

«Разработка, отладка и испытание алгоритмов и программ с использованием строк (символьных массивов)»

Для учащегося предназначены задания, соответствующих его номеру по списку группы (по журналу класса).

Строки и символы в C++

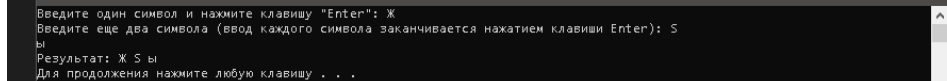
Для хранения отдельных символов используются переменные типа char.

При использовании классов ввод-вывод осуществляется как с помощью операции помещения в поток << и извлечения из потока >>, так и методов get() и get(char).

Ниже приведен пример применения операций:

```
#include <iostream>
#include <Windows.h> //библиотека для обеспечения поддержки русского языка (локализации)
using namespace std; //использование пространства имен std обеспечивает "краткий" синтаксис cin>>, cout<<, endl вместо std::cin>>, std::cout<<, std::endl

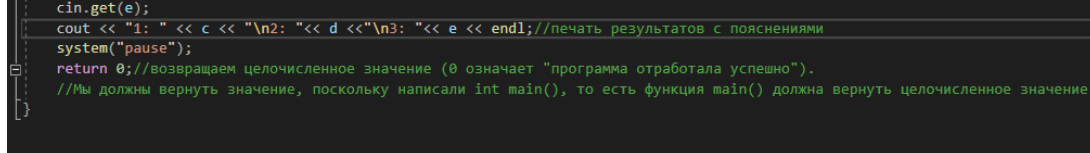
int main()
{
    SetConsoleOutputCP(1251); //установить кодовую страницу № 1251 (русский язык) для вывода из оперативной памяти на консоль
    SetConsoleCP(1251); //установить кодовую страницу № 1251 (русский язык) для ввода в оперативную память с клавиатуры
    char c, d, e;
    cout << "Введите один символ и нажмите клавишу \"Enter\": "; //если в печатаемой строке нужны кавычки как символ, то их можно экранировать обратным слэшем
    cin >> c; //проверьте на английских и русских символах (буквах)
    cout << "Введите еще два символа (ввод каждого символа заканчивается нажатием клавиши \"Enter\"): "; //два подряд расположенных спецсимвола становятся одним "обычным" символом
    cin >> d >> e;
    cout << "Результат: " << c << ' ' << d << ' ' << e << endl;
    system("pause");
    return 0; //возвращаем целочисленное значение (0 означает "программа отработала успешно").
    //Мы должны вернуть значение, поскольку написали int main(), то есть функция main() должна вернуть целочисленное значение
}
```



Для ввода любого символа включая пробел можно воспользоваться методами get() или get(char x).

```
#include <iostream>
#include <Windows.h> //библиотека для обеспечения поддержки русского языка (локализации)
using namespace std; //использование пространства имен std обеспечивает "краткий" синтаксис cin>>, cout<<, endl вместо std::cin>>, std::cout<<, std::endl

int main()
{
    SetConsoleOutputCP(1251); //установить кодовую страницу № 1251 (русский язык) для вывода из оперативной памяти на консоль
    SetConsoleCP(1251); //установить кодовую страницу № 1251 (русский язык) для ввода в оперативную память с клавиатуры
    char c, d, e;
    cout << "1) Введите один символ: ";
    c = cin.get(); //функция get() из потока(класса) ввода cin возвращает символ, который нужно поместить в символьную переменную
    cin.ignore(); //функция ignore() из потока ввода cin "чистит" поток от последнего введенного перед этим символа (нажатия Enter), который может помешать последующей функции
    cout << "2) Введите один символ: "; // считывания символов с клавиатуры (последний ранее введенный Enter заканчивает "новый" ввод, который не успевает начаться)
    cin.get(d); //перегруженная функция get() из потока ввода cin считывает вводимый с клавиатуры символ и помещает его в символьную переменную d
    cin.ignore(); //функция чистит буфер от последнего нажатого символа (Enter'a). Закомментируйте эти функции и посмотрите на изменения в работе программы
    cout << "3) Введите один символ: ";
    cin.get(e);
    cout << "1: " << c << "\n2: " << d << "\n3: " << e << endl; //печать результатов с пояснениями
    system("pause");
    return 0; //возвращаем целочисленное значение (0 означает "программа отработала успешно").
    //Мы должны вернуть значение, поскольку написали int main(), то есть функция main() должна вернуть целочисленное значение
}
```



Строка в C++ – это массив символов, заканчивающийся нуль-символом – ‘\0’ (нуль-терминатором). По положению нуль-терминатора определяется фактическая длина строки. Количество элементов в таком массиве на 1 больше, чем изображение строки.

