**Лекція № 9.** **Символьні та рядкові величини.** **Використання рядків типу string.**

**ДАНІ СИМВОЛЬНОГО ТИПУ**

Рядок (або ще кажуть "рядкова змінна або рядкова величина") являє собою масив символів, який закінчується нуль-символом. Нагадаємо, що нуль-символ має код, що дорівнює **0**, і запис у вигляді керуючої послідовності **‘\0’**. За розташуванням нуль-символу визначається фактична довжина рядка. Кількість елементів символьного масиву складається з кількості символів у рядку плюс **1**, тому що нуль-символ також є елементом масиву.

Адреса першого символу рядка може використовуватися по-різному:

* якщо рядок застосовується при ініціюванні масиву типу **char**, адреса його першого елемента стає синонімом імені масиву. Наприклад, ідентичними є такі описи масиву:
* **char st [  ] = “Слово”;**
* **char st [6] = “Слово”;**
* **char st [6] = {‘С’ ‘л’ ‘о’ ‘в’ ‘о’ ‘\0’};**

При описі символьного масиву його ім’я — не змінна, а покажчик-константа на початок рядка, тому її не можна використовувати в деяких операціях адресної арифметики. Зокрема, не можна здійснювати операцію присвоювання вигляду:

**char st [20];**

**st = “Петренко”;** — **запис неправильний**, тому що не можна змінити значення **st**.

**Виконання дій з елементами символьного масиву** здійснюється через індекси або через покажчики. Покажчики ми розглядати в цьому курсі не будемо. Для доступу до будь - якого символу рядка використовується індекс масиву **char**. Тобто, якщо описана змінна **char str [3];**, то третім елементом ма­сиву можна скористатися, записавши: **str [2]**.

Для роботи з символьними масивами використовують масиви покажчиків, кожен з елементів яких містить адресу рядка масиву даних у пам’яті. Такий спосіб дозволяє зберігати дані з рваними краями, наприклад, деяку текстову інформацію. Масив з «рваними» краями схожий на двовимірну таблицю, рядки якої можуть мати різну довжину. Використання масиву покажчиків (**char \*fio[ ]**) для збереження рядків дозволяє заощаджувати пам’ять, а процес обробки рядків виконується значно швидше, бо змінюються тільки покажчики, а не вміст рядків.



Рисунок 1. Приклад масиву з "рваними" краями

**Приклад 1**. Виведення даних з масиву з "рваними" краями

**#include <iostream>**

**using namespace std;**

**main ( )**

**{ system("color F0");**

**char \*fio[ ] = { "Petrenko",**

**"Golovko",**

**"Korz",**

**"Kutz",**

**"Ushko",**

**"Plush" }; *//*** *ініціалізація масиву покажчиків*

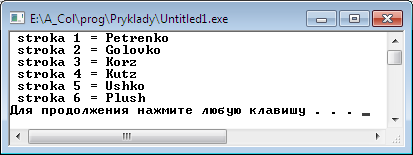
**int str;**

**for (str = 0; str <=5; str++)**

**cout << " stroka " << (str + 1) << " = " << \*(fio + str) << endl;**

**system("pause"); // для затримки екрану виведення результатів**

**}**

Результати виконання програми:  


**У процесі роботи з елементами двовимірного масиву** застосовують також індекси масиву. Якщо описаний список прізвищ **char°spis°[5]°[15];**, то для використання символу масиву слід записати: **spis [і][j]**.

Аналогічно, якщо оголошений масив покажчиків **char \*str [5]**, що містить 5 елементів, кожний з яких вказує на рядок, то доступ до символу рядка можна здійснити з використанням запису **\*(str [і] + j)**.

**Введення рядків** можна здійснювати різними способами, найбільш розповсюдженими з яких є:

* введення шляхом ініціювання при оголошенні символьних масивів:

**char st 15] = “Диск”;**  
**char st [  ] = “Диск”;**

У цьому випадку двовимірні масиви можна ініціювати по-різному, наприклад, у вигляді:

**char str [5][20] = {“Петренко И. И. “Головко С. С. “, . . . ,};**  
**char str [ ][20] = {“Петренко И. И. “, “Головко С. С. . . . ,};**

* використання потокового введення **сіn >>**. Здійснюється у випадку, коли рядок не містить пропусків, тому що символ пропуску є роздільником введення даних, наприклад:

**char st [5];  сіn >> st;**  
**char str [5][20];  сіn >> str [i];**

* посимвольне введення за допомогою функції **get( )**, наприклад: **get (st[i]);**
* введення за допомогою функції **cin.get**:

**cin.get (str[i], size, endl);,**

де **size** — кількість символів, що зчитуються;

* введення з використанням функції **cin.getline**:

**cin.getline (str[i], sizeof (str[i]-l));,**  
де **sizeof()** — функція визначення розміру рядка.

Виведення рядкових даних реалізується з використанням стандартного вихідного потоку **cout**:

**cout << st;**

**cout.write(st, size);** тощо.

Для потокового введення-виведення доцільно застосовувати функції **setw(w),setprecision(d), cout.width(w) і cout.precision(d).**

Введення-виведення символьних масивів можна здійснити за допомогою відповідних функцій заголовного файла **stdio.h.**, наприклад:

* для введення рядків — **gets(st); та scanf (% s,st);**
* для виведення рядків **— puts(st); і prin,tf(% s,st);.**

ранніх версіях С++ рядки розглядалися як символьні масиви. Для роботи з ними розроблено бібліотеку функцій **string.h,** що містить ефективні засоби для роботи з рядками. Згодом була розроблена стандартна бібліотека шаблонів **Standard TemplateLibrary (STL),** яка надає більш потужні засоби, об’єднані в клас **string**.

Для обробки символьних типів даних бібліотека функцій **string.h** має велику кількість вбудованих функцій:

* функції перевірки символів;
* функції перетворення символів;
* функції перевірки рядків;
* функції маніпулювання рядками.

Функції наводяться у вигляді списків, що згруповані за їх розташуванням у заголовних файлах.

**Використання рядків типу string**

**Р**аніше розглядалися питання обробки символьних даних мови С++, у тому числі і символьних рядків. Однак в останніх версіях мови С++, починаючи з С++4.5, введена стандартна бібліотека шаблонів — **Standard Template Library (STL)**, яка містить клас **string** з більш потужними засобами обробки рядків.

Для підключення цього класу до програми слід записати директиву:

**#include <string>** (без розширення ***.h***)

і підключити простір імен бібліотеки шаблонів у вигляді

**using namespace std; .**

Після цього можна оголошувати змінні типу **string:**

**string strl, str2;.**

Ініціювання рядків при оголошенні виконується одним із способів:

**string st1 = “Це рядок класу string”;**  
**string st2 (“Це інший рядок класу string”);.**

Значення рядка **string** містить будь-який набір символів, записаний у лапках.

Для рядків типу **string** визначено такі операції:

* конкатенації (приєднання), котрі позначаються символом «+»;
* відношення («==», «!=», «>», «>=», «<», «<=»).

Наприклад, фрагмент  
**string st1 = “Приклад”;**  
**string st2 = “рядка”;**  
**string st3 = st1+’ ‘+st2;**  
**cout << st3 << endl;**  
дозволить вивести на екран повідомлення: **Приклад рядка.**

Для введення рядків **string**, крім операторів присвоювання, застосовують оператори введення даних:  
**сіn >> st;**  
**cin.getline(st, siseof(st));**  
**getline(cin,st, ‘\n’); тощо.**

Виведення рядків на екран здійснюється шляхом використання звичайних операторів виведення даних.

Рядки можна об’єднувати у масиви, які оголошуються звичайним засобом, тобто

**string sp[10];** — оголошення масиву, що містить 10 рядків.

Доступ до символів рядка здійснюється шляхом запису порядкового номера символу —індексу, який починається з нуля. Індекси можна записувати як у квадратних дужках, так і у звичайних, круглих.

Наприклад, якщо записати  
**string str=”Mій рядок”;,**  
то **str[2]** — це буде літера **‘й’.**

Для масивів рядків потрібний символ визначається шляхом запису двох індексів: індексу елемента масиву та індексу символу в цьому елементі, тобто у вигляді **mas[i][j].**

**Функції для обробки рядків типу string**

Існує багато функцій для обробки рядків типу **string,** розглянемо деякі з них.

