**Лекція 8.** **Використання рядків 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”);.**

**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 «Новаястрока».

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 = “Динамическая”;**  
**string st2 = “память”;**  
**string st3 = st1+’ ‘+st2;**  
**cout << st3 << endl;**  
дозволить вивести на екран повідомлення: **Динамическая память.**

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.

Наприклад, якщо записати  
**string str=”Moя строка”;,**  
то **str[2]** — це буде літера **‘я’.**

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.

**Переваги та недоліки дає використання класу string у порівнянні з типом char\***

Створення нового типу string було обумовлене недоліками роботи з рядками символів, що демонстрував тип char\*. У порівнянні з типом char\* тип string має такі основні переваги:

* можливість обробки рядків стандартними операторами C++ (**=**, **+**, **==**, **<>** тощо). Як відомо, при використанні типу char\* навіть найбільш прості операції з рядками виглядали складно і вимагали написання надмірного програмного коду;
* забезпечення кращої надійності (безпеки) програмного коду. Наприклад, при копіюванні рядків, тип string забезпечує відповідні дії, що можуть виникнути у випадку, якщо рядок-джерело має більший розмір за рядок-приймач;
* забезпечення рядка, як самостійного типу даних. Оголошення типу string як рядка є єдине для усіх змінних в програмі, що забезпечує непротирічність даних.

Основним недоліком типу string у порівнянні з типом char\*, є уповільнена швидкість обробки даних. Це зв’язано з тим, що тип string – це, фактично, контейнерний клас. А робота з класом вимагає додаткової реалізації програмного коду, що, в свою чергу займає зайвий час.

***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:

*//--------------- добавление строки*

**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 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:  
// удаление подстроки из строки  
**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:  
// *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:  
*//————- выделение части строки*  
**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:  
// *позиция вхождения подстроки в строку*  
**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.**

***Функція присвоювання усього рядка або його частини іншому рядку:***

* **str.assign(st)** — присвоює весь рядок **st** типу **string** або масив **char[ ]** — рядку **str**типу **string;**
* **str.assign(st,k,n)** — присвоює **n** символів рядка **st** рядку **str,** починаючи з **k**-ої позиції.

Ці функції можна використовувати для перетворення рядка типу **char** у рядок **string**.Наприклад:  
*//————————– присваивание строк*  
**char st[ ]=”Пришло жаркое лето!”;**  
**string str;**  
**cout<<” str = “<< str.assign(st,7,12)<<endl;**

Результат виконання:  
**str = жаркое лето!**  
або  
**string str, st=”O, лето красное, любил бы я тебя!”;  
cout<<“str = “<< str.assign(st,3,12) << endl;**

Результат виконання:  
**str = лето красное.**  
***Функція порівняння рядків або їхніх частин:***

* **str.compare(st)** — порівнює рядки **st** та **str** і повертає значення:

**<0 — якщо st < str;**  
**=0 — якщо st = str;**  
**>0 — якщо st > str;;**

* **str.compare(st,k,n)** — порівнює n символів рядка **st** з рядком **str,** починаючи з **k**-ої позиції. Наприклад:

*//————— сравнение частей строк*  
**string str, st=”oв”;**  
**str=”Иванов”;**  
**if (str.compare(st,4,2)==0)**  
**cout<<“str= “<< str << endl;**  
Результат виконання:  
**str = Иванов.**  
***Функція перетворення рядка типу string у рядок типу char:***

* **str.c\_str()** — перетворює рядок типу **string** у рядок типу **char.**

**Як конвертувати рядок в ціле число**

C++ — **строго типізована** мова програмування. Це означає, що коли ви створюєте змінну, ви повинні явно оголосити, який тип значення буде зберігатися в ній.

Як згадувалося раніше, C++ є строго типізованою мовою.

Якщо ви спробуєте надати значення, яке не відповідає типу даних, ви отримаєте помилку.

