МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Физико-технический факультет

Кафедра информатики и вычислительной техники (ИВТ)

Лабораторная работа №07

**Программирование с использованием одномерных массивов**

Выполнил студент 585 гр.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.М. Губченко

Проверил: к.т.н,, доцент каф. ВТиЭ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.Г. Скурыдин

Лабораторная работа защищена

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Барнаул 2019

1. **Формулировка задачи**

Дана строка символов и некоторый символ sym. Сформируйте новую строку, вставив после каждого вхождения символа sym запятую.

1. **Постановка задачи**

На вход программы поступает буквенная строка и символ sym. После этого программа ставит запятую после каждого встреченного символа sym в этой строке. На выход поступает новая строка, полученная после работы программы.

1. **Математическая модель**

Для построения математической модели рассмотрим пример. Инициализируется строка «Hi, how are you doing with the session?» и символ «o». После этого программа проверяет каждый символ строки на совпадение с символом «о». Если встретится символ «о», то программа ставит после него запятую. В конце выполнения алгоритма программа выведет строку «Hi, ho,w are yo,u do,ing with the sessio,n?»

1. **Описание алгоритма**

**Начало алгоритма**

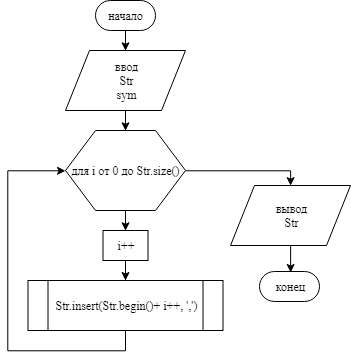
* 1. Ввод строки Str
  2. Ввод символа sym
  3. Присвоение переменной i значения 0
  4. Если i>=Str.size() переход к пункту 4.7
  5. Если (Str[i] != sym), то переход к пункту 4.4
  6. Выполнение Str.insert(Str.begin()+ i++, ',') и переход к пункту 4.4
  7. Вывод строки Str

**Конец алгоритма**

1. **Проект программы**

В данном случае программа должна состоять из 9 частей: 2 блока ввода, цикл с параметром[три блока арифметических действий, два блока проверки условий, стандартная функция insert()]

1. **Опорный граф (блок-схема) алгоритма**



1. **Текст программы:**

**на языке С++**

#include <iostream>

#include <string>

**using** **namespace** std;

**int** main()

{

**char** sym;

string Str;

cout << "enter the string:" << '\n';

getline(cin, Str);

cout << "enter character: ";

cin >> sym;

**for**(size\_t i=0; i<Str.size(); i++)

{

**if**(Str[i] == sym)

{

Str.insert(Str.begin()+ i++, ',');

}

}

cout<<Str<<'\n';

**return** 0;

}

1. **Проверка работоспособности (тестирование) программы**

Для проверки работоспособности программы необходимо просто посмотреть на то, что выведет программа в конце работы.

1. **Сравнительный анализ и оценка эффективности работы программ на разных языках программирования**

Ввиду простоты программы она одинаково эффективно выполняется для обоих языков программирования.

1. **Формулировка задачи**

Дана строка слов, разделённых пробелами и запятыми. Подсчитайте количество слов в строке и сформируйте новую строку из самых длинных слов подстрок (заключённых между запятыми)

1. **Постановка задачи**

На вход программы поступает буквенная строка слов, разделённых пробелами и запятыми. Программа посчитывает количество слов в строке и формирует новую строку из самых длинных слов подстрок, заключённых между запятыми. На выход поступает буквенная строка и натуральное вещественное число слов в исходной строке.

1. **Математическая модель**

Для построения математической модели рассмотрим пример. Инициализируется строка «Hi, how are you doing with the session». После этого программа проверяет каждый символ на то, является ли он пробелом или запятой, если не является, то она добавляет его в другую строку. Таким образом программа запоминает всё слово и его длину. Если встречается пробел, то полученная строка сохраняется, и длина данного слова считается максимальной. После этого программа считает длину нового слова. Если длина нового слова больше предыдущего, то оно становится максимальным. Если встречается запятая, то программа сравнивает длину найденного слова с максимумом и если это слово больше максимального, то оно выводится на экран, иначе выводится максимальное. После этого программа ищет самое длинное слово в новой подстроке. В данном случае программы выведет строку «Hi session» и число 8.

