza. Određivanje minimalnog elementa niza se radi na isti način kao i za najveći, sa tom razlikom što se uvodi promenljiva za minimum (npr. **min**) i što se prilikom poređenja vrednosti promenljive min i elementa niza, u slučaju da je element niza manja vrednost od vrednosti promenljive, vrši se dodela vrednosti promenljivoj preko vrednosti tekućeg elementa niza.

Primer C programa koji učitava i ispisuje zaradu po mesecima, te pronalazi i ispisuje najmanju i najveću zaradu:

```
/*Program sa zaradama, prolascima kroz niz, minimumom i maksimumom*/
#include <stdio.h>
main()
{
int i;
float zarada[12];
printf("Unesite zaradu po mesecima.\n");
for(i = 0; i < 12; i++)
{
   printf("Zarada za %d. mesec: ", i + 1);
   scanf("%f", &zarada[i]);
}
float maksz=zarada[0];
float minz=zarada[0];
printf("\n");
for(i = 0; i < 12; i++)
{
   printf("Zarada za %d. mesec iznosi %f.\n", i + 1, zarada[i]);
   if (maksz<zarada[i]){maksz=zarada[i];}
   if (minz>zarada[i]){minz=zarada[i];}
}
printf("Najveca zarada iznosi %f.\n", maksz);
printf("Najmanja zarada iznosi %f.\n", minz);
return 0;
}
```

Rezultat izvršavanja:



19. SORTIRANJE I PRETRAŽIVANJE NIZA

Sortiranje predstavlja proces preuređivanja skupa podataka po nekom utvrđenom poretku. Jedna od osnovnih uloga sortiranja je omogućavanje lakšeg pretraživanja.

Selection sort - Osnovna strategija ove metode je da se razlikuju dva dela vektora: sortirani i nesortirani deo. U svakom koraku se sekvencionalni pretraživanjem nesortiranog dela pronađe najmanji element niza, pa on zameni mesto sa prvim elementom. U sledećem koraku se ponovno posmatra neuređeni deo niza (od 2 do n-tog člana) i sekvencionalnom pretragom utvrdi sledeći najmanji element tog dela vektora, pa se on zameni sa drugim elementom. Postupak se ponavlja sve dok se neuređeni, tj. nesortirani deo vektora ne svede samo na jedan element.

Boubble sort - Boubble sort je jedan od najjednostavnijih i najpopularnijih metoda sortiranja. Algoritam više puta sekvencionalno prolazi kroz vektor i upoređuje svaki element sa narednim u vektoru, pa ako ova dva elementa nisu u pravilnom poretku, zamene im se mesta. U svakom prolasku najmanje jedan element dolazi na svoje mesto. Može se desiti da vektor bude sortiran i pre n-1 koraka.

Pretraživanje vektora - Algoritam pretraživanja ima zadatak da utvrdi mesto gde se zadati podatak (vrednost elementa) nalazi u vektoru. Osnovnim metodama pretraživanja se smatraju sekvencionalno pretraživanje kod neuređenih vektora i binarno pretraživanje kod uređenih vektora.

Sekvencionalno pretraživanje vektora - Sekvencionalno pretraživanje je najjednostavnija, ali obično i najmanje efikasna tehnika pretraživanja. Ona podrazumeva da se tražena vrednost uzastopno (sekvencionalno) upoređuje sa po jednim elementom iz neuređenog dela vektora, sve dok se ne dođe do jednakosti ili dok se ne ispitaju svi elementi vektora.

Binarno pretraživanje - Binarno pretraživanje vrši uvek poređenje na polovini intervala. Ovom metodom prvo se upoređuje traženi podatak sa elementom koji se nalazi na srednjoj poziciji vektora. Ako se utvrdi jednakost pretraživanje se završava kao uspešno. Ako je traženi podatak manji (niz je sortiran po rastućem kriterijumu sortiranja), gornja polovina elemenata se odbacuje i postupak nastavlja sa donjom polovinom. Ako je podatak veći postupak se nastavlja sa gornjom polovinom intervala. U svakom koraku broj elemenata vektora sa kojim se vrši poređenje se prepolovi, pa je zbog toga pretraga dobila naziv binarno pretraživanje.

Primer: Učitavanje elemenata niza i **sortiranje u opadajućem redosledu**.

