УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите теоретические сведения к лабораторной работе, выполните практическое задание.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 08.05.2023.

Требования к отчету:

Отчет предоставляется преподавателю в электронном варианте и должен содержать:

- **название работы, постановку цели, вывод;**
- **ответы на контрольные вопросы, указанные преподавателем.**

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лабораторная работа № 22

Тема: «Строки в С++»

Цель: изучить принципы описания строки и работы ними в С++, функции для работы со строками.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. Описание строк

В языке С++ для удобной работы со строками есть класс string, для использования которого необходимо подключить заголовочный файл string.

Строка — последовательность (массив) символов. Если в выражении встречается одиночный символ, он должен быть заключен в *одинарные кавычки*. При использовании в выражениях строка заключается в *двойные кавычки.* Признаком конца строки является нулевой символ **\0**. В С++ строки можно описать с помощью массива символов (массив элементов типа **char**), в котором следует предусмотреть место для хранения признака конца строки.

Строки можно объявлять и одновременно присваивать им значения:

```
string S1, S2 = "Hello";
```

Строка S1 будет пустой, строка S2 будет состоять из 5 символов.

К отдельным символам строки можно обращаться по индексу, как к элементам массива или С-строк. Например S[0] - это первый символ строки.

Для того, чтобы узнать длину строки можно использовать метод size() строки. Например, последний символ строки S это S[S.size() - 1].

Конструкторы строк

Строки можно создавать с использованием следующих конструкторов: `string()` - конструктор по умолчанию (без параметров) создает пустую строку.

`string(string & S)` - копия строки `S`

`string(size_t n, char c)` - повторение символа `c` заданное число `n` раз.

`string(size_t c)` - строка из одного символа `c`.

`string(string & S, size_t start, size_t len)` - строка, содержащая не более, чем `len` символов данной строки `S`, начиная с символа номер `start`.

Конструкторы можно вызывать явно, например, так:

```
S += string(10, 'z');
```

В этом примере явно вызывается конструктор `string` для создания строки, состоящей из 10 символов `'z'`.

Неявно конструктор вызывается при объявлении строки с указанием дополнительных параметров. Например, так:

```
string S(10, 'z');
```

Подробней о конструкторах для строк читайте здесь.

Ввод-вывод строк

Строка выводится точно так же, как и числовые значения:

```
cout << S;
```

Для считывания строки можно использовать операцию ">>" для объекта cin:

```
cin >> S;
```

В этом случае считывается строка из непробельных символов, пропуская пробелы и концы строк. Это удобно для того, чтобы разбивать текст на слова, или чтобы читать данные до конца файла при помощи while (cin >> S).

Можно считывать строки до появления символа конца строки при помощи функции getline. Сам символ конца строки считывается из входного потока, но к строке не добавляется:

getline(cin S);

Арифметические операторы

Со строками можно выполнять следующие арифметические операции:

= - присваивание значения.

+= - добавление в конец строки другой строки или символа.

+ - конкатенация двух строк, конкатенация строки и символа.

==, != - посимвольное сравнение.

<, >, <=, >= - лексикографическое сравнение.

То есть можно скопировать содержимое одной строки в другую при помощи операции S1 = S2, сравнить две строки на равенство при помощи S1 == S2, сравнить строки в лексикографическом порядке при помощи S1 < S2, или сделать сложение (конкатенацию) двух строк в виде S = S1 + S2.

Методы строк

У строк есть разные методы, многие из них можно использовать несколькими разными способами (с разным набором параметров).

Рассмотрим эти методы подробней.

size

Метод size() возращает длину длину строки. Возвращаемое значение является беззнаковым типом (как и во всех случаях, когда функция возращает значение, равное длине строке или индексу элемента - эти значения беззнаковые). Поэтому нужно аккуратно выполнять операцию вычитания из значения, которое возвращает size(). Например, ошибочным будет запись цикла, перебирающего все символы строки, кроме последнего, в виде for (int i = 0; i < S.size() - 1; ++i).

Кроме того, у строк есть метод length(), который также возвращает длину строки.

resize

S.resize(n) - Изменяет длину строки, новая длина строки становится равна n. При этом строка может как уменьшится, так и увеличиться. Если вызвать в виде S.resize(n, c), где c - символ, то при увеличении длины строки добавляемые символы будут равны c.

clear

S.clear() - очищает строчку, строка становится пустой.

empty

S.empty() - возвращает true, если строка пуста, false - если непуста.

push_back

S.push_back(c) - добавляет в конец строки символ c, вызывается с одним параметром типа char.

append

Добавляет в конец строки несколько символов, другую строку или фрагмент другой строки. Имеет много способов вызова.

S.append(n, c) - добавляет в конец строки n одинаковых символов, равных c. n имеет целочисленный тип, c - char.

S.append(T) - добавляет в конец строки S содержимое строки T. T может быть объектом класса string или C-строкой.

S.append(T, pos, count) - добавляет в конец строки S символы строки T начиная с символа с индексом pos количеством count.

erase

S.erase(pos) - удаляет из строки S с символа с индексом pos и до конца строки.

