**Строки и символы**

Сегодня мы изучаем строки – одну из самых полезных вещей в реальных задачах. Мы уже немного с ними работали — например, когда выводили конкретную строку в кавычках.

**Чтение строк**

Для начала вам нужно подключить библиотеку string. То, что в русском языке называется одним словом «строка», в английском может означать разное. «String» – это строка как последовательность символов, а «line» – последовательность символов, оканчивающаяся переводом строки (нечто похожее на наш «абзац»).

Решим такую задачу: пользователь вводит свое имя, а программа здоровается с ним. Полное решение будет записано так:

#include <iostream>

#include <string>

using namespace std;

int main()

{

    string s;

    cin >> s;

    cout << "Hello, " + s;

    return 0;

}

В этой программе мы воспользовались сложением строк – они просто склеились. Можно было сделать то же самое, просто выведя отдельно «Hello, » и переменную s.

При использовании cin чтение будет происходить по словам. Например, если нам понадобится считать два слова, это можно сделать, считав с помощью cin две переменные типа string. Слова могут быть разделены любым количеством пробелов, табуляций и переводом строк, но в переменных окажутся только непробельные символы.

Часто возникает необходимость считать строку (в понимании line) целиком, а не пословно. Для этого есть специальная функция getline(cin, s). Первый параметр в этой функции указывает на поток ввода (cin), а второй – на строку, в которую нужно считывать.

**Коды символов**

В компьютере всё, в том числе и строки, хранится в виде чисел (строка — набор чисел, которыми кодируются символы). Для хранения одного символа используется тип char (от слова character, символ).

Можно обращаться к отдельным символам строки, написав после её имени в квадратных скобках номер символа. Нумерация символов в строке начинается с нуля, так же как и в векторах. Узнать длину строки можно с помощью метода size.

Рассмотрим на программу, которая считывает строку, сохраняет первый (на самом деле нулевой) символ строки в переменную типа char и выводит его:

string s;

char c;

cin >> s;

c = s[0];

cout << c;

 Когда мы выводим переменную типа char, то выводится символ. Хотя на самом деле char – числовая переменная и обозначает номер символа в кодовой таблице. По аналогии с переводом вещественных чисел в целые мы можем сделать перевод из типа char в тип int, чтобы узнать код символа. Вывод кода символа выглядит так:

cout << (int) c;

Сам по себе код символа для нас не несет никакой пользы, но важно знать, что в кодовой таблице символы расположены по порядку. То есть код маленькой латинской буквы b на единицу больше кода буквы a и так далее. Где-то отдельно в этой таблице лежат цифры, также расположенные по порядку (0 идет перед 1).

**Выделение цифр числа**

Теперь приступим к решению большой задачи: получим из html-кода страницы информацию о курсах акций, чтобы заработать на их колебаниях кучу денег. Первым делом нам нужно выделить из строки только цифры. Мы будем считать, что в строке есть только одно целое число и его и нужно получить. Для решения этой задачи мы будем проходить по всей строке и, если символ – цифра, будем её печатать.

string s;

getline(cin, s);

for (auto c : s) {

    if (c >= '0' && c <= '9') {

        cout << c;

    }

}

В этой программе мы проходим по всем символам строки (так же, как мы проходили по всем элементам вектора). Еще мы научились узнавать код конкретного символа – для этого нужно записать этот символ в одинарных кавычках. Если код очередного символа лежит в пределах от 0 до 9, то этот символ – цифра.

**Поиск подстроки в строке**

Вернемся к нашей задаче по зарабатыванию денег. Пусть в скачанном нами файле содержится много строк, но нам интересна только та, где есть название компании, акциями которой мы хотим торговать. Например, это «Рога и копыта» с кодом на бирже rkpt. Дальше наша задача усложняется: среди N строк нужно найти ту, которая содержит подстроку rkpt (то есть где-то внутри строки встречается эта последовательность символов) и вывести число, записанное в этой строке. К счастью, у строк есть много методов, способных облегчить нашу жизнь. Например, метод find, который как раз предназначен для поиска подстроки в строке. Код решения выглядит так:

int n;

cin >> n;

string s;

getline(cin, s);

for (int i = 0; i < n; i++) {

    getline(cin, s);

    if (s.find("rkpt") != -1) {

        for (auto c : s) {

            if (c >= '0' && c <= '9') {

                cout << c;

            }

        }

    }

}

Метод find работает следующим образом: если подстрока нашлась, то она возвращает число, равное номеру символа, с которого началось первое вхождение подстроки в строку. А если подстроки не нашлось, то этот метод возвращает -1.

Обратите внимание на getline перед циклом. Он необходим, потому что после считывания числа в этой строке остается еще и символ перевода строки. Так что когда мы сделаем первый getline, то он считает пустую строку (ведь до перевода строки ничего не осталось).

**Изменение регистра символа**

Рассмотрим ещё один случай. Допустим, название компании может быть написано как большими, так и маленькими буквами или даже вперемешку. А значит нам нужно научиться определять регистр. Чтобы проверить, является ли символ в переменной заглавной буквой, достаточно написать следующее условие:

if (c >= 'A' && c <= 'Z')

Это возможно, потому что буквы расположены в алфавитном порядке. Теперь мы можем определить, какая по счёту буква алфавита нам встретилась:

int temp = c – 'A’;

Поскольку переменная на самом деле является числом, мы можем вычитать один символ из другого — это будет просто вычитание одного кода из другого. Теперь, чтобы получить в переменной *c* соответствующую маленькую букву, нужно к коду буквы «a» (маленькой) прибавить порядковый номер нужной буквы в алфавите:

c = 'a' + temp;

Ну и наконец, если мы хотим преобразовать все буквы в строке, то должны заранее создать пустую строку:

string s2 = "";

А затем добавлять к ней символы по одному.