**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР**

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Программирование»**

**Тема: Работа со строкой**

Студент гр. 2302 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Николаев В.Ю.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Калмычков В.А.

Санкт-Петербург

2023 г.

# Исходная формулировка

Заданная строка, возможно, состоит из слов, разделенных одним или несколькими пробелами. Преобразовать слово с заданным номером следующим образом: удалить из каждой группы подряд идущих цифр, которой не предшествует точка, все начальные нули (кроме последнего, если за ним идет точка).

Часть 1. Составить программу(-ы) для двух входных файлов: в первом располагается символ-маркер, символ-ограничитель и входная строка, во втором – символ-маркер, максимальное количество символов в строке и входная строка. Реализовать с использованием структуры.

Часть 2. Составить программу(-ы) для двух входных файлов: в первом располагается символ-ограничитель и входная строка, во втором – максимальное количество символов в строке и входная строка. Реализовать с использованием класса.

# Анализ задачи и устранение неясностей

Задана строка. В строку могут входить только заглавные и прописные буквы из латинского алфавита, цифры, пробелы, знаки препинания и прочие не служебные символы. Сперва необходимо найти слово с заданным номером. Затем необходимо удалить из каждой группы подряд идущих цифр, которой не предшествует точка, все начальные нули (кроме последнего, если за ним идет точка). Таким образом слово “00.” перейдёт в слово “0.”; “0” => “”; “00.00” => “00.00” и т.д.

# Математическая постановка задачи

Используемые условия:

C1: !fin.is\_open()

C2: !ScanSymbol(fin, string.marker)

C3: !ScanSymbol(fin, limiter)

C4: !ScanNumber(fin, number)

C5: fin >> c && c != '\n' && c != limiter && c != string.marker

C6: !ScanNumber(fin, limit)

C7: size < limit && fin >> c && c != '\n' && c != string.marker

С8: string.pA[0] != ' ' && string.pA[0] != string.marker

C9: number == wordNumber

C10: string.pA[i] != string.marker

C11: wordNumber == number

C12: beginWord == -1

C13: string.pA[i] == ' ' && string.pA[i + 1] != ' ' && string.pA[i + 1] != string.marker

C14: string.pA[i] != ' ' && string.pA[i] != string.marker

C15: ('1' <= string.pA[i] && string.pA[i] <= '9') || string.pA[i] == '.' || (string.pA[i] == '0' && startNumber)

C16: string.pA[i] == '0' && !startNumber && string.pA[i + 1] != '.'

C17: string.pA[j - 1] != string.marker

# Контрольные примеры

Строка: \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10

| Номер изменяемого слова | Полученная строка |
| --- | --- |
| 1 | \*\*0.sdfklaf\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 2 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 3 | \*0\*00000.sdfklaf000\* ab ab 0.10 00.001 00.00ab000.0 10 |
| 4 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 5 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 0.001 00.00ab000.0 10 |
| 6 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 0.00ab0.0 10 |
| 7 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |

# **Организация UI**

Макет O1:

Do you have a file with a limiter or number of symbols (enter number of answer)?

1) File with limiter;

2) File with number of symbols.\n

Макет I1:

l

Макет O2:

l isn't '1' or '2'. Enter '1' or '2'.\n

Макет E1:

Error. File \'input\' isn\'t open.\n

Макет E2:

Error. File \'input\' doesn\'t have a marker.\n

Макет E3:

Error. File \'input\' doesn\'t have a limiter.\n

Макет E4:

Error. File \'input\' doesn\'t have a processing word number.\n

Макет E5:

Error. File \'input\' doesn\'t have a number of a symbol.\n

Макет E6:

Error. There is no word number " << number << " in the sentence.\n

Макет E7:

Error. File \'output\' isn\'t open.\n

# Структуры данных

| struct StrMD  {  char \*pA;  char marker;  }; | class StrS  {  public:  char \*pA;  int size;  }; |
| --- | --- |
| \*pA - указатель на массив переменных типа char, в котором хранится сама строка;  marker - последний элемент этого массива, который гарантированно не встречается ранее в строке. | \*pA - указатель на массив переменных типа char, в котором хранится сама строка;  size - размер используемой части массива. |
|  |  |