1. **Описание алгоритма**

**Начало алгоритма**

* 1. Присвоение переменной temp значения 0
  2. Присвоение переменной maxlen значения 0
  3. Присвоение переменной j значения 0
  4. Присвоение переменной word значения 0

// ввод строки Str

* 1. getline(cin, Str)
  2. Присвоение переменной i значения 0

//цикл по строке

* 1. Проверка условия, если i>=Str.size(), то переход к пункту 4.32
  2. i++

//если запятая

* 1. Проверка условия, если Str[i] != ‘,’ то переход к пункту 4.19
  2. Проверка условия, если temp<=maxlen, то переход к пункту 4.13
  3. Присвоение переменной maxlen значения temp
  4. Присвоение переменной MaxWord значения TempWord
  5. Вывод MaxWord
  6. MaxWord.clear()
  7. memset(TempWord, 0, sizeof(TempWord))
  8. Присвоение переменной temp значения 0
  9. Присвоение переменной j значения 0
  10. Присвоение переменной maxlen значения 0

//если это символ

* 1. Проверка условия, если Str[i]==’ ‘ и Str[i]==’,’ то переход к пункту 4.25
  2. Temp++
  3. Присвоение переменной TempWord[j] значения Str[i]
  4. j++
  5. Проверка условия, если Str[i+1]!=’ ‘ или Str[i+1]!=’,’ то переход к пункту 4.25
  6. word++

//если пробел

* 1. Проверка условия, если Str[i]!=’ ‘, то переход к пункту 4.7
  2. Проверка условия, если temp<=maxlen, то переход к пункту 4.29
  3. Присвоение переменной maxlen значения temp
  4. Присвоение переменной MaxWord значения TempWord
  5. memset(TempWord, 0, sizeof(TempWord))
  6. Присвоение переменной temp значения 0
  7. Присвоение переменной j значения 0

//конец выполнения

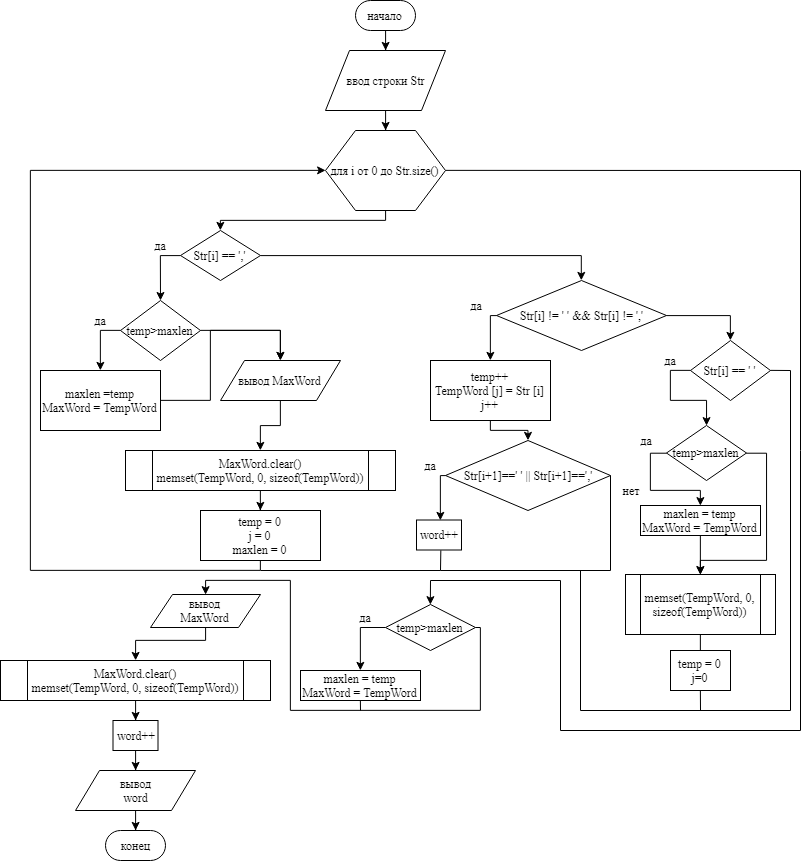
* 1. Проверка условия, если temp <= maxlen, то переход к пункту 4.35
  2. Присвоение переменной maxlen значения temp
  3. Присвоение переменной MaxWord значения TempWord
  4. Вывод MaxWord
  5. MaxWord.clear()
  6. memset(TempWord, 0, sizeof(TempWord))
  7. word++
  8. вывод word