Niz, pre sortiranja: 0 5 7 2 4

Postupak sortiranja preko 2 ugnježdena ciklusa**:**
i=1 j=0 5 0 7 2 4
i=2 j=1 5 7 0 2 4
i=2 j=0 7 5 0 2 4
i=3 j=2 7 5 2 0 4
i=4 j=3 7 5 2 4 0
i=4 j=2 7 5 4 2 0

Niz, posle sortiranja: 7 5 4 2 0

Listing programa:

```
/*C program za sortiranje niza u rastucem poretku*/
#include <stdio.h>
int main()
{
    //definisanje velicine niza
    int n;
    int i;
    printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);
    //deklarisanje niza i unos elemenata niza
    int a[n];
    printf("\nUnesite elemente: \n");
    for(i=0; i<n; i++)
    {
       scanf("%d", &a[i]);
    }
    //zamena mesta susednim elementima niza
    //u slucaju da je naredni veci od prethodnog
    int j;
    int iTemp;
    for(i=1; i<n; i++)
    {
       for(j=i-1; j>=0 && a[j]<a[j+1]; j--)
       {
           iTemp=a[j];
           a[j]=a[j+1];
           a[j+1]=iTemp;
       }
    }
    //ispis elemenata sortiranog niza
    printf("\nIspis niza sortiranog u opadajucem redosledu:\n");
    for(i=0; i<n; i++)
    {
       printf("%d ", a[i]);
    }
    return 0;
}
```

Rezultat izvršavanja:

```
Unesite broj elemenata niza: 8

Unesite elemente:
5 22 365 101 2 8 78 9

Ispis niza sortiranog u opadajucem redosledu:
365 101 78 22 9 8 5 2
```

Sortiranje niza u rastućem poretku: prvo se kreira niz, u primeru, promenljive dužine koja se na početku definiše unosom veličine od strane korisnika. Zatim se kroz ciklus, unose elementi niza celih brojeva koji nije sortiran. Da bi se izvršilo sortiranje formiraju se dva ciklusa, jedan u drugom. U prolasku kroz ciklus, poredi se element sa svakim koji je pre njega u nizu. U slučaju da je tekući element veći od sledećeg, zamene im se mesta, sve dok se ne dođe do kraja niza.

```
/*C program za sortiranje niza u rastucem poretku*/
#include <stdio.h>
void main()
{
    //definisanje velicine niza
    int n, i;
    printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);
    //deklarisanje niza i unos elemenata niza
    int number [n];
    printf("\nUnesite elemente: \n");
    for(i=0; i<n; i++)
    {
       scanf("%d", &number[i]);
    }
    //sortiranje niza
    int j, a;
    for (i = 0; i < n; ++i)
      {
      for (j = i + 1; j < n; ++j)
         {
           if (number[i] > number[j])
             {
                    a =  number[i];
                    number[i] = number[j];
                    number[j] = a;
              }
         }
      }
      //ispis elemenata sortiranog niza
      printf("Elementi niza sortirani u rastucem poretku: \n");
      for (i = 0; i < n; ++i)
         printf("%d\t", number[i]);
 }
```

Rezultat izvršavanja:

```
Unesite broj elemenata niza: 10

Unesite elemente:
123 105 1 4 8 3 5 22 754 300

Elementi niza sortirani u rastucem poretku:
1       3       4       5       8       22      105     123     300     754
```

Pretraživanje niza - prvo se kreira niz, u primeru, promenljive dužine koja se na početku definiše unosom veličine od strane korisnika. Zatim se kroz ciklus, unose elementi niza celih brojeva. Sledi unos podatka, tj. vrednosti koja će se pretraživati. Pretraga se radi kroz for ciklus. U prolasku kroz ciklus, poredi se traženi podataka sa svakim elementom niza. U slučaju da je tekući element veći jednak traženom, ispisuje se poruka, pozicija na kojoj je pronađena vrednost, te se završava funkcija. U slučaju prolaska kroz ceo niz i nepronalaska tražene vrednosti, ispisuje se poruka da podataka nije pronađen.

```
#include<stdio.h>
int main()
{
    //definisanje velicine niza
     int n, i;
     printf("Unesite broj elemenata niza: ");
     scanf("%d", &n);
     //deklarisanje niza i unos elemenata niza
     int a[n], element, pos=0;
     printf("Unesite elemente niza: ");
     for(i=0; i<n; i++)scanf("%d", &a[i]);
     printf("Unesite vrednost za pretragu: ");
     scanf("%d",&element);
     for(i=0; i<n; i++)
     {
       if(a[i]==element)
       {
          printf("%d pronadjen na poziciji %d", element, i+1);
          return 0;
       }
     }
     printf("%d nije pronadjen.", element);
     return 0;
}
```

