Класа string

Функции за работа со текстуални низи

Постојат 7 основни функции кои овозможуваат работа со текстуални низи (сите дефинирани во датотеката "<cstring>"):

- **strlen(char[] niza)** - ја враќа должината на низата niza (без null знак)
- **strcpy(destinacija, izvor)** - копира текстуална низа (од izvor во destinacija), вклучувајќи го и null знакот. Внимавајте: низата destinacija треба да има доволна големина за да ги собере сите знаци од izvor.
- **strncpy(destinacija, izvor, N**) - копира најмногу N знаци од текстуална низа (од izvor во destinacija). Знакот '\0' ќе се ископира само доколку се појави во првите N знаци - инаку, истиот мора да го додадеме самите. Внимавајте: низата destinacija треба да има доволна големина за да ги собере сите потребни знаци.
- **strcmp(prva, vtora)** - споредува две низи (резултатот е 0 ако низите се еднакви, инаку е 1 и променливата во која се сместува резултатот се деклрир како логичка променива)
- **strncmp(prva, vtora, int N)** - споредува N знаци (или помалку, ако '\0' се појави претходно) од низите prva и vtora (резултатот е 0 ако првите N знаци од низите се еднакви)
- **strcat(prva, vtora**) - ја надоврзува низата vtora на prva. Внимавајте: низата prva треба да има доволна големина за да ги соберете знаците од двете низи.
- **strncat(prva, vtora, N)** - ги надоврзува првите N знаци (или помалку, ако '\0' се појави претходно) од низата vtora на prva. Внимавајте: низата prva треба да има доволна големина за да ги собере сите потребни знаци.

Основни функции кои овозможуваат работа со string (сите дефинирани во датотеката "<string>"):

- споредување на стрингови – **sting1.compare(sting2); -** враќа 0 ако стринговите се еднакви, а ако не се 1
- спојување на стрингови – **string1.append(string2) или со знак „+“;**
- издвојување на подстринг – **string1.substr(poz,br_karakteri);**
- заменување на стрингови – **string1.swap(string2);**
- наоѓање на подстринг – **string1.find(string2) и string1.rfind(string2) ;**
- замена на карактери во стринг – **string1.replace(poz,br_karakteri, string2);**

- вметнување на еден стринг во друг – **string1.insert(poz,string2);**
- должина на стринг – **string1.length();**

Функции за работа со знаци:

- **isalpha(ch)** – true ако ch е алфабетски знак инаку false
- **isupper(ch)-** true ако ch е голема буква инаку false
- **islower(ch)-** true ако ch е мала буква инаку false
- **isdigit(ch)-** true ако ch е цифра од 0 до 9 инаку false
- **toupper(ch), tolower(ch)** – претвораат во мала т.е. во голема буква

Стандардната библиотека на шаблони нуди посебен контејнер (string) за работа со текстуални податоци. string всушност дава еден вид проширување на операциите кои се нудат од страна на vector<char>. Како таков контејнер, string ги нуди повеќето методи кои ги нуди и контејнерот вектор - size, empty, push_back, begin, end, resize, clear, insert и erase. Дополнително, string (како и вектор) овозможува пристапување до елемент на произволна позиција - преку операторот '[p]' (во овој случај, тоа се знаците во текстот). За разлика од vector<char>, string користи посебен механизам за управување со меморија и нуди неколку методи за манипулирање со текстуални податоци.

Функцијата **str.size()** враќа како резултат должина на стрингот str изразена во бајти.

Функцијата **str.length()** враќа како резултат должина на стрингот str изразена во бајти.

Пр. програма со која се печати должината на внесен стринг

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
  string str, str1;
  getline(cin,str);
  cout<<"Goleminata na stringot str iznesuva "<<str.size()<<" znaci"<<endl;
  getline(cin, str1);
  cout<<"Goleminata na stringot str iznesuva "<<str1.length()<<" znaci";
return 0;
}
```

Пр.

