

Оптимизация работы со строками с помощью строчковых потоков

Фатеенков Данила Витальевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассматривается функционал библиотеки «sstream», которая предназначена для организации строчковых потоков ввода и вывода информации в программах, написанных с применением языка программирования C++. Также проводится сравнение подходов через алгоритмизацию без использования и с использованием строчковых потоков к решению задач с задействованием строк.

Ключевые слова: строки, строчковые потоки, оптимизация, C++, iostream, stringstream

Optimizing string handling with string streams

Fateenkov Danila Vitalievich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article describes the functionality of the "sstream" library, which is designed to organize string input and output streams in programs written with the C++ programming language. It also compares approaches through algorithmization without and with the use of string streams to solve problems involving strings.

Keywords: strings, string streams, optimization, C++, iostream, stringstream

1. Введение

1.1 Актуальность

Работа со строками в алгоритмах и программном обеспечении является одной из основных частей разработки. Операции и функции, предназначенные для обработки строчковых данных, во многих случаях получаются громоздкими из-за недостатка функций основного потока ввода и вывода для обработки строк. В таких случаях реализованы отдельные модули и библиотеки, которые упрощают разработку и зачастую оптимизируют процесс обработки информации, которая представлена в виде строк.

Одной из таких библиотек является «sstream». Данный заголовочный файл входит в стандартную библиотеку C++ и предназначен для организации строчковых потоков в приложении разработчика.

1.2 Обзор исследований

Guram Kashmadze описал алгоритм построения трёхмерной структуры, включающей в себя построение всех внутренних точек – сфера Блоха [1]. В статье описал код построения структуры через нечёткое множество, представленный на языке программирования C++, используя также библиотеку «sstream» для преобразования типов данных.

А.В. Шевчук, Л.Р. Сотников, Ю.Д. Копойцев и А.А. Харичев в своей статье описали разработку программы для видеозахвата изображения с экрана компьютера персонального компьютера [2]. При разработке использовалась библиотека «sstream» для организации строковых потоков для обработки информации в программе.

Wu Renke, Huang Linpeng, Yu Peng и Zhou Haojie в своей исследовательской работе «SunwayMR: A distributed parallel computing framework with convenient data-intensive applications programming» представили структуру параллельных вычислений под названием SunwayMR [3]. Данная структура является программируемой и масштабируемой, а для её работы необходима только среда разработки g++. При реализации задачи также использовались строковые потоки, и их организация осуществлялась с помощью библиотеки «sstream».

1.3 Цель исследования

Цель – описать методы оптимизации и упрощения работы со строками, применяя строковые потоки.

2. Материалы и методы

Для реализации поставленной задачи используется язык программирования C++ и библиотека «sstream».

3. Результаты и обсуждения

Библиотека «sstream» является встроенной (то есть входит в стандартную библиотеку C++) в язык программирования C++, поэтому нет необходимости скачивать её со сторонних ресурсов. Библиотека «sstream» является заголовочным файлом с классами, которые предназначены для организации работы со строками через потоки.

«sstream» входит также в стандартное пространство имён, то есть все функции и переменные данного заголовочного файла входят в std (таким образом для работы с ними можно задать стандартное пространство имён при написании кода программы).

«sstream» также можно использовать для организации потоков ввода и вывода данных в программе. Для этого реализованы классы istringstream и ostreamstream. Эти классы объединены в один общий класс, который позволяет организовать потоки ввода и вывода: stringstream (он будет использоваться в дальнейшем).

Строковые потоки в первую очередь делят исходную строку на массив слов по заранее определённому разделителю – пробелу. Это позволяет

упростить обработку строки, если необходимо найти какое-либо слово. Ниже представлен пример реализации задачи по разделению строки на слова без использования строковых потоков:

```
string s;
getline(cin, s);
string buf = "";
for (int i=0;i<s.length();i++) {
    if (s[i] == ' ') {
        cout << buf << endl;
        buf = "";
    }
    else {
        buf += s[i];
    }
}
cout << buf << endl;
```

Алгоритм вносит каждую букву строки в буфер, пока не будет достигнут пробел. Если символ является пробелом, то слово выводится и буфер очищается.