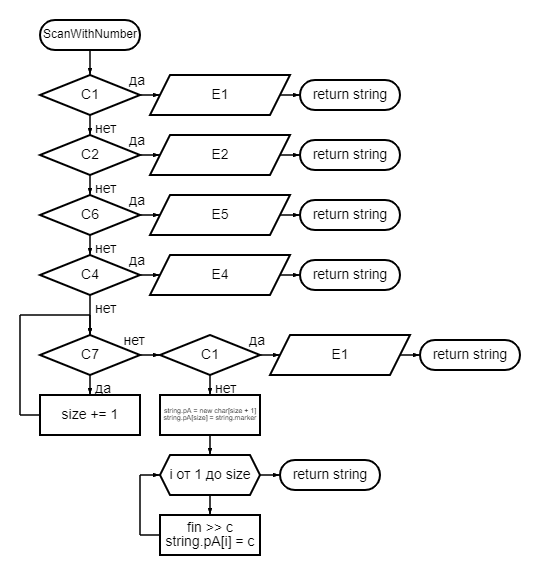
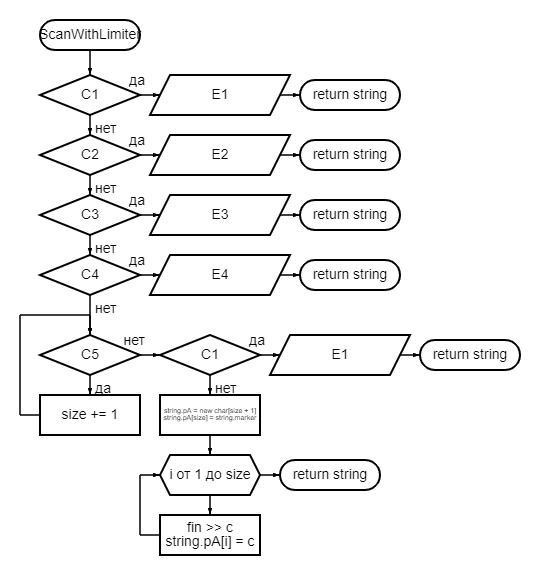
# **Способ хранения данных**

| Тип данных | Название переменной | Описание переменной |
| --- | --- | --- |
| int | number | Номер изменяемого слова |
| StrMD | string | Строка |
| fstream | fin | Файл input, из которого идёт ввод основных данных |
| int | size | Длина строки |
| int | beginWord | Номер первого символа в изменяемом слове |
| fstream | fout | Файл output, в который идёт вывод полученной строки |

# Описание функций

| Тип | Название | Передаваемые переменные | Описание переменных | Описание функции |
| --- | --- | --- | --- | --- |
| bool | ScanSymbol | fstream& fin | Поток ввода | Получает символ из потока ввода. Возвращает true, если получилось. Иначе false. |
| char& symbol | Получаемый символ |
| bool | ScanNumber | fstream& fin | Поток ввода | Получает число из потока ввода. Возвращает true, если получилось. Иначе false. |
| int& number | Получаемое число |
| StrMD | ScanWithLimiter | int& number | Номер изменяемого слова | Ввод строки с использованием ограничителя |
| StrMD | ScanWithNumber | int& number | Номер изменяемого слова | Ввод строки с использованием максимальной длины слова |
| StrMD | Scan | int& number | Номер изменяемого слова | Ввод строки |
| void | Processing | StrMD& string | Строка | Обработка строки |
| const int& number | Номер изменяемого слова |
| void | Print | const StrMD& string | Строка | Вывод строки в файл |