```
#include<iostream>
#include<string>
```

```
using namespace std;
int main()
{
  char a[5]="abc";
   cout<<"goleminata na nizata a e "<<sizeof(a);
   cout<<endl;
   string b,s;
   s=a;
   cout<<"goleminata na stringot s e ";
   cout<<s.length();
   cout<<endl;
   cout<<"goleminata na stringot s e ";
   cout<<s.size();
  cout<<endl;
    b="abcd";
   cout<<"Goleminata na stringot b e : "<<b.length();
   return 0;
}
```

Спојување на два string-a.

Првиот начин на спојување е со помош на **операторот '+'**: имено, два string-a str1 и str2 може да ги споиме во нов string со нивно едноставно "собирање" - str1+str2, или, доколку е тоа потребно, str1+=str2.

Вториот начин на спојување го користи методот **append(vtor): prv.append(vtor)** ќе ги спои string-овите prv и vtor и ќе го смести резултатот во променливата prv.

За користење на string, потребно е вклучување на соодветната датотека која ја содржи неговата дефиниција (#include <string>).

Пр. програма за спојување на стрингови

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
string a = "pocetok";
string b = " i ";
string c = "kraj";
cout << a.size() << endl;                    //pechati '7'
for (int i=0; i<c.size(); i++)
cout << c[i] << " ";                         //pechati 'k r a j ';
cout << endl;
a += b;                                              //a='pocetok i '
a.append(c);                                 //a='pocetok i kraj'
a.push_back('!');                            //a='pocetok i kraj!'
string res = b + c;                          //res=' i kraj'
cout << res << endl;                         //pechati ' i kraj'
```

```
return 0;
}
```

Контејнерот string нуди метод substr преку кој може да вратиме одреден дел од почетниот string. Притоа, доколку го повикаме методот преку наведување на само еден аргумент од тип int,

на пример **str.substr(x),**

тој ќе врати дел од оригиналниот string str кој почнува од позицијата x и завршува со завршување на string-от str.

Ако наведеме два аргумента од тип int,

на пример **str.substring(x, len),**

операцијата ќе врати дел од оригиналниот string str кој почнува од позиција x и има големина најмногу len (доколку од x до крајот на оригиналниот string има помалку од len знаци, се враќаат само онолку знаци колку што има до крајот на string-от).

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
string str = "alo zdravo kako si";
cout << str.substr(0, 3) << endl;                 //pechati 'alo'
cout << str.substr(0, 5) << endl;                 //pechati 'alo z'
cout << str.substr(12) << endl;                   //pechati 'ako si'
cout << str.substr(12, 100) << endl;              //pechati 'ako si'
return 0;
}
```

Контејнерот string може едноставно да се претвори во стандардна текстуална низа (C тип на текстуален податок - char*) со повикување на методот c_str(). На пример, следниве две линии претвораат string во низа од char знаци:

```
char cstr[str.size()+1];
strcpy (cstr, str.c_str());
```

Пр. програма со која се внесуваат два стринга а потоа тие се претвораат во низа знаци

```
#include <iostream>
#include <cstring>
using namespace std;
int main ()
{
  string str, str1;
```

```
  int k,i,k1;
  getline(cin,str);
  getline(cin,str1);
  k=str.size();
  k1=str1.size();
  cout<<"Dol na prvata niza e"<<k<<endl;
  cout<<"dol na vtorata niza e "<<k1<<endl;
  char s[k],s1[k1];
    strcpy (s,str.c_str());
    strcpy (s1,str1.c_str());
    cout<<"Nizite znaci se"<<endl;
    for(i=0;i<k;i++)
        cout<<s[i];
    cout<<endl;
    for(i=0;i<k1;i++)
        cout<<s1[i];
    cout<<endl;
return 0;
}
```

Пр. Програма што ќе избори и печати колку мали самогласки има во една реченица внесена од тастатура

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
char tekst[100];
bool palindrom;
int b=0,dolzina;
cout << "Vnesi tekst: ";
cin.getline(tekst,100);
dolzina=strlen(tekst);

for(int i=dolzina-1;i>=0;i--)
   {
     if (tekst[i]=='a' || tekst[i]=='o' || tekst[i]=='e' || tekst[i]=='i' ||tekst[i]=='u')
         b++;
             }
           cout<<"Vneseniot tekst ima "<<b<< "samoglaski"<<endl;
return 0;
}
```

Пр. Програма со која сите букви во дадена низа знаци се претвораат во мали.

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
```

```
{
   char a[100];
   int d,i;
cout<<"Vnesi niza "<<endl;
cin.getline(a,100);
d=strlen(a);
for(i=0;i<=d-1;i++)
   a[i]=tolower(a[i]);
for(i=0;i<=d-1;i++)
   cout<<a[i];
return 0;
}
```

Пр. програма со која се определува колку знаци во дадена низа знаци се букви.

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   char a[100];
   int d,i,b=0;
cout<<"Vnesi niza "<<endl;
cin.getline(a,100);
d=strlen(a);
for(i=0;i<=d-1;i++)
   if (isalpha(a[i])) b++;

   cout<<"ima "<<b;
return 0;
}
```

Пр. Програма со која се проверува дали два внесени текста се еднакви или не.