***Функції визначення довжини рядка:***

* **str.size();**
* **str.length();**
* **str.max\_size(); .**

Наприклад:

*//--------------------------- определение длины строки*

**#include <iostream.h>**

**#include <string>**

**using namespace std;**

**main()**

**{ string str, st ="И вот сама пришла волшебница зима!";**

**str =" волшебница";**

**cout << "Длина строки str = " << str.size() << "Длина строки st = " << st.length() << endl;**

**}**

Результат виконання:  
**Длина строки str =11 Длина строки st = 34**

***Функції додавання одного рядка або його частини до іншого рядка:***

* **Str.append(st);** — додає рядок **st** до кінця рядка **srt;**
* **Str.append(st,k,n);** — додає до рядка str n символів рядка **st,** починаючи з позиції **k.**

Наприклад:

*//--------------- додавання рядка*

**string str, st = "Substring in text";**

**str.append(st,3,6);**

**cout << "str = "<<str << endl;**

Результат виконання:  
**str = string**

***Функція включення рядка в рядок:***

* **str.insert(k,st)** — вставляє в рядок **str** з позиції **k** рядок **st**;
* **str.insert(k1,st,k2,n)** — вставляє в рядок **str** з позиції **kl** n символів рядка **st**, починаючи з позиції **k2** в рядку **st**.

Наприклад:  
*// вставка підрядка в рядок*  
**string str, st = “string”;**  
**str = “My text”;**  
**str.insert(3,st);**  
**cout << “str = “<< str << endl;**

Результат: **str = My string text**  
**string str, st = “string “;**  
**str = “My text”;**  
**str.insert(3,st,2,4);**  
**cout << ” str = ” << str << endl;**

Результат: **str = My ring text**

***Функція вилучення символів із рядка:***

* **str.remove(k,n)** — вилучає **n** символів з рядка **str**, почи­наючи з позиції **k**.

Наприклад:  
// вилучення підрядка з рядка  
**string str = “My text string”;**  
**str.remove(3,4);**  
**cout << “str = “<< str << endl;**

Результат виконання:  
**str = My string.**

***Функція заміни частини рядка або усього рядка:***

* **str.replace(st)** — заміняє рядок **str на st**;
* **str.replace(k,n,st)** — заміняє в рядку **str n** символів, по­чинаючи з позиці**ї k** рядка **st**;
* **str.replace(kl,nl,st,k2,n2)** — заміняє в рядку **str nl** сим­волів з позиції **kl** частиною в**n2** символи рядка **st**, починаючи з позиції **k2**.

Наприклад:  
// *заміна підрядка в рядку*  
**string str, st = “substring”;**  
**str =”My text string”;**   
**str.replace(8,9,st);**  
**cout <<“str = “<< str<<endl;**

Результат виконання:  
**str = My text substring**  
 **string str, st = ” substring”;**  
**str=”My text long”;**  
**str.replace(3,5,st,l,3);**  
**cout <<” str = “<< str<<endl;**

Результат виконання:  
**str = My sublong**

***Функція обміну змістом двох рядків:***

* **str.swap(st)** — обмінює зміст рядків **str та st**.

Наприклад:  
*//————- обмін вмісту рядків*  
**string str, st=”Coлнечная погода”;**  
**str = “Дождливая погода”;**  
**str.swap(st);**  
**cout <<” str = “<<str<<” st = “<<st<<endl;**

Результат виконання:  
**str = Солнечная погода st = Дождливая погода.**

***Функція виділення частини рядка:***

* **str.substr(k,n)** — повертає частину рядка **str** в **n** символів, починаючи з позиції **k.**

Наприклад:  
*//————- виділення частини рядка*  
**string str, st;  
str=”Moй текст string “;  
st=str. substr(4,13);  
cout << ” st = ” << st << endl;**

Результат виконання:  
**st = текст string.**

***Функція пошуку позиції входження підрядка в рядок:***

* **str.find(st,k)** — шукає зліва граничну позицію входження рядка **st** в рядок **str**, починаючи з **k**-ої позиції рядка **str;**
* **str.rfind(st,k)** — шукає справа граничну позицію входження рядка **st** в рядок **str**, починаючи з **k**-ої позиції рядка **str.**

Наприклад:  
// *позиція входження підрядка в рядок*  
**string str, st=”nopa”;**  
**str=”Oceнняя пора – дождливая пора!”**   
**int p=str.find(st,0);**  
**cout <<” p=”<<p<<endl;**

Результат виконання:  
**р=8.**  
**string str, st=”nopa”;**  
**str=”Oceнняя пора – дождливая пора!”  
int p=str.rfind(st,str.length);  
cout <<” p=”<<p<<endl;**  
Результат виконання:  
**р=25.**

**Приклади використання рядків типу string.**

**В** останніх версіях мови С++, починаючи з С++4.5, введена стандартна бібліо­тека шаблонів — **Standard Template Library (STL)**, яка містить клас **string** із засобами обробки рядків.