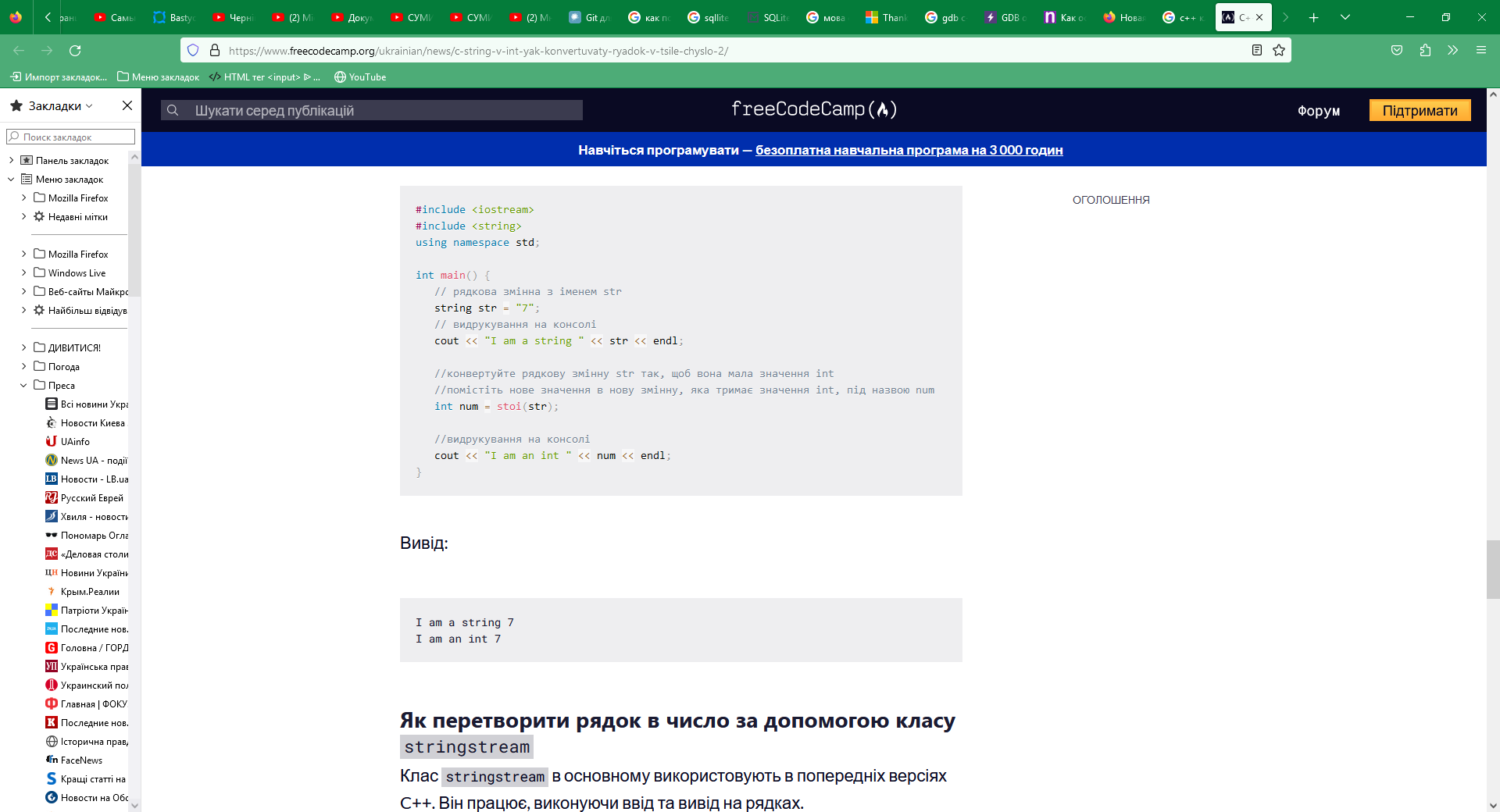
Крім того, конвертувати рядок в ціле число не так просто, як використати приведення типів, яке можна використовувати під час конвертації double в int.

Наприклад, ви **не можете** зробити так:

Одним з ефективних способів перетворити рядковий об’єкт в число — це використати функцію stoi().

Цей метод зазвичай використовується для новіших версій C++, і вводиться в C++11.

Вона приймає рядкове значення як ввід та повертає ціле число як вивід.



Перелік функцій перетворення

|  |  |
| --- | --- |
| [stod](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stod) | Перетворює послідовність символівв **double**. |
| [stof](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stof) | Перетворює послідовність символів в **float**. |
| [stoi](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stoi) | Перетворює послідовність символів в **int**. |
| [stold](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stold) | Перетворює послідовність символів в **long double**. |
| [stoll](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stoll) | Перетворює послідовність символів в **long long**. |
| [stoul](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stoul) | Перетворює послідовність символів в **unsigned long**. |
| [stoull](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#stoull) | Перетворює послідовність символів в **unsigned long long**. |
| [to\_string](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#to_string) | Перетворює значення в string. |
| [to\_wstring](https://learn.microsoft.com/ru-ru/cpp/standard-library/string-functions?view=msvc-170#to_wstring) | Перетворює значення в розширений рядок. |

**wstring** - це еквівалент vector<wchar\_t> , вектор «широких», локалізованих символів, представляють інтернаціональну кодування символами **Unicode**. Розбір вмісту (пошук, виділення слів, розбиття на рядки тощо) російськомовної або будь-якого іншого рядка (китайської наприклад) можна робити тільки в форматі **wstring** (не **string**).