A\0	A
"A" строка (2 байта)	'A' символ (1 байт)

Рисунок – Представление строки и символа

Присвоить значение строке с помощью оператора присваивания нельзя. Поместить строку в массив можно либо при вводе, либо с помощью инициализации.

Для работы со строками существуют специальные библиотечные функции, которые содержатся в заголовочном файле string.h.

Строки, при передаче в функцию, в качестве фактических параметров могут быть определены либо как одномерные массивы типа char[], либо как указатели типа char*. В отличие от обычных массивов в этом случае нет необходимости явно указывать длину строки.

Операции со строками

Для строк не определена операция присваивания. Присваивание выполняется с помощью функций стандартной библиотеки или посимвольно «вручную» (что менее предпочтительно, так как чревато ошибками).

Присвоить значение строке с помощью оператора присваивания нельзя. Поместить строку в массив можно либо при вводе, либо с помощью инициализации.

```
char s1[10]="string1";//инициализация
```

```
char s2[]="string2";//инициализация
```

```
char s3[10];
```

```

cin >> s3;//ввод
//выделение памяти под динамическую строку
char* s4 = new char[strlen(s3) + 1];
strcpy(s4, s3);//копирование строки s3 в строку s4
const int len_str = 80;
char str[len_str];

```

Для работы со строками существуют специальные библиотечные функции, которые содержатся в заголовочном файле string.h.

Таблица – Библиотека обработки символов

Прототип		Описание функции	
Сигнатура		Сигнатура	
int	isdigit	(int c)	Возвращает true, если входной элемент c является цифрой, в противном случае — false (0)
int	isalpha	(int c)	Возвращает true, если входной элемент c является буквой, в противном случае — false (0)

Таблица – Функции преобразования строк

Прототип	Описание функции
double atof (const char* nPtr)	Преобразует строку nPtr в число типа double и возвращает это число
int atoi (const char* nPtr)	Преобразует строку nPtr в число типа int и возвращает это число
long atol (const char* nPtr)	Преобразует строку nPtr в число типа long int и возвращает это число


Ниже пример использования функции atoi() для преобразования строки в число типа int.

```

#include <iostream>
#include <stdlib.h>//содержит прототип функции atoi(const char*);//работает и без явного подключения данной библиотеки
using namespace std;

void main()
{
    char s[] = "2593";//создадим строку символов, в которую поместим фразу "2593" (это символы)
    int i = atoi(s);//преобразуем строку символов s в целое число, которое поместим в переменную i
    cout << "The string \" << s << "\" changes into int = " << i << "\nThis value - 593 = " << i - 593 << endl;
    //распечатаем содержимое строки s и значение, которое помещено в i. i - переменная с числом, а потому с ним можно делать вычисления
    system("pause");
    return;//ничего не возвращаем, поскольку выше написали void main(), заявив тем самым, что main() ничего не должен возвращать
}

```



Для работы со строками существует специальная библиотека string.h. Примеры прототипов функций для работы со строками из библиотеки string.h:

Прототип функции	Описание назначения функции
unsigned strlen (const char* str);	Вычисляет длину строки str без учета завершающего ноль-терминатора ('\0') и возвращает эту длину (число символов в строке)
int strcmp (const char* str1, const char* str2);	Сравнивает строки str1 и str2 посимвольно слева направо до первого встреченного различающегося символа. Если str1 < str2, то возвращает результат отрицательный (отрицательное число), если str1 == str2, то результат равен 0, если str1 > str2, то результат положительный.
int strncmp (const char* str1, const char* str2, int n);	Сравнивает n первых символов строк str1 и str2 посимвольно слева направо до первого встреченного различающегося символа. Если str1 < str2, то возвращает результат отрицательный (отрицательное число), если str1 == str2, то результат равен 0, если str1 > str2, то результат положительный.
char* strcpy (char* s1, const char* s2);	Копирует байты (все символы) из строки s2 в строку s1 (старое содержимое строки s1 удаляется) и возвращает указатель на начало строки s1 с помещенным в нее содержимым из строки s2. Удостоверьтесь, что размер строки s1 достаточен для помещения в него всех символов из строки s2 и завершающего ноль-терминатора