# **Представление алгоритма решения задачи**



# 

# 

# 

# 

# 

# 

# 

# 

# 

# Текст программы

| main1.cpp | main2.cpp |
| --- | --- |
| #include <iostream>  #include <fstream>  struct StrS  {  char \*pA;  char marker;  };  bool ScanSymbol(std::fstream& fin, char& symbol)  {  char c;  return bool(fin >> symbol) && bool(fin >> c) && c == '\n';  }  bool ScanNumber(std::fstream& fin, int& number)  {  char c;  for (; fin >> c && '0' <= c && c <= '9'; number = number \* 10 + c - '0');  return c == '\n';  }  StrS ScanWithLimiter(int& number)  {  StrS string;  std::fstream fin("input1", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input1\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  if (!ScanSymbol(fin, string.marker))  {  std::cout << "Error. File \'input1\' doesn\'t have a marker.\n";  fin.close();  return string;  }  char limiter;  if (!ScanSymbol(fin, limiter))  {  std::cout << "Error. File \'input1\' doesn\'t have a limiter.\n";  fin.close();  return string;  }  if (!ScanNumber(fin, number))  {  std::cout << "Error. File \'input1\' doesn\'t have a processing word number.\n";  fin.close();  return string;  }  int size = 0;  char c = ' ';  for (; fin >> c && c != '\n' && c != limiter && c != string.marker; size++);  fin.close();  fin.open("input1", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input1\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  string.pA = new char[size + 1];  string.pA[size] = string.marker;    for (int i = 0; i < 3 && fin >> c; i += c == '\n');  for (int i = -1; fin >> c && string.pA[++i] != string.marker; string.pA[i] = c);  fin.close();  return string;  }  StrS ScanWithNumber(int& number)  {  StrS string;  std::fstream fin("input1", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input1\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  if (!ScanSymbol(fin, string.marker))  {  std::cout << "Error. File \'input1\' doesn\'t have a marker.\n";  fin.close();  return string;  }  int limit = 0;  if (!ScanNumber(fin, limit))  {  std::cout << "Error. File \'input1\' doesn\'t have a number of a symbol.\n";  fin.close();  return string;  }  if (!ScanNumber(fin, number))  {  std::cout << "Error. File \'input1\' doesn\'t have a processing word number.\n";  fin.close();  return string;  }  int size = 0;  char c = ' ';  for (; size < limit && fin >> c && c != '\n' && c != string.marker; size++);  fin.close();  fin.open("input1", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input1\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  string.pA = new char[size + 1];  string.pA[size] = string.marker;    for (int i = 0; i < 3 && fin >> c; i += c == '\n');  for (int i = -1; fin >> c && string.pA[++i] != string.marker; string.pA[i] = c);  fin.close();  return string;  }  StrS Scan(int& number)  {  std::cout << "Do you have a file with a limiter or number of symbols (enter number of answer)?\n"  << " 1) File with limiter;\n"  << " 2) File with number of symbols.\n";    char c = ' ';  while (std::cin >> c && c != '1' && c != '2')  std::cout << "\'" << c << "\' isn\'t \'1\' or \'2\'. Enter \'1\' or \'2\'.\n";    if (c == '1')  return ScanWithLimiter(number);  return ScanWithNumber(number);  }  void Processing(StrS& string, const int& number)  {  int wordNumber = string.pA[0] != ' ' && string.pA[0] != string.marker, beginWord = number == wordNumber && number == 1 ? 0 : -1;  for (int i = -1; wordNumber != number && string.pA[++i] != string.marker && beginWord != i; beginWord = (wordNumber += string.pA[i] == ' ' && string.pA[i + 1] != ' ' && string.pA[i + 1] != string.marker) == number ? i + 1 : -1);  if (beginWord == -1)  {  std::cout << "Error. There is no word number " << number << " in the sentence.\n";  return;  }  bool startNumber = false;  for (int i = beginWord - 1; string.pA[++i] != ' ' && string.pA[i] != string.marker; startNumber = ('1' <= string.pA[i] && string.pA[i] <= '9') || string.pA[i] == '.' || (string.pA[i] == '0' && startNumber))  if (string.pA[i] == '0' && !startNumber && string.pA[i + 1] != '.')  for (int j = i--; string.pA[++j - 1] != string.marker; string.pA[j - 1] = string.pA[j]);  }  void Print(const StrS& string)  {  std::fstream fout("output", std::ios::out);  if (!fout.is\_open())  {  std::cout << "Error. File \'output\' isn\'t open.