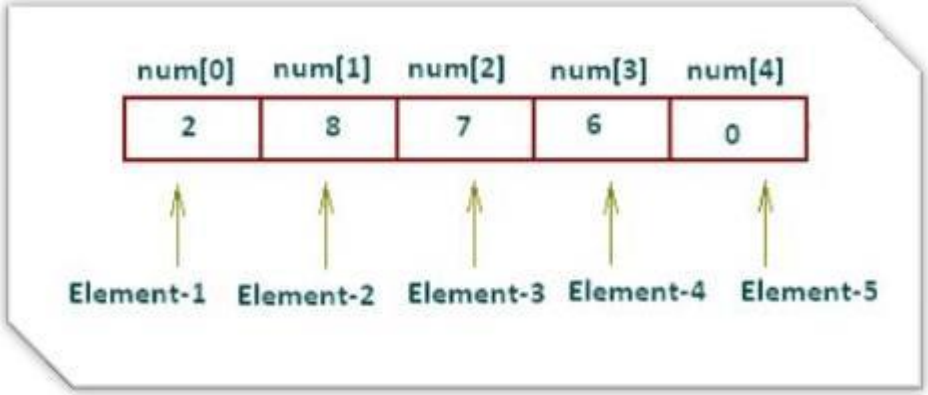
Nizovi

predavač: Nadežda Jakšić

Programiranje programski jezik C++

Jednodimenzionalni niz

- jednom komandom definišemo više promenljivih
- najjednostavniji i najčešće korišćen složeni tip podatka
- niz podataka istog tipa koji se u memoriji smeštaju jedan iza drugog
- nazivaju se elementima niza i identifikuju se pomoću rednog broja (indeksa)
- prvi element niza ima indeks nula
- ako niz ima n elemenata, poslednji element niza ima indeks n-1



Jednodimenzionalni niz

- deklarisanje
  tip identifakator [n]; //n je broj elemenata niza
- elementi niza mogu da pripadaju nekom od elementarnih tipova (char, int, float, double), a mogu da budu i stringovi, objekti i drugi nizovi
- funkcije nije dozvoljeno direktno koristiti kao elemente, ali se mogu predstaviti deklarisanjem pokazivača na funkciju koji će dobiti element niza
- elementima niza se pristupa preko indeksa

Inicijalizacija i unos

- inicijalizacija podrazumeva postavljanje početnih vrednosti

```
int dani[12];
dani[0]=31;
dani[1]=28;
 …
dani[11]=31;
```

ili

```
int dani[12]={31,28,31,30,31,30,31,31,30,31,30,31};
```

ili unos vrednosti preko tastature

```
int dani[12];
for (int i=0;i<12;i++)
{
      cout<<"Unesite broj dana " <<endl;
      cin>>dani[i];
}
```

Inicijalizacija i unos

- deklarisati niz od 5 elemenata i inicijalizovati elemente

```
int visina [12];
visina [0]=189;
visina [1]=172;
visina [2]=193;
visina [3]=180;
visina [4]=179;
```

- deklarisati niz od 5 elemenata i uneti vrednosti elemenata preko tastature

```
int visina [5];
for (int i=0;i<5;i++)
{
      cout<<"Unesite visinu ucenika " <<endl;
      cin>>visina[i];}
```

Inicijalizacija i unos

- dužina deklarisanog niza se ne može menjati u toku izvršavanja programa, a mora biti veća ili jednaka broju članova niza - ako je deklarisana dužina niza veća od broja članova niza kojima je pridružena vrednost, preostalim članovima se pridružuje podrazumevana vrednost (za int, float i double nula, za pokazivač null, za karakter vrednost \0)

```
int main ()
{
int A[6]={2,34,1,67,99,7};
for (int i=0;i<6;i++){
cout<<"A["<<i<<"]="<<A[i]<<"    ";
}
return 0;}
```

A[0]=2 A[1]=34 A[2]=1 A[3]=67 A[4]=99 A[5]=7

```
int A[6]={2,34,1,67};
for (int i=0;i<6;i++)
 {
 cout<<"A["<<i<<"]="<<A[i]<<"    ";
 }
```