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
 char tekst1[100],tekst[100];
 int i,dolzina;
 bool t;
 cout << "Vnesi tekst: ";
 cin.getline(tekst,100);
cin.getline(tekst1,100);
  t=strcmp(tekst, tekst1);
  if(t==0) cout<<"Tekstovite se isti";
  else cout<<"tekstovite ne se isti";
 return 0;
}
```

Пр. Програма што ќе избори и печати колку мали самогласки има во една реченица внесена од тастатура

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
char tekst[100];
bool palindrom;
int b=0,dolzina;
cout << "Vnesi tekst: ";
cin.getline(tekst,100);
dolzina=strlen(tekst);

for(int i=dolzina-1;i>=0;i--)
  {
   if (tekst[i]=='a' || tekst[i]=='o' || tekst[i]=='e' || tekst[i]=='i' ||tekst[i]=='u')
      b++;
         }
        cout<<"Vneseniot tekst ima "<<b<< "samoglaski"<<endl;
return 0;
}
```

Пр. Програма со која се проверува дали два внесени текста се еднакви или не.

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
        string st1,st2;
        int rezultat;
        cout<<"Vnesete prv tekst"<<endl;
        getline(cin,st1);
        cout<<"Vnesete vtor tekst"<<endl;
        getline(cin,st2);
        rezultat=st1.compare(st2);
        иf (rezultat==0) cout<<"da";
        else cout<<"ne";
        return 0;
}
```

Пример : споредува два стринга, ако се еднакви враќа 0, ако се различни враќа -1

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
```

```
	string st1,st2;
	int rezultat;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	cout<<"Vnesete vtor tekst"<<endl;
	getline(cin,st2);
	rezultat=st1.compare(st2);
	if(rezultat==-1) cout<<"ne";
	else cout<<"da";
	//cout<<rezultat;
	return 0;
}
```

Пример : спојување на стрингови

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
	string st1,st2,st3;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	cout<<"Vnesete vtor tekst"<<endl;
	getline(cin,st2);
	st3=st1.append(st2);
	cout<<st3;
	return 0;
}
```

Пример: Издвојување на подстринг

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
	string st1,st2;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	st2=st1.substr(2,4);
	cout<<st2;
	return 0;
}
```

Пример: замена на стрингови

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
```

```
	string st1,st2;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	cout<<"Vnesete vtor tekst"<<endl;
	getline(cin,st2);
	st1.swap(st2);
	cout<<st1<<" "<<st2;
	return 0;
}
```

Пример: наоѓање на почетна позиција на подстринг (find – прво појавување и rfind-последно појавување)

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
	string st1,st2;
	int rezultat;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	cout<<"Vnesete vtor tekst"<<endl;
	getline(cin,st2);
	rezultat=st1.rfind(st2);
	cout<<rezultat;
	return 0;
}
```

Пример: замена на карактери во стринг

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
	string st1,st2;
	int rezultat;
	cout<<"Vnesete prv tekst"<<endl;
	getline(cin,st1);
	st1.replace(2,3,”abc”);
	cout<<st1;
	return 0;
}
```

Пример: вметнување на еден стринг во друг од одредена позиција

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
```

```
        string st1,st2;
        int n;
        cout<<"Vnesete prv tekst"<<endl;
        getline(cin,st1);
        cout<<"Vnesete vtor tekst"<<endl;
        getline(cin,st2);
        cout<<"Vnesete pozicija"<<endl;
        cin>>n;
        st1.insert(n,st2);
        cout<<st1;
        return 0;
}
```

Пр. Програма со која се проверува дали внесениот текст е палиндром или не. (Палиндром е текст кој се чита исто во двете насоки – од лево кон десно и од десно кон лево)

```
#include<iostream>
#include<cstring>
using namespace std;
int main(){
char tekst[100],tekst_obratno[100];
bool palindrom;
int i,dolzina;
cout << "Vnesi tekst: ";
cin.getline(tekst,100);
dolzina=strlen(tekst);

for(int i=dolzina-1;i>=0;i--)
   tekst_obratno[i]=tekst[dolzina-i-1];

for(int i=0;i<=dolzina-1;i++)
  {
    if (tekst[i]==tekst_obratno[i])
       palindrom=true;
    else
      {
        palindrom=false;
        cout<<"Vneseniot tekst ne e palindrom"<<endl;
        break;
      }
  }

    if (palindrom==true)
          cout<<"Vneseniot tekst e palindrom"<<endl;
return 0;
}
```