Ниже представлена реализация алгоритма, который решает ту же самую задачу, но с задействованием строкового потока:

```
string s;
getline(cin, s);
stringstream ss(s);
string temp;
while (ss >> temp) {
    cout << temp << endl;
}
```

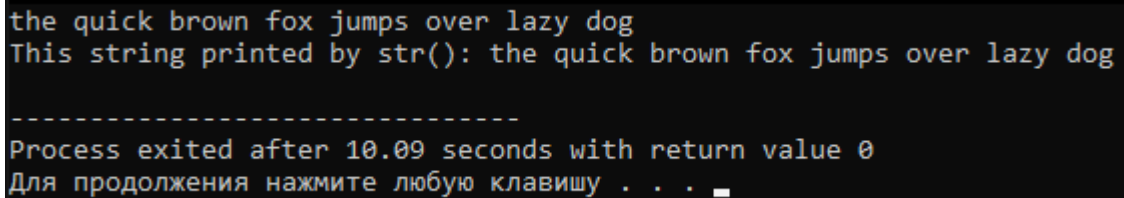
Алгоритм получился компактнее и простым для понимания:

Сначала объявляется объект класса «stringstream» (строковый поток) и в него заносится заранее введённая пользователем строка. После во временную переменную вносятся слова из строкового потока до тех пор, пока не будет достигнут конец строки.

В данном случае оператор «>>», который указан в условии выполнения цикла, является разделителем для строки, которая содержится в потоке ss. Данный оператор перебирает строковый поток и на каждое его использование приходится одно слово из строки.

Если необходимо получить всю строку целиком без деления, то можно обратиться к содержимому потока через метод «str()»:

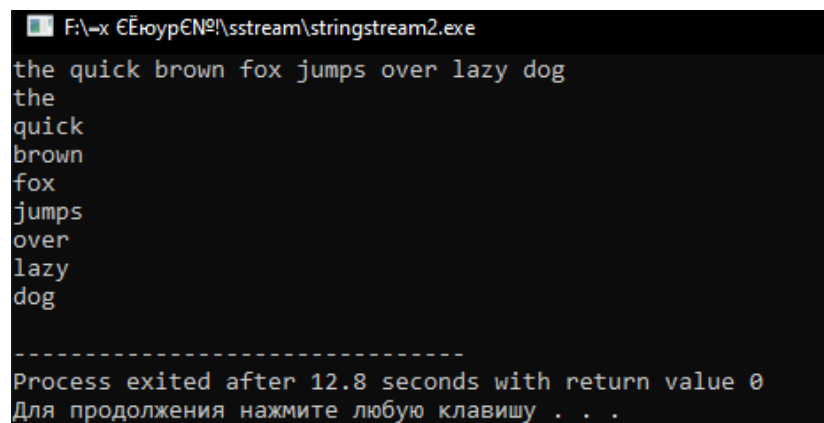
```
string s;
getline(cin,s);
stringstream ss(s);
string temp;
cout << "This string printed by str(): " <<
ss.str() << endl;
```



```
the quick brown fox jumps over lazy dog
This string printed by str(): the quick brown fox jumps over lazy dog
-----
Process exited after 10.09 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1. Вывод строки из потока через str()

Разделение исходной строки на слова упрощает ряд задач: подсчёт количества слов, вероятность появления того или иного слова, разбор слов на составляющие и др.

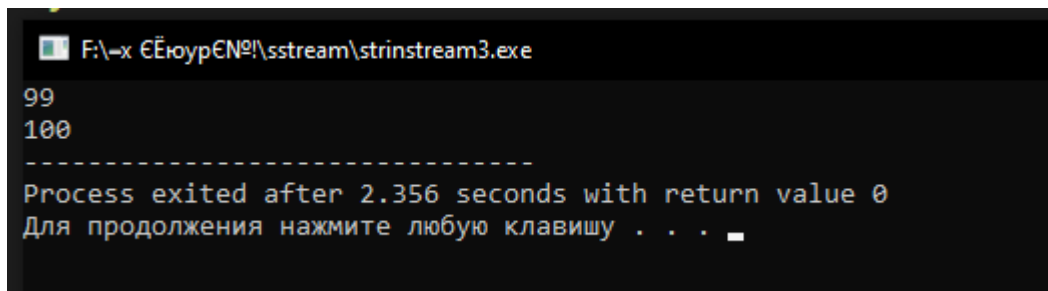


```
F:\-x EЮрpEN№\sstream\stringstream2.exe
the quick brown fox jumps over lazy dog
the
quick
brown
fox
jumps
over
lazy
dog
-----
Process exited after 12.8 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2. Пример работы алгоритма разделения строки на слова