A[0]=2 A[1]=34 A[2]=1 A[3]=67 A[4]=0 A[5]=0

Inicijalizacija i unos

ako se niz samo deklariše u okviru main funkcije i prikažu se vrednosti njegovih elemenata, dobiće se slučajne vrednosti u memoriji

```
int main()
{
	int brojevi[10];
	for (int i = 0; i < 10; i++)
	{
		cout << brojevi[i] << endl;
	}
	cout << endl;
	return 0;}
```

ako se niz deklariše kao globalna promenljiva, izvan funkcije main, inicijalne vrednosti će biti nule

```
#include <iostream>
using namespace std;
int brojevi[10];
int main()
{
	for (int i = 0; i < 10; i++)
	{
		cout << brojevi[i] << endl;
	}
	cout << endl;
	return 0;}
```

Primer

// uneti n članova niza, pa ih ispisati i sabrati (n<=10)

```
int main ()
{int n,i,zbir=0,X[10];
do
{cout<<"Upisi broj clanova niza:";
 cin>>n;
}
while ((n<=0)||(n>10));
for (i=0;i<n;i++)
{
cout<<endl<<"Unesi X["<<i<<"]:";
cin>>X[i];
}
```

```
cout<<"Uneli ste:"<<endl;
for (i=0;i<n;i++)
{
cout<<X[i]<<endl;
zbir+=X[i];
}
cout<<endl<<"n="<<n<<". Zbir "<<n
<<"clanova niza iznosi "<<zbir <<endl;
return 0;
}
```

Granice niza

- C++ ne proverava granice niza prilikom pristupa članovima niza
- ako se navede prevelik ili negativan indeks, kompajler neće javiti grešku i pri izvršavanju programa pristupiće memorijskoj adresi koja nije u području koje je rezervisano za niz
- to će prouzrokovati neispravan rad programa
- ako se pristupa članu sa nedozvoljenim indeksom, rezultat je slučajan broj
- ako se vrednost pridružuje elementu sa nedozvoljenim indeksom, vrednost će biti sačuvana u memoriji na mestu koje je predviđeno za druge sadržaje, što će najverovatnije prouzrokovati pogrešan rad programa

Primer

// korisnik unosi dužinu niza (najviše 10 elemenata), učitavaju se celi brojevi u niz, računa se i prikazuje zbir unetih brojeva – unos i računanje se ponavlja sve dok korisnik kao dužinu niza ne unese broj koji je manji od nule ili veći od 10 – tada se prekida se unos nizova

```
int main ()
{
const int DUZ = 10;
while (true)
{
    cout << "\nDuzina niza? ";
    int n;
    cin >> n;
    if (n <= 0 || n > DUZ) break;
    int a[DUZ]; //deklariše se niz dužine DUZ
    cout << "Elementi niza? ";
```

```
for (int i=0; i<n; cin >> a[i++]) ;
 int s = 0;
 for (int i=0; i<n; s+=a[i++]) ;
 cout << "Zbir elemenata: " << s << endl;
}
return 0;
}
```

Najmanji element niza

```
int n;
cout << "Unesite broj clanova niza:";
cin >> n;
int a[n];
cout << "Unesite clanove niza:"<<endl;
for (int i=0; i< n; i++){
    cin >> a[i];
}
cout << "Ispis" << endl;
int tmax=a[0];
for (int i=1; i< n; i++){
    if (tmax < a[i])
    {  tmax=a[i]; }
}
cout<<"Najveci element niza je : "<<tmax<<endl;
```

Prebrojavanje

- učitati niz i prikazati koliko je unetih brojeva deljivo sa tri

```
int n, br=0;
cout << "Unesite broj clanova niza:";
cin >> n;
int a[n];
cout << "Unesite clanove niza:"<<endl;
for (int i=0; i< n; i++){
    cin >> a[i];
    if (a[i] %3==0) br++;
    }
cout << br << "unetih elemenata je deljivo sa 3"<<endl;
```

Prebrojavanje

- učitati niz X od n članova - naći najmanji od članova sa parnim indeksima

```
int n;
cout << "Unesite broj clanova niza:";
cin >> n;
int a[n];
cout << "Unesite clanove niza:"<<endl;
for (int i=0; i< n; i++){
      cout<<"a["<<i<<"]-->";
      cin >> a[i];    }
cout << "Ispis" << endl;
int tmin=a[2];
```

```
for (int i=4; i< n; i++){
   if((tmin>a[i]) && ( i % 2 == 0))
        {
          tmin=a[i];
        }
      }
cout<<"Najmanji od clanova sa
parnim indeksom je :
"<<tmin<<endl;
```