**Конец алгоритма**

1. **Проект программы с подсчётом слов в строке, и выводом самых длинных слов подстрок**

В данном случае программа должна состоять из 53 частей: 6 блоков арифметических действий, два блока ввода цикл с параметром(7 блоков проверки условия, 15 блоков арифметических действий, один блок вывода, 3 стандартные функции), блок проверки условия, 3 блока арифметических действий, два блока вывода, две стандартные функции.

1. **Опорный граф (блок-схема)**



1. **Текст программы:**

**на языке С++**

#include <iostream>

#include <cstring>

**using** **namespace** std;

**int** main()

{

string Str; //исходная строка

// string TempWord; //текущее слово

**char** TempWord [20];

string MaxWord;//максим. слово

MaxWord [0] = '\0';

**char** ClearStr;

ClearStr = ' ';

**int** temp = 0;

**int** maxlen = 0;

**int** j=0;

**int** word = 0;

cout << "enter the string:" << '\n';

getline(cin, Str); //ввод строки

**for** (size\_t i=0; i<Str.size(); i++) //цикл по всей строке

{

**if** (Str[i] == ',') // если запятая, то

{

**if**(temp>maxlen) // если длина текущего слова больше максимальной длины, то

{

maxlen = temp; // обновление максимальной длины

MaxWord = TempWord; // обновление буфера максимальных слов

// cout << "MaxWord:" << MaxWord << '\n';

}

cout << MaxWord << " "; // вывод текущего макс.слова

MaxWord.clear(); //очистка буфера максим.слов

memset(TempWord, 0, **sizeof**(TempWord)); // очистка буфера текущего слова

//cout << "Temp clear" << '\n';

temp = 0;

j=0;

maxlen = 0;

}

**if** (Str[i] != ' ' && Str[i] != ',') // если это не пробел и не запятая, то

{

temp++; // кол-во букв в слове +1

TempWord [j] = Str [i]; // запись символа в буфер слов

//cout << "TempWord: " << TempWord << '\n';

j++;

**if** (Str[i+1]==' ' || Str[i+1]==',') {

word++;

}

}

**if** (Str[i] == ' ') // если это пробел, то

{

**if**(temp>maxlen) // если длина текущего слова больше максимальной длины, то

{

maxlen = temp; // обновление максимальной длины

MaxWord = TempWord; // обновление буфера максимальных слов

// cout << "MaxWord:" << MaxWord << '\n';

}

memset(TempWord, 0, **sizeof**(TempWord));

//cout << "Temp clear" << '\n';

temp = 0;

j=0;

}

}

**if**(temp>maxlen) // если длина текущего слова больше максимальной длины, то

{

maxlen = temp; // обновление максимальной длины

MaxWord = TempWord; // обновление буфера максимальных слов

//cout << "MaxWord:" << MaxWord << '\n';

}

cout << MaxWord << " "; // вывод текущего макс.слова

MaxWord.clear(); //очистка буфера максим.слов

memset(TempWord, 0, **sizeof**(TempWord)); // очистка буфера текущего слова

//cout << "Temp clear" << '\n';

word++;

cout << " " << '\n';

cout << "Word= " << word << '\n';

**return** 0;

}

1. **Проверка работоспособности (тестирование) программы**

Для проверки работоспособности программы необходимо посмотреть на то, что выведет программа.

1. **Сравнительный анализ и оценка эффективности работы программ на разных языках программирования**

Ввиду простоты программы она одинаково эффективно выполняется для обоих языков программирования.