S.erase(pos, count) - удаляет из строки S с символа с индексом pos количеством count или до конца строки, если pos + count > S.size().

insert

Вставляет в середину строки несколько символов, другую строку или фрагмент другой строки. Способы вызова аналогичны способам вызова метода append, только первым параметром является значение i - позиция, в которую вставляются символы. Первый вставленный символ будет иметь индекс i, а все символы, которые ранее имели индекс i и более сдвигаются вправо.

S.insert(i, n, c) - вставить n одинаковых символов, равных c. n имеет целочисленный тип, c - char.

S.insert(i, T) - вставить содержимое строки T. T может быть объектом класса string или C-строкой.

S.insert(i, T, pos, count) - вставить символы строки T начиная с символа с индексом pos количеством count.

substr

S.substr(pos) - возвращает подстроку данной строки начиная с символа с индексом pos и до конца строки.

S.substr(pos, count) - возвращает подстроку данной строки начиная с символа с индексом pos количеством count или до конца строки, если pos + count > S.size().

replace

Заменяет фрагмент строки на несколько равных символов, другую строку или фрагмент другой строки. Способы вызова аналогичны способам вызова метода append, только первыми двумя параметрами являются два числа: pos и count. Из данной строки удаляется count символов, начиная с символа pos, и на их место вставляются новые символы.

S.replace(pos, count, n, c) - вставить n одинаковых символов, равных c. n имеет целочисленный тип, c - char.

S.replace(pos, count, T) - вставить содержимое строки T. T может быть объектом класса string или C-строкой.

S.replace(pos, count, T, pos2, count2) - вставить символы строки T начиная с символа с индексом pos количеством count.

find

Ищет в данной строке первое вхождение другой строки str. Возвращается номер первого символа, начиная с которого далее идет подстрока, равная строке str. Если эта строка не найдена, то возвращается константа string::npos (которая равна -1, но при этом является беззнаковой, то есть на самом деле является большим безннаковым положительным числом).

Если задано значение pos, то поиск начинается с позиции pos, то есть возращаемое значение будет не меньше, чем pos. Если значение pos не указано, то считается, что оно равно 0 - поиск осуществляется с начала строки.

S.find(str, pos = 0) - искать первое входение строки str начиная с позиции pos. Если pos не задано - то начиная с начала строки S.

S.find(str, pos, n) - искать в данной строке подстроку, равную первым n символам строки str. Значение pos должно быть задано.

rfind

Ищет последнее вхождение подстроки ("правый" поиск). Способы вызова аналогичны способам вызова метода find.

find_first_of

Ищет в данной строке первое появление любого из символов данной строки str. Возвращается номер этого символа или значение string::npos.

Если задано значение pos, то поиск начинается с позиции pos, то есть возращаемое значение будет не меньше, чем pos. Если значение pos не указано, то считается, что оно равно 0 - поиск осуществляется с начала строки.

S.find_first_of(str, pos = 0) - искать первое входение любого символа строки str начиная с позиции pos. Если pos не задано - то начиная с начала строки S.

find_last_of

Ищет в данной строке последнее появление любого из символов данной строки str. Способы вызова и возвращаемое значение аналогичны методу find_first_of.

find_first_not_of

Ищет в данной строке первое появление символа, отличного от символов строки str. Способы вызова и возвращаемое значение аналогичны методу find_first_of.

find_last_not_of

Ищет в данной строке последнее появление символа, отличного от символов строки str. Способы вызова и возвращаемое значение аналогичны методу find_first_of.

c_str

Возвращает указать на область памяти, в которой хранятся символы строки, возвращает значение типа char*. Возвращаемое значение можно рассматривать как С-строку и использовать в функциях, которые должны получать на вход С-строку.

2. Операции над строками

Строку можно обрабатывать как массив символов, используя алгоритмы обработки массивов или с помощью специальных функций обработки строк, некоторые из которых приведены ниже. Для работы с этими строками необходимо подключить библиотеку **cstring**.

Для преобразования числа в строку можно воспользоваться функцией **sprintf** из библиотеки **stdio.h**.

Некоторые функции работы со строками:

Прототип функции	Описание функции
size_t strlen(const char *s)	вычисляет длину строки s в байтах.
char *strcat(char *dest, const char *scr)	присоединяет строку src в конец строки dest, полученная срока возвращается в качестве результата
char *strcpy(char *dest, const char *scr)	копирует строку scr в место памяти, на которое указывает dest

char strncat(char *dest, const char *dest, size_t maxlen)	присоединяет строку maxlen символов строки src в конец строки dest
char *strncpy(char *dest, const char *scr, size_t maxlen)	копирует maxlen символов строки src в место памяти, на которое указывает dest
int ctrcmp(const char *s1, const char *s2)	сравнивает две строки в лексикографическом порядке с учетом различия прописных и строчных букв, функция возвращает 0, если строки совпадают, возвращает — 1, если s1 располагается в упорядоченном по алфавиту порядке раньше, чем s2, и 1 — в противоположном случае.
int strncmp(const char *s1, const char *s2, size_t maxlen)	сравнивает maxlen символов двух строк в лексикографическом порядке, функция возвращает 0, если строки совпадают, возвращает — 1, если s1 располагается в упорядоченном по алфавиту порядке раньше, чем s2, и 1 — в противоположном случае.
double atof(const char *s)	преобразует строку в вещественное число, в случае неудачного преобразования возвращается число 0
long atol(const char *s)	преобразует строку в длинное целое число, в случае неудачного преобразования возвращается 0
char *strchr(const char *s, int c);	возвращает указатель на первое вхождение символа **c** в строку, на которую указывает **s**. Если символ **c** не найден, возвращается NULL
char *strupr(char *s)	преобразует символы строки, на которую указывает s, в символы верхнего регистра, после чего возвращает ее

Задание к лабораторной работе:

1. Ознакомиться с теоретическим материалом.