Розглянемо приклади використання наведених функцій при обробці рядків типу **string.**

***Приклад 9.*** Увести до пам’яті комп’ютера будь-який текст, відокремити в ньому всі слова, вивести їх на екран та визначити найдовше слово

// *9.СРР — выделить слова в тексте и найти среди них наибольшее*

**#include <iostream>**

**#include <string>**

**#include <conio.h>**

**using namespace std;**

**main ()**

{ **string txt, s1, s1max;**          *//* *shnax — наибольшее слово*

**int і, k=0, n=0, max=0;**

*//---------------- ввод текста*

**cout <<"\*\*\*\*\* Enter text\n";**

**getline(cin,txt,'\n');**

**for (i = 0; і <= txt.length()-l; i++)**

**{ if ((txt[i]==' ') || (txt[i]==',') || (txt[i]=='.'))**

**{n++;**   *// n — счетчик пропусков и знаков препинания*

**if (n>1) continue; k++;**   //*k — счетчик слов*

**cout<< k <<" slovo - "<< s1<<" = "<<s1.size() << " simbols\n";**

*//--------------- определение слова с наибольшим количеством символов*

**if (sl.size() > max)**

**{ max = sl.size();**

**slmax=sl; }**

**sl="";**    *//освобождение sl для формирования слова*

}

**else {n=0; sl=sl+txt[i]; }** *//переменная s1 — формирует слово*

**}**

**cout<<"\nslmax — "<< slmax<<" = "<<slmax.size()<<" simbols\n";**

**getch ();**

**}**

Результати обчислень:  
**\*\*\*\*\* Enter text**  
**Учитель, воспитай ученика, чтобы было у кого учиться.**   
**1   slovo — Учитель = 7 simbols**  
**2   slovo — воспитай = 8 simbols**  
**3   slovo — ученика = 7 simbols**  
**4   slovo — чтобы = 5 simbols**  
**5   slovo — было = 4 simbols**  
**6   slovo — у = 1 simbols**  
**7   slovo — кого = 4 simbols**  
**8   slovo — учиться = 7 simbols**  
**slmax — воспитай = 8 simbols.**

У програмі текст вводиться за допомогою функції **getline()** (зверніть увагу на її вигляд), слова виділяються не з використанням функцій, а шляхом конкатенації символів, якщо вони не є символами-роздільниками.  
***Приклад 10.*** Увести список прізвищ і відсортувати його за алфавітом

// *10.СРР — сортировка фамилий по алфавиту*

**#include <iostream.h>**

**#include <string>**

**#include <conio.h>**

**using namespace std;**

**main()**

**{ const n=5;**

**string spis[n];**

**int i, k;**

*//--------------------------- ввод списка фамилий*

**for(i = 0; і < n; i++)**

**{ cout<<"\*"\*\* Enter "<<(i+l)<<" name\n";**

**getline(cin,spis[i],'\n');  }**

//--------------------------- сортировка списка фамилий

**for (k = 1; k < n; k++)**

**for (і = 0; і < n-k; i++)**

**if (spis[i] > spis[i+l]) spis[i].swap(spis[i+l]);**

*//--------------------------- вывод отсортированного списка*

**cout<<"\n\*\*\*\*\* Rezult spisok\n";**

**for(i = 0; і < n; i++)**

**cout << (i+1) << ' ' << spis[i] << endl;**

**getch ();**

**}**

Результати виконання:  
**\*\*\*\*\* Enter 1 name**  
**Котин С. Б.**  
**\*\*\*\*\* Enter 2 name**  
**Бобров М. Т.**  
**\*\*\*\*\* Enter 3 name**  
**Авдеев P. Л.**  
**\*\*\*\*\* Enter 4 name**  
**Попов Г. Д.**  
**\*\*\*\*\* Enter 5 name**  
**Елкин П. P.**  
**\*\*\*\*\* Rezult spisok**  
**1   Авдеев P. Л.**  
**2   Бобров M. Т.**  
**3   Елкин П. Р.**  
**4   Котин С. Б.**  
**5   Попов Г. Д.**

Для порівняння прізвищ у програмі використовується звичайна операція «>», а для взаємозаміни — функція **swap()** (проаналізуйте програми прикладу 7.10 і прикладу 7.4).

**4   слово — обещая — содержит 6 символов**  
**5   слово — великого — содержит 8 символов**

***Контрольні запитання для самоперевірки***.

1. Які операції можна здійснити з рядками типу string?
2. Як можна ініціювати рядки типу string?
3. Як можна визначити символ у рядках типу string?
4. Як здійснюється ініціювання покажчика на рядок типу string?
5. Як здійснюється введення рядків типу string?
6. Як виконується порівняння даних рядків типу string?
7. Наведіть приклад використання операції конкатенації для рядків типу string.
8. Як визначити кількість символів у рядку типу string?
9. 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>