В C++ может быть проблематичным преобразование между строками и целочисленными типами данных. Строковые потоки также упрощают данный процесс: если в поток на выход поместить строку, которую необходимо преобразовать в число и после на вход подать целочисленную переменную, то она получит значение строки на выходе и произойдёт преобразование типа переменной (см. рис. 3):

```
string s;
getline(cin,s);
stringstream ss;
ss << s;
int temp;
ss >> temp;
cout << temp + 1;
```



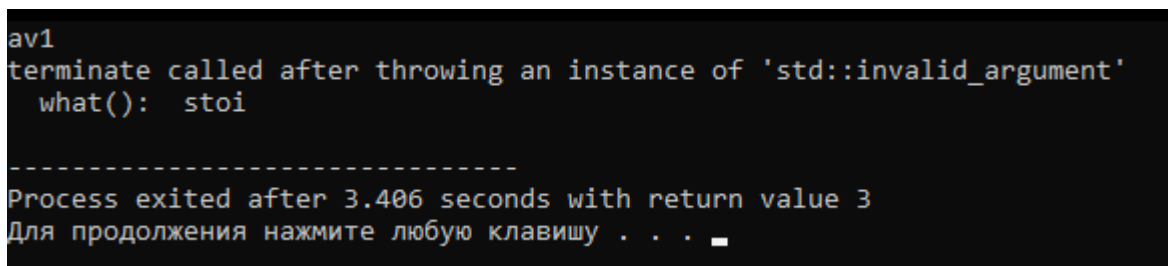
```
F:\>x €Ёюр€N!\sstream\strinstream3.exe
99
100
-----
Process exited after 2.356 seconds with return value 0
Для продолжения нажмите любую клавишу . . . _
```

Рисунок 3. Пример работы алгоритма преобразования строки в число

В C++ присутствуют команды для преобразования типов данных (строку в целочисленное и символ в целочисленное): `stoi()` и `atoi()`. Данные функции были представлены в языке программирования C, но также доступны и в C++. Пример реализации алгоритма преобразования строки в число с использованием `stoi()`:

```
string s;
getline(cin, s);
int temp = stoi(s);
cout << temp + 1;
```

В данном случае код получается компактнее, и он также прост для понимания, но у строковых потоков есть значительное преимущество: строковые потоки отслеживают исключения, которые могут привести к критическому сбою, и при попытке преобразовать буквенные символы в строку будет возвращён 0. При попытке преобразовать строку, которая содержит буквы, через `stoi()`, будет возвращена ошибка (см. рис. 4).



```
av1
terminate called after throwing an instance of 'std::invalid_argument'
what(): stoi
-----
Process exited after 3.406 seconds with return value 3
Для продолжения нажмите любую клавишу . . . _
```

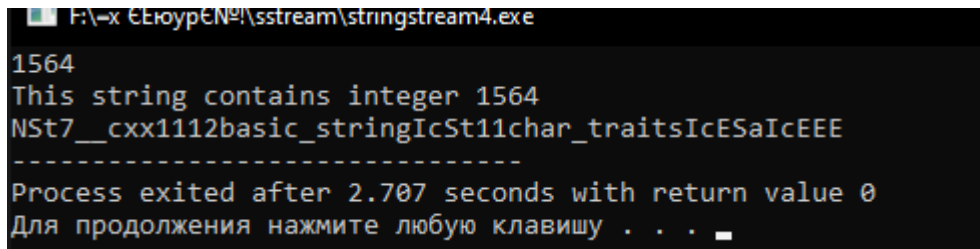
Рисунок 4. Результат преобразования буквенной строки в целое число

Соответственно, можно также и число преобразовать в строку с помощью потоков:

```
stringstream ss;
int num;
cin >> num;
ss << num;
string s;
ss >> s;
```

```
string result = "This string contains integer " +
s;
cout << result << endl << typeid(result).name();
```

typeid().name() не всегда возвращает конкретное название переменной, поэтому результат может отличаться от ожидаемого (см. рис. 5), но в результате работы алгоритма тип переменной result является строковым.