Для **підключення** цього **класу** до програми слід записати директиву:

**#include <string>** (без розширення ***.h***)

і **підключити простір імен** бібліотеки шаблонів у вигляді

**using namespace std; .**

Після цього можна **оголошувати змінні** типу **string:**

**string strl, str2;.**

**Ініціювання** рядків при оголошенні виконується одним із способів:

**string st1 = “Це рядок класу string”;**  
**string st2 = (“Це інший рядок класу string”);.**

**Оголошення** **покажчика** на рядок здійснюється так:  
**string \*pst;.**

Пам’ять для покажчика може бути виділена з будь-яким початковим значенням за допомогою функції **new**, наприклад:  
**string \*pstr1 = new string;** — оголошується порожній рядок,  
**string \*pstr2 = new string (“Новая строка”);** — покажчик вказує на рядок «Новая строка».

Раніше оголошеному покажчику **\*pst,** який ні на що не вказує, можна присвоїти значення у вигляді  
**pst = new string (“Это первая строка”);.**

Значення рядка **string** містить будь-який набір символів, записаний у лапках.

Для рядків типу **string** визначено такі операції:

* конкатенації (приєднання), котрі позначаються симво­лом «+»;
* відношення («==», «!=», «>», «>=», «<», «<=»).

Наприклад, фрагмент  
**string st1 = “An”;**  
**string st2 = “apple”;**  
**string st3 = st1+’ ‘+st2;**  
**cout << st3 << endl;**  
дозволить вивести на екран повідомлення: **An apple.**

Для введення рядків **string**, крім операторів присвоювання, застосовують оператори введення даних:  
**сіn >> st;**  
**cin.getline(st, siseof(st));**  
**getline(cin,st, ‘\n’); тощо.**

Виведення рядків на екран здійснюється шляхом використання звичайних операторів виведення даних.

Рядки можна об’єднувати у масиви, які оголошуються звичайним засобом, тобто

**string sp[10];** — оголошення масиву, що містить 10 рядків.

Доступ до символів рядка здійснюється шляхом запису порядкового номера символу — індексу, який починається з нуля. Індекси можна записувати як у квадратних дужках, так і у звичайних, круглих.

Наприклад, якщо записати  
**string str=” An apple ”;,**  
то **str[2]** — це буде літера **‘n’.**

Для масивів рядків потрібний символ визначається шляхом запису двох індексів: індексу елемента масиву та індексу символу в цьому елементі, тобто у вигляді **mas[i][j].**

Існує багато функцій для обробки рядків типу **string,** розглянемо деякі з них.

***Функції визначення довжини рядка:***

* **str.size();**
* **str.length();**
* **str.max\_size(); .**

Наприклад:

*//--------------------------- определение длины строки*

**#include <iostream.h>**

**#include <string>**

**using namespace std;**

**main()**

**{ string str, st ="**As busy as a bee**";**

**str ="**bee**";**

**cout << "Length str = " << str.size() << " Length строки st = " << st.length() << endl;**

**}**

Результат виконання:  
**Длина строки str =3 Длина строки st = 16**

***Функції додавання одного рядка або його частини до іншого рядка:***

* **Str.append(st);** — додає рядок **st** до кінця рядка **srt;**
* **Str.append(st,k,n);** — додає до рядка str n символів рядка **st,** починаючи з позиції **k.**

***Функція включення рядка в рядок:***

* **str.insert(k,st)** — вставляє в рядок **str** з позиції **k** рядок **st**;
* **str.insert(k1,st,k2,n)** — вставляє в рядок **str** з позиції **kl** n символів рядка **st**,починаючи з позиції **k2** в рядку **st**.

Наприклад:  
*// вставка подстроки в строку*  
**string str, st = “string”;**  
**str = “My text”;**  
**str.insert(3,st);**  
**cout << “str = “<< str << endl;**

Результат: **str = My string text**  
**string str, st = “string “;**  
**str = “My text”;**  
**str.insert(3,st,2,4);**  
**cout << ” str = ” << str << endl;**

Результат: **str = My ring text**

***Функція вилучення символів із рядка:***

* **str.remove(k,n)** — вилучає **n** символів з рядка **str**, почи­наючи з позиції **k**.