2. Проверить свою теоретическую подготовку по контрольным вопросам.

3. В соответствии с вариантом составить блок-схемы и программы для решения индивидуальной задачи.

4. Подготовить отчет по лабораторной работе.

Таблица 1.Задание

№ варианта	Задание	Текст
1	В тексте заменить все символы 'а' на символ 'в'.	Звуки вальса особенно волнуют того, кто не умеет танцевать.
2	Подсчитать количество символов в тексте, а также количество пропусков.	Не соблазняйтесь мыслью быть личным благотворителем рода человеческого.
3	Подсчитать количество символов 'а' в тексте.	Души ранимой и больной не помогает щит стальной

4	Изъять все символы 'а' из текста.	Человек без сомнений должен непременно вызывать сомнению окружающих.
5	Проверить баланс скобок в арифметическом выражении. Сообщить результат проверки.	S=2*X1+3*X2/ (X2+3*X1) / (X2+25*X1)
6	В тексте сделать удваивание символа 'а'.	Идеалы голодного желудка после сытного обеда переменчивы.
7	Зашифровать введенный из клавиатуры текст заменой исходных символов на символы с кодом, больше на три единицы. Провести дешифровку.	Успех лидера выносится на плечах бегущих по пятам.
8	Определить и вывести на экран номера позиций и количество повторений запрашиваемого символа в строке, введенного из клавиатуры.	Любитель постоянно опираться на плече своих друзей остается в конечном итоге без них.
9	В тексте заменить все символы 'а' на 'А'.	Творческая настойчивость что умеют танцевать чаще всего называется стилем.
10	Модифицировать текст таким образом, изменить регистр букв на большой.	Каждый несчастный воспринимается людьми через ощущение радости своего благополучия.
11	Подсчитать в тексте количество больших букв.	В звездной системе есть большие планеты, например : Сатурн, Юпитер, Уран, Нептун.
12	Подсчитать количество букв в тексте, не учитывая другие символы.	Все мы не хуже актеров умеем подбирать лицо к определенному случаю.
13	В тексте заменить все символы 'в' на символ 'а'.	Неудачно, что пролилось светло, истины часто оставляет несмываемые пятна.
14	В тексте заменить все пропуски на символ '_'.	Маленькая пуля медведя валит.
15	Изъять из текста все посторонние символы, должны остаться только буквы.	Старый ~|~ дурак исторически терпеть не; может / младшего умника[6].
16	Подсчитать количество букв 'у' в тексте.	Бездарные труженики искусства любят носить власяницу мстудента.
17	В тексте заменить все пропуски на точку с запятой.	Что попал под микроскоп не понимает своих размеров.
18	Определить сколько букв 'а' в тексте.	Кто умеет делает, а кто не умеет кричит и учит.
19	Определить позицию первого пропуска, и выписать первое слово.	У платежной ведомости Розумовский пожинал плоды своего ума и образованности.
20	Заменить символом '|' все пропуски.	В глазах волка светилась искренняя тоска и неукротимая любовь к баранине.

21	В арифметическом выражении подсчитать количество операций произведения.	S=2*X1+3*X2/ (X2+3*X1) / (X2+25*X1)
22	Определить сколько раз в тексте встречается буква 'м'.	Частокол холодного безразличия к другим относится в основном самому себе.
23	Заменить пропуски словом 'ПРОПУСК'	Музыкальное искусство всегда любило хвататься за дирижерскую палочку.
24	Найти букву 'к' в тексте и сообщить ее позицию.	Человек, влюбленный у самого себя, не оставляет места влюбленности для другого.
25	Изъять символ 'о' из текста, и сообщить сколько изъято символов.	В азбуке лица и внешнего вида не все одинаково грамотные.
26	Модифицировать текст таким образом, перевести все буквы в нижний регистр.	В звездной системе есть большие планеты, например : Сатурн, Юпитер, Уран, Нептун.
27	Изъять символ 'а' из текста, и сообщить сколько изъято символов.	Сколько качко не мудрствуй, а лебедем не будешь.
28	Заменить символ 'а' символами 'XXX'.	Труд здоровья не отбирает.
29	В арифметическом выражении подсчитать количество операций вычитания	S=2*X1+3*X2/ (X2+3*X1) / (X2+25*X1)
30	В арифметическом выражении подсчитать количество операций сложения	S=2*X1+3*X2/ (X2+3*X1) / (X2+25*X1)