\n";  fout.close();  return;  }  for (int i = -1; string.pA[++i] != string.marker; fout << string.pA[i]);  fout << '\n';  fout.close();  }  int main()  {  int number = 0;  StrS string = Scan(number);  if (string.pA == nullptr)  return 0;  Processing(string, number);  Print(string);  delete string.pA;  return 0;  } | #include <iostream>  #include <fstream>  class StrS  {  public:  char \*pA;  int size;  };  bool ScanSymbol(std::fstream& fin, char& symbol)  {  char c;  return bool(fin >> symbol) && bool(fin >> c) && c == '\n';  }  bool ScanNumber(std::fstream& fin, int& number)  {  char c;  for (; fin >> c && '0' <= c && c <= '9'; number = number \* 10 + c - '0');  return c == '\n';  }  StrS ScanWithLimiter(int& number)  {  StrS string;  std::fstream fin("input2", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input2\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  char limiter;  if (!ScanSymbol(fin, limiter))  {  std::cout << "Error. File \'input2\' doesn\'t have a limiter.\n";  fin.close();  return string;  }  if (!ScanNumber(fin, number))  {  std::cout << "Error. File \'input2\' doesn\'t have a processing word number.\n";  fin.close();  return string;  }  int size = 0;  char c = ' ';  for (; fin >> c && c != '\n' && c != limiter; size++);  fin.close();  fin.open("input2", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input2\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  string.pA = new char[size + 1];  string.size = size;    for (int i = 0; i < 2 && fin >> c; i += c == '\n');  for (int i = -1; fin >> c && ++i < size; string.pA[i] = c);  fin.close();  return string;  }  StrS ScanWithNumber(int& number)  {  StrS string;  std::fstream fin("input2", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input2\' isn\'t open.\n";  return string;  }  fin << std::noskipws;    int limit = 0;  if (!ScanNumber(fin, limit))  {  std::cout << "Error. File \'input2\' doesn\'t have a number of a symbol.\n";  fin.close();  return string;  }  if (!ScanNumber(fin, number))  {  std::cout << "Error. File \'input2\' doesn\'t have a processing word number.\n";  fin.close();  return string;  }  int size = 0;  char c = ' ';  for (; size < limit && fin >> c && c != '\n'; size++);  fin.close();  fin.open("input2", std::ios::in);  if (!fin.is\_open())  {  std::cout << "Error. File \'input2\' isn\'t open.\n";  return string;  }  fin << std::noskipws;  string.pA = new char[size + 1];  string.size = size;    for (int i = 0; i < 2 && fin >> c; i += c == '\n');  for (int i = -1; fin >> c && ++i < size; string.pA[i] = c);  fin.close();  return string;  }  StrS Scan(int& number)  {  std::cout << "Do you have a file with a limiter or number of symbols (enter number of answer)?\n"  << " 1) File with limiter;\n"  << " 2) File with number of symbols.\n";    char c = ' ';  while (std::cin >> c && c != '1' && c != '2')  std::cout << "\'" << c << "\' isn\'t \'1\' or \'2\'. Enter \'1\' or \'2\'.\n";  if (c == '1')  return ScanWithLimiter(number);  return ScanWithNumber(number);  }  void Processing(StrS& string, const int& number)  {  int wordNumber = string.pA[0] != ' ' && string.size != 0, beginWord = number == wordNumber && number == 1 ? 0 : -1;  for (int i = -1; wordNumber != number && ++i != string.size && beginWord != i; beginWord = (wordNumber += string.pA[i] == ' ' && string.pA[i + 1] != ' ' && i + 1 != string.size) == number ? i + 1 : -1);  if (beginWord == -1)  {  std::cout << "Error. There is no word number " << number << " in the sentence.\n";  return;  }  bool startNumber = false;  for (int i = beginWord - 1; string.pA[++i] != ' ' && i != string.size; startNumber = ('1' <= string.pA[i] && string.pA[i] <= '9') || string.pA[i] == '.' || (string.pA[i] == '0' && startNumber))  if (string.pA[i] == '0' && !startNumber && string.pA[i + 1] != '.')  for (int j = i--; ++j - 1 != string.size; string.pA[j - 1] = string.pA[j]);  }  void Print(const StrS& string)  {  std::fstream fout("output", std::ios::out);  if (!fout.is\_open())  {  std::cout << "Error. File \'output\' isn\'t open.\n";  fout.close();  return;  }  for (int i = -1; ++i != string.size; fout << string.pA[i]);  fout << '\n';  fout.close();  }  int main()  {  int number = 0;  StrS string = Scan(number);  if (string.pA == nullptr)  return 0;  Processing(string, number);  Print(string);  delete string.pA;  return 0;  } |

# **Результаты работы программы**

Исходная строка: \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10

| Номер изменяемого слова | Полученная строка |
| --- | --- |
| 1 | \*\*0.sdfklaf\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 2 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 3 | \*0\*00000.sdfklaf000\* ab ab 0.10 00.001 00.00ab000.0 10 |
| 4 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |
| 5 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 0.001 00.00ab000.0 10 |
| 6 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 0.00ab0.0 10 |
| 7 | \*0\*00000.sdfklaf000\* ab 0ab 0.10 00.001 00.00ab000.0 10 |

Все некорректные ситуации программа обрабатывает корректно (сообщает о полученной ошибке).

# Выводы о проделанной работе

В этой лабораторной работе мы научились создавать строку и работать с ней.