Наприклад:  
**string str = “My text string”;**// удаление подстроки из строки  
**str.remove(3,4);**  
**cout << “str = “<< str << endl;**

Результат виконання:  
**str = My string.**  
***Функція заміни частини рядка або усього рядка:***

* **str.replace(st)** — заміняє рядок **str на st**;
* **str.replace(k,n,st)** — заміняє в рядку **str n** символів, починаючи з позиці**ї k** рядка **st**;
* **str.replace(kl,nl,st,k2,n2)** — заміняє в рядку **str nl** символів з позиції **kl** частиною в**n2** символи рядка **st**, починаючи з позиції **k2**.

Наприклад:  
// *замена подстроки в строке*  
**string str, st = “substring”;**  
**str =”My text string”;**   
**str.replace(8,9,st);**  
**cout <<“str = “<< str<<endl;**

Результат виконання:  
**str = My text substring**  
 **string str, st = ” substring”;**  
**str=”My text long”;**  
**str.replace(3,5,st,l,3);**  
**cout <<” str = “<< str<<endl;**

Результат виконання:  
**str = My sublong**  
  
***Функція обміну змістом двох рядків:***

* **str.swap(st)** — обмінює зміст рядків **str та st**.

Наприклад:  
*//————- обмен содержимым строк*  
**string str, st=”Coлнечная погода”;**   
**str = “Дождливая погода”;**  
**str.swap(st);**  
**cout <<” str = “<<str<<” st = “<<st<<endl;**

Результат виконання:  
**str = Солнечная погода st = Дождливая погода.**

***Функція виділення частини рядка:***

* **str.substr(k,n)** — повертає частину рядка **str** в **n** симво­лів, починаючи з позиції **k.**

Наприклад:  
**string str, st;** *//————- выделение части строки* **str=”Moй текст string “;  
st=str. substr(4,13);  
cout << ” st = ” << st << endl;**

Результат виконання:  
**st = текст string.**

***Функція пошуку позиції входження підрядка в рядок:***

* **str.find(st,k)** — шукає зліва граничну позицію входжен­ня рядка **st** в рядок **str**,починаючи з **k**-ої позиції рядка **str;**
* **str.rfind(st,k)** — шукає справа граничну позицію входження рядка **st** в рядок **str**,починаючи з **k**-ої позиції рядка **str.**

Наприклад:  
// *позиция вхождения подстроки в строку*  
**#include <iostream>**

**#include <string>**

**#include <conio.h>**

**using namespace std;**

**void main ()**

**{ string str, st="as";**

**str="as large as life!" ;**

**int p=str.find(st,0);**

**cout <<" p="<<p<<endl;**

**p=str.rfind(st,str.length());**

**cout <<" p="<<p<<endl;**

**getch ();**

**}**Результат виконання:  
**р=0 та р=9.**

***Контрольні запитання для самоперевірки***.

1. Що таке рядки та значення елементів символьного типу?
2. Що являє собою масив символьного типу?
3. Як здійснюється введення символьних даних?
4. Які операції можна здійснити з рядками типу string?
5. Як можна ініціювати рядки типу string?
6. Як можна визначити символ у рядках типу string?
7. Як здійснюється введення рядків типу string?
8. Як виконується порівняння даних рядків типу string?
9. Як визначити кількість символів у рядку типу string?
10. Які функції мови С++ необхідні для виділення підрядка з рядка типу string?

**Для самостійного вивчення** *(2 години)*: Вивчення лекційного матеріалу та додаткових джерел. Розгляд запитань і виконання завдань для самостійної роботи, запропонованих на лекції.

**Рекомендована література**

1. Ковалюк Т. В. Алгоритмізація та програмування: Підручник. — Львів: «Магнолія 2006», 2013. — 400 с., ил.
2. Павловская Т.А. С/С++. Программирование на языке высокого уровня. СПб.: Питер, 2003. – 461 с. URL: <http://www.ph4s.ru/bookprogramir_1.html>
3. Вступ до програмування мовою С++. Організація обчислень: навч. посіб. / Ю. А. Бєлов, Т. О. Карнаух, Ю. В. Коваль, А. Б. Ставовський. – К. : Видавничо-поліграфічний центр "Київський університет", 2012. – 175 с. с.: іл. ISBN (укр.) . URL: <http://csc.knu.ua/uk/library/books/belov-24.pdf>
4. Джейс Либерти Освой самостоятельно С++ за 21 день: 3-е изд. пер. с англ.: Уч. пос. – М.: Издательский дом „Вильямс”, 2001. – 816 с.: ил..

5. Дейтел Х., Дейтел П. Основы программирования на С++. – М.: Бином, 1999. – 1024 с. URL: <http://ijevanlib.ysu.am/wp-content/uploads/2018/03/deytel.pdf>