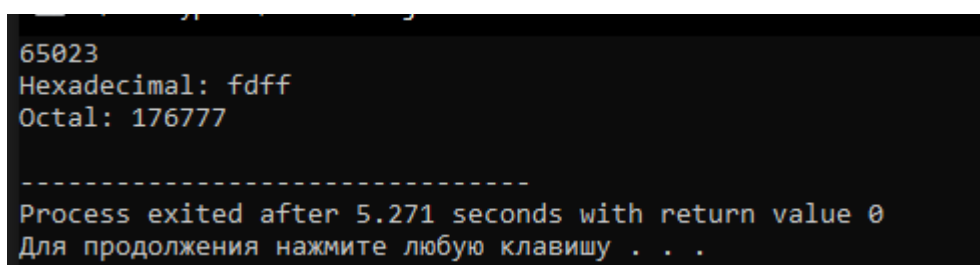


```
F:\x\ЕроупЕН\stream\stringstream4.exe
1564
This string contains integer 1564
NSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEE
-----
Process exited after 2.707 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Рисунок 5. Пример преобразования числа в строку через строковые потоки

Помимо преобразования строки в число, также можно переводить десятичные числа в другие разряды (и наоборот). Не используя сторонние библиотеки, можно преобразовать в восьмеричный и шестнадцатеричный разряд (для двоичного необходимо использовать поток вывода отдельно):

```
stringstream ss;
int num;
string result;
cin >> num;
ss << hex << num;
result = ss.str();
cout << "Hexadecimal: " << result << endl;
ss.str("");
ss << oct << num;
result = ss.str();
cout << "Octal: " << result << endl;
ss.str("");
```



```
65023
Hexadecimal: fdff
Octal: 176777
-----
Process exited after 5.271 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Рисунок 6. Пример преобразования десятичного числа в восьмеричное и шестнадцатеричное числа

Помимо обработки числовых данных и работы с разрядами, строковые потоки также позволяют организовать вывод не только в стандартные потоки

вывода и вывода, но также и в системные. В C++ предусмотрена библиотека для ведения системного журнала, в которой можно вести логирование и при этом не затрагивать «iostream» – библиотека «syslog» (также есть и другие).

«syslog» является заголовочным файлом стандартной библиотеки C++ и поставляется с остальным пакетом инструментов только на операционных системах семейства Linux. В Windows можно отдельно установить стороннюю библиотеку, которая реализует функционал ведения системного журнала.

В данный журнал можно выводить информацию о процессах, запускаемых пользователем во время работы пользователем, а также выводить информацию об ошибках и предупреждениях во время исполнения программного кода.

Главный недостаток «syslog» - объём функционала: в пользовании доступно всего 3 функции:

1. openlog() – нужна для начала вывода сообщений в системный журнал. На вход поступают идентификатор программы, настройки открываемого соединения и тип программы.

2. syslog() – нужна для вывода тех или иных сообщений на экран (всего есть 9 типов сообщений). На вход поступают приоритет (всего 9 типов) и формат вывода.

3. closelog() – нужна для прекращения вывода сообщений в системный журнал. На вход ничего не поступает.

В данной статье была рассмотрена библиотека «sstream» и методы оптимизации обработки строковых данных в приложениях и алгоритмах, написанных на языке программирования C++. Были рассмотрены основные функции библиотеки, а также их применение для решения задач при работе со строками.

Библиографический список

1. Kashmadze G. Fuzzyfication of the Bloch Ball //Computer Sciences and Telecommunications. 2017. №. 2. С. 30-36.
2. Шевчук А.В. и др. Разработка программы видеозахвата с экрана монитора персонального компьютера на базе операционной системы windows 10 // Научная мысль. 2019. Т.7. № 1 (31). С. 148-150.
3. Wu R. et al. SunwayMR: a distributed parallel computing framework with convenient data-intensive applications programming //Future Generation Computer Systems. 2017. Т. 71. С. 43-56.
4. std::basic_stringstream – cppreference URL: https://en.cppreference.com/w/cpp/io/basic_stringstream.
5. stringstream in C++ and its Applications - GeeksforGeeks URL: <https://www.geeksforgeeks.org/stringstream-c-applications/>
6. <sstream> | Microsoft Learn URL: <https://learn.microsoft.com/ru-ru/cpp/standard-library/ssstream?view=msvc-170>.

7. syslog, openlog, closelog, or setlogmask Subroutine - IBM URL:
<https://www.ibm.com/docs/en/aix/7.2?topic=s-syslog-openlog-closelog-setlogmask-subroutine>.