Niz kao argument funkcije

```
//funkcija vraća aritmetičku sredinu elemenata niza
float nadjiProsek (int n, float niz[])
   {int i;  float s=0;
            for (i=0;i<n;i++)
            s+=niz[i];
return (s/n);}
int main ()
{ int i,n;f loat niz[10];
            cout<<"Koliko elemenata ima niz? " <<endl;
            cin>>n;
            for (i=0;i<n;i++)
                        {cout<< " Unesite " << i+1<< ". element niza " <<endl;
                         cin>>niz[i];}
cout<<"Aritmeticka sredina unetih vrednosti je " << nadjiProsek (n,niz);
return 0;}
```

Primeri

- učitati niz od n brojeva i pomnožiti ih
- učitati niz, svaki element niza uvećati za 5 i prikazati novi niz
- prikazuje se svaki drugi element niza počevši od poslednjeg
- napisati funkcije za unos elemenata u niz i za prikazivanje niza
- pronaći indeks najmanjeg elementa niza
- od učitanih nizova a i b formira se niz c tako što se sabiraju parovi a i b, ako je indeks niza paran broj, a množe se parovi a i b , ako je indeks niza neparan broj

Primeri

- meteorološka stanica je za n dana posmatranja formirala tablicu vrednosti atmosferskog pritiska, temperature i vlažnosti vazduha; napisti program koji:

  učitava vrednosti za pritisak, temperaturu i vlažnosti vazduha

  određuje maksimalne i minimalne vrednosti za sve tri veličine koje se posmatraju, kao i redni broj odgovarajućeg dana

  određuje srednju vrednost u periodu posmatranja za sve tri veličine

napisati funkcije koje određuju minimalnu, maksimalnu i srednju vrednost niza

Pretraživanje nizova

pretraživanje nizova – pronalaženje elementa niza, liste, stabla ili neke druge strukture podataka koji zadovoljava unapred definisani kriterijum
linerano pretraživanje – uzastopno upoređivanje elemenata niza sa traženom vrednošću; jednostavan algoritam ali prilično spor

- primer: niz a je dužine n, funkcija traži indeks traženog elementa t unutar niza, ako ga ne pronađe vraća -1

```
int trazi (int a[], int n, int t)
{
int i;
for (i=0;i<n;i++)
        if (a[i]==t) return i;
        return (-1);}
```

Linerano pretraživanje

```
include <stdio.h>
int linearnaPretraga (int niz[], int brElem, int x)
{
int i;
for (i = 0; i<brElem; i++)
if (niz[i] == x) return i;
return -1;
}
int main()
{
int a[] = {4, 3, 2, 6, 7, 9, 11};
int x,i;
cout<<"Unesite broj koji trazite : " <<endl;
cin>>x;
i = linearnaPretraga (a,7, x);
```

```
if (i == -1)
cout<<"Element  “<<x<< " nije nadjen";
else
cout<< " Indeks elementa "<<x<< " je "
<<i<< " a redni broj u nizu je " <<i+1;
return 0;}
//korisnik unosi elemente niza
```

Binarna pretraga

binarna pretraga – niz mora da bude uređen u rastućem ili opadajućem poretku; poredi se traženi element sa središnjim elementom niza; ako nije jednak, određuje se u kojoj polovini se može naći traženi element, a druga polovina se odbacuje; proces se ponavlja

- primer: binarno pretraživanje u rastućem nizu

```
int trazi (int a[]), int n, int t) //n je dužina niza, t je tražena vrednost
{ int levi, desni, srednji;
 levi=0;
 desni=n-1;
 while (levi<= desni)
{ srednji=(levi+desni)/2;
            if (a[srednji] == t) return (srednji);
     if (t<a[srednji]) desni=srednji-1;
            else levi=srednji+1;
} return(-1); }
```

binarna pretraga je brža od linearne
mana – ne može da se pretražuje niz koji nije sortiran