Rezultat izvršavanja:

```
Unesite broj elemenata niza: 7
Unesite elemente niza: 45 65 85 35 65 75 47
Unesite vrednost za pretragu: 75
75 pronadjen na poziciji 6

Unesite broj elemenata niza: 5
Unesite elemente niza:  1 2 4 8 9
Unesite vrednost za pretragu: 45
45 nije pronadjen
```

20. ROTIRANJE I SAŽIMANJE NIZA

Rotiranje niza - prvo se definiše, tj. kreira niz određene dužine i izvrši inicijalizacija niza. Zatim se unosi pozicija elementa **N** u nizu od kog će početi rotacija elemenata. Uz pomoć for ciklusa se vrši pomeranje elementa niza za jedno mesto tačno onoliko puta kolika je vrednost pozicije **N**. Nakon završetka svih iteracija for ciklusa, niz će biti rotiran za navedeni broj elemenata.

```
#include <stdio.h>
int main()
{
  // deklarisanje i inicijalizacija niza
  int niz[] = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100};
  int i=0;
  int duzina=10;
  int n=3; //pozicija za rotaciju
  int j;
  // ispis elemenata niza
  for(i; i<duzina; i++)
  {
     printf("%d ", niz[i]);
  }
  int temp=0;
  // rotacija
  for(i = 0; i < n; i++)
    {
        temp = niz[duzina - 1];
        for(j = duzina - 1; j > 0; j--)
        {
            niz[j] = niz[j - 1];
        }
        niz[j] = temp;
    }
  // ispis elemenata rotiranog niza
  printf("\nRotirani niz:\n");
  for (i=0; i<duzina; i++)
   {
      printf("%d ", niz[i]);
    }
  return 0;
}
```

Rezultat izvršavanja:

```
Za N=3
10 20 30 40 50 60 70 80 90 100
Rotirani niz:
80 90 100 10 20 30 40 50 60 70
Za N=1
10 20 30 40 50 60 70 80 90 100
Rotirani niz:
100 10 20 30 40 50 60 70 80 90
Za N=6
10 20 30 40 50 60 70 80 90 100
Rotirani niz:
50 60 70 80 90 100 10 20 30 40
```

Uklanjanje elementa niza (sažimanje) - prvo se deklariše niz određene dužine i izvrši unos elemenata niza. Zatim se unosi pozicija elementa **P** u nizu, od kog će se izvršiti sažimanje niza. Uz pomoć for ciklusa se vrši pomeranje elementa niza za jedno mesto od pozicije P-tog elementa do kraja niza. Nakon završetka svih iteracija for ciklusa, niz će biti smanjen za jedan element.

```
#include <stdio.h>
int main()
{
   //definisanje velicine niza
   int n, i;
   printf("Unesite broj elemenata niza: ");
   scanf("%d", &n);
   //deklarisanje niza i unos elemenata niza
   int a[n], element, pos=0;
   printf("Unesite elemente niza: ");
   for(i=0; i<n; i++)scanf("%d", &a[i]);
   //unos lokacije za sazimanje
   int p, c;
   printf("Unesite lokaciju u nizu sa koje zelite da obrisete element:
\n");
   scanf("%d", &p);
   if (p>=n+1)
      printf("Brisanje nije moguce.\n");
   else
   {
      //sazimanje
      for (c=p-1; c<n-1; c++)
         a[c]=a[c+1];
      printf("Rezultujuci sazeti niz:\n");
      for (c=0; c<n-1; c++)
         printf("%d\n", a[c]);
   }
   return 0;
}
```

Rezultat izvršavanja 1:

```
Unesite broj elemenata niza: 8
Unesite elemente niza: 1 2 3 4 5 6 7 8
Unesite lokaciju u nizu sa koje zelite da obrisete element:
3
Rezultujuci sazeti niz:
1
2
4
5
6
7
8
```

Rezultat izvršavanja 2:

```
Unesite broj elemenata niza: 5
Unesite elemente niza: 1 2 3 4 5
Unesite lokaciju u nizu sa koje zelite da obrisete element:
8
Brisanje nije moguce.
```