char* strncpy (char* s1, const char* s2, int n);	Копирует n первых символов строки s2 в строку s1. Удостоверьтесь, что размер строки s1 достаточен для помещения в него n первых символов из строки s2 и завершающего ноль-терминатора
char* strcat (char* s1, const char* s2);	В строке s1 удаляет ноль-терминатор и к концу строки s1 дописывает символы из строки s2 и завершающий ноль-терминатор в конце. Возвращает указатель на начало дописанной строки s1
char* strncat (char* s1, const char* s2, int n);	В строке s1 удаляет ноль-терминатор и к концу строки s1 дописывает n первых символов из строки s2 и завершающий ноль-терминатор в конце. Возвращает указатель на начало дописанной строки s1
char* strchr (char* s, int c);	Ищет первый совпадающий с символом c символ в строке s и возвращает указатель на него или NULL если совпадения не нашло
char* strrchr (char* s, int c);	Ищет последний совпадающий с символом c символ в строке s и возвращает указатель на него или NULL если совпадения не нашло
char* strstr (char* s1, char* s2);	Ищет подстроку s2 в строке s1 и возвращает указатель на начало найденной подстроки s2 в строке s1 или NULL если совпадения не нашло
char* strtok (char* s1, char* s2);	Дробит строку s1 в местах, где находятся символы-разделители из символьного массива s2 и возвращает указатель на первый отрезанный участок или NULL если отрезать уже нечего (не находит больше символов-разделителей)
char* strdup (const char* str);	Выделяет память чтобы перенести в нее копию строки str и возвращает указатель на созданную копийную строку
char* strnset (char* s, int c, int n);	Заменяет первые n символов строки s символом c

Строки, при передаче в функцию, в качестве фактических параметров могут быть определены либо как одномерные массивы типа char[], либо как указатели типа char*. В отличие от обычных массивов в этом случае нет необходимости явно указывать длину строки, поскольку функция strlen(s) возвращает фактическую длину строки s, не учитывая завершающий ноль-терминатор.


Ввод-вывод строк

Библиотека <stdio.h> содержит функции, специально предназначенные для ввода-вывода строк: gets_s() и puts().

В нижеследующем примере пользователь вводит фразу с пробелами «I am man.». С учетом точки и пробелов фраза занимает 9 символов, а десятый фактически зарезервирован под ноль-терминатор. Можно ввести фразу и из меньшего количества символов, но обязательно **не больше** 9 символов.

```
#include <iostream>
#include <stdio.h> //подключить библиотеку с функциями ввода-вывода gets_s, puts//работает и без нее
using namespace std;

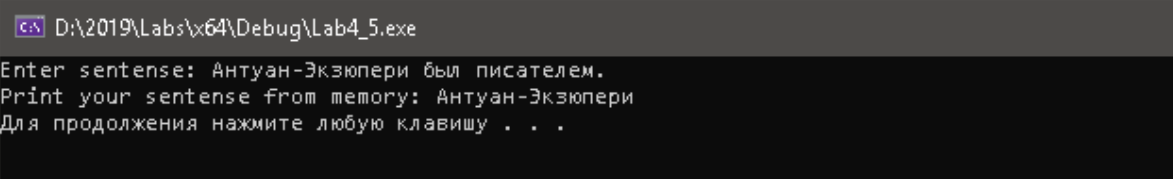
int main()
{
    const int n = 10;
    char s[n]; //строка символов размером n символов, в которую можно поместить n-1 символов, т.к. один будет занят ноль-терминатором '\0'
    gets_s(s, sizeof(s)); //можно считать с клавиатуры 9 символов и поместить в строку s. Функция безопасная в плане взлома программы
    puts(s); //функция печати на консоль содержимого строки s
    system("pause");
    return 0;
}
```