Binarna pretraga - primer

```
#include <stdio.h> //niz je uređen u neopadajućem redosledu
int binarnaPretraga (int a[], int n, int x) //n je dužina niza, tražimo x
{int levi = 0;
int desni = n-1;
while (levi <= desni)
   {
    int sr = (levi+desni)/2;
    if (x == a[sr])
       return sr;
    else
       if (x < a[sr])
           desni = sr-1;
       else  levi = sr+1;
 }
return -1;}
```

```
int main()
{
 int a[] = {3, 5, 7, 9, 11, 13, 15};
 int x;
 int i;
 cout<<"Unesi element koji trazimo : "<<endl;
 cin>>x;
 i = binarnaPretraga (a, 7, x);
 if (i==-1)
        cout<< " Ne postoji element " <<x;
 else
        cout<<"Pronadjen na poziciji " << i+1;
return 0;}
```

Binarna pretraga - rekurzivno

```
int binarnaPretraga (int a[], int levi, int desni, int x)
{
int sr;
if (levi > desni)
  return -1;
sr=levi+(desni-levi)/2;
if (x>a[sr])
return binarnaPretraga (a,sr+1,desni,x);
else if (x<a[sr])
  return binarnaPretraga (a,levi,sr-1,x);
else
  return sr;
}
```

```
int main()
{
int a[] = {3, 5, 7, 9, 11, 13, 15};
int x,i,levi=1,desni=7;
cout<<"Koji element se trazi: "<<endl;
cin>>x;
i = binarnaPretraga (a,levi,desni,x);
if (i == -1)
cout<<"Ne postoji element "<<x;
else
cout<<"Na poziciji  "<< i+1;
return 0;
}
```

Sortiranje nizova

selection sort

- u prvoj iteraciji poredi se prvi element niza sa ostalim članovima niza i ako se naiđe na element koji je manji od prvog, zamene se mesta (na kraju iteracije na prvom mestu se nalazi najmanji element niza)
- u drugoj iteraciji poredi se drugi element sa ostalim elementima niza...u poslednjoj iteraciji se porede pretposlednji i poslednji element niza

```
int niz[10], i, j, temp;
for (i=0;i<9;i++) //od prvog do pretposlednjeg
for (j=1;j<10;j++) //od drugog do poslednjeg
        if (niz[i]<niz[j])
                {temp=n[i]; //zamena mesta preko promenljive temp
                n[i]=n[j];
                n[j]=temp;}
```

efikasniji algoritmi za sortiranje su quick sort, bubble sort i merge sort

Sortiranje nizova - primer

```
//sortiranje u nerastućem poretku a[0]>=a[1]>=...a[n-1]
/#include <iostream>
#include <cstdlib> //zbog exit
using namespace std;
#define MAXDUZ 100
int main()
{int a[MAXDUZ];
int n,pom,i,j;
cout<<"Unesite dimenziju niza"<<endl;
cin>>n;
if (n>MAXDUZ)
   {cout<<"Nedozvoljena duzina niza";
   exit(1);}
for (i=0; i<n; i++)
   {cout<<"Unesite "<<i+1<<". clan niza"<<endl;
   cin>>a[i];}
```

```
for (i=0; i<n-1; i++)
   for (j=i+1; j<n; j++)
      if (a[i]<a[j])
        {
        pom=a[i];
        a[i]=a[j];
        a[j]=pom;
        }
cout<<"Sortirani niz:"<<endl;
    for (i=0; i<n; i++)
    cout<<a[i]<<" ";
    cout<<endl;
return 0;}
```

Funkcija sort

- biblioteka STL (standard template library) - heder "algorithm" omogućava upotrebu sortiranja, kao i mnogih drugih algoritama
- najjednostavniji oblik: funkcija sort ima dva argumenta – prvi je adresa početnog elementa niza, a drugi je adresa iza poslednjeg elementa; na primer, ako niz ima n elemenata, treba pisati sort (a, a+n);
- sortiranje u nerastući (opadajući) poredak: za celobrojni niz treba pisati sort (a, a+n, greater<int>()); - ako su elementi niza drugog tipa, njihov tip treba da stoji umesto reči int
- ako je potreban neki specijalan poredak, može se napisati sopstvena funkcija za poređenje elemenata niza, takozvanim komparatorom - fukcija kao argumente prima dve vrednosti (istog tipa kao što su elementi niza), a vraća vrednost tipa bool tj.vraća true ako prvi argument u uređenom nizu treba da se nalazi pre drugog, a false u suprotnom - napisanu funkciju navodimo kao treći argument pri pozivu algoritma za sortiranje

Funkcija sort - primer

```
#include <iostream> // input/output
#include <algorithm> // sort
#include <functional> // greater
using namespace std;
const int n = 8;
int a[n] = {5, 3, 4, 1, 3, 7, 2, 2};
void Output () //ispis elemenata
{for (int i = 0; i < n; i++)
cout << a[i] << " ";
cout << endl;}
bool Order1(int a, int b)
{ return a > b; }
bool Order2(int a, int b)
{int ra = a % 2;
 int rb = b % 2;
if (ra != rb) return ra < rb; // ako su brojevi različite parnosti, paran ide pre
else return a < b; } // ako su iste parnosti, manji ide pre
```

```
int main()
{
sort(a, a+n); // rastući poredak
Output();
// opadajući poredak
 sort (a, a+n, greater<int>());
Output();
 // opadajući poredak sa svojim komparatorom
sort (a, a+n, Order1);
Output();
// svi parni u rastućem poretku, pa svi neparni u rastućem poretku
sort (a, a+n, Order2);
Output(); return 0;}
```

```
1 2 2 3 3 4 5 7
7 5 4 3 3 2 2 1
7 5 4 3 3 2 2 1
2 2 4 1 3 3 5 7
```

Dvodimenzionalni nizovi

- niz čiji su elementi jednodimenzionalni nizovi – članovima se pristupa preko dva indeksa - ako zamislimo dvodimenzionalni niz kao tabelu, onda prvi indeks određuje red, a drugi indeks određuje kolonu - prvi član niza ima indeks [0][0] a posljednji član ima indeks [(broj redova-1)][(broj kolona-1)]
- deklarisanje

tip identifakator [n] [m];

matrica nxm, nRedova x mKolona

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}

- inicijalizacija

int Matrica [3][4]={{0,10,8,0},{4,7,8,2},{56,12,3,9,}}

3 jednodimenzionalna niza sa po 4 elementa

0	10	8	0
4	7	8	2
56	12	3	9

- unos elemenata matrice

```
int a[3][4], i,j;
for (i=0;i<3;i++) //redovi od 0 do 2
for (j=0;j<4;j++) //kolone od 0 do 3
     cin>>a[i][j];
```

Dvodimenzionalni nizovi

//uneti članove dvodimenzionalnog niza pa ih ispisati u obliku tabele
int X[2][3]= {{1,2,3}, {4,5,6}};
ispis je oblika:
1 2 3
4 5 6

```
int i,j;
int X[2][3]= { {1,2,3},{4,5,6}};
for (i=0;i<2;i++)
{
   for (j=0;j<3;j++)
      {
       cout<<X[i][j]<<" ";
      }
   cout<<endl; }
```

ako hoćemo da se pri ispisu pojave i indeksi elemenata onda:

```
for(i=0;i<2;i++)
{
for(j=0;j<3;j++)
{
cout<<"X["<<i<<"]["<<j<<"]= "<<X[i][j]<<"  ";
}
cout<<endl;
}
```

Kvadratna matrica

- matrica koja ima isti broj redova i kolona (a[i][j] tj. a[4][4])

pojmovi

$$\begin{matrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{matrix}$$

- glavna dijagonala
  ako je i = j onda je a[i][j] na glavnoj dijagonali
- sporedna dijagonala
  i+j = n-1
- elementi iznad glavne dijagonale
  ako je i<j onda je a[i][j] iznad glavne dijagonale
- elementi ispod glavne dijagonale
  ako je i>j onda je a[i][j] ispod glavne dijagonale

Primeri

- učitati matricu veličine nxm (sa tastature), a zatim prikazati novu matricu čiji su elementi za jedan veći od elemenata početne matrice

- učitati kvadratnu matricu (nxn), a zatim prikazati:
  - matricu u obliku tabele
  - broj neparnih elemenata ispod glavne dijagonale
  - najveći element na glavnoj dijagonali
  - najmanji element na sporednoj dijagonali
  - broj nula iznad glavne dijagonale
  - zbir svih parnih elemenata matrice

Primeri

- matricom dimenzije n data je tabela jesenjeg dela fudbalskog šampionata, čiji su elementi:

  0 ako je ekipa i izgubila od ekipe j

  1 ako je rezultat nerešen

  2 ako je ekipa i pobedila ekipu j

  napisati program kojim se izračunava:

  broj ekipa koje su imale više pobeda nego poraza

  broj ekipa koje su prošle prvenstvo bez poraza

  sadržaj na glavnoj dijagonali zanemariti