Пример в нотации C++:

```
#include <iostream>
#include <stdio.h> //подключить библиотеку с функциями ввода-вывода gets_s, puts //работает и без нее
using namespace std;

int main()
{
    const int n = 80;
    char s[n]; //создадим строку для хранения максимум 79 символов и ноль-терминатора
    cout << "Enter sentence: ";
    cin >> s; //попробуйте ввести строку с пробелами
    cout << "Print your sentence from memory: ";
    cout << s << endl; //посмотрим, что сохранилось в строке s на самом деле
    system("pause");
    return 0;
}
```

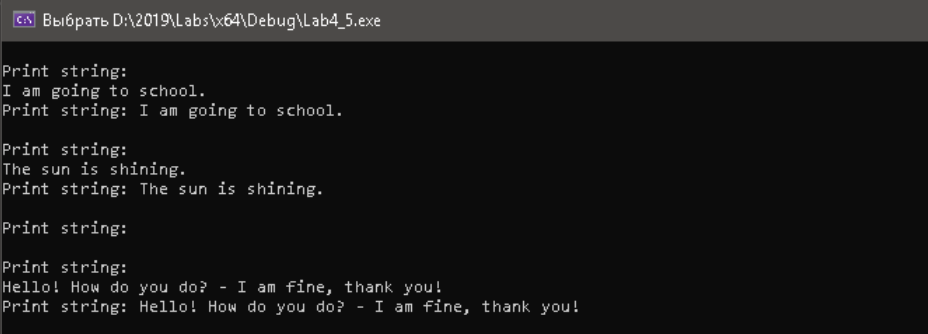


cin>> выполняет ввод с клавиатуры ДО первого пробельного символа (то есть пробела, знака табуляции или символа перевода строки '\n').

Если требуется ввести строку, состоящую из нескольких слов, разделенных пробелами, в одну строковую переменную, используются методы `cin.getline(массивКудаПоместитьФразу, максимальноеКоличествоСимволов)` и `get()` класса-потока `istream`, объектом которого является `cin`.

```
#include <iostream>
using namespace std;

int main()
{
    const int n = 80;
    char s[n];
    while (cin.getline(s, n)) //пока вводятся строки с клавиатуры, сохранять их в строку s, доступный для заполнения размер которой n-1
    { //цикл бесконечный
        cout << "Print string: " << s << endl;
        //здесь может быть код по обработке строки
    }
    system("pause");
    return 0;
}
```



Метод `getline` считывает из входного потока до $(n - 1)$ символов или менее, если символ перевода строки встретился раньше, и записывает их в строковую переменную `s`. Символ перевода строки также считывается (удаляется) из входного потока. Вместо символа перевода строки в строковой переменной размещается завершающий ноль-терминатор.

Если есть подозрения, что из входного потока не удален последний введенный символ перевода строки (нажатие клавиши `Enter`), то для корректной работы дальнейших выражений по считыванию строк кода используйте `cin.ignore()`; перед ними.

Метод `get` работает аналогично, но оставляет в потоке символ перевода строки. В строковую переменную добавляется завершающий ноль-терминатор.

Типичная ошибка: нельзя обращаться к методу `get()` с двумя аргументами два раза подряд, не удалив `'\n'` из входного потока.

После ввода первой строки метод `get()` «встретит» символ `'\n'`, оставленный во входном потоке после первого вызова этого метода. В результате на экран будут выведены еще три пустые строки, а символ `'\n'` так и останется во входном потоке.

Этот символ надо удалить из потока путем вызова `get()` без параметров или `cin.ignore()`;

Пример.

Можно следующим образом записать цикл, в котором печатаются целые значения кодов прописных (маленьких) букв английского алфавита:

```

C:\D>D:\2019\labi\64\Debug\Lab4_5.exe

Size: 26
char      dec      oct      hex
'a' =    97      = 0141 = 0x61
'b' =    98      = 0142 = 0x62
'c' =    99      = 0143 = 0x63
'd' =   100      = 0144 = 0x64
'e' =   101      = 0145 = 0x65
'f' =   102      = 0146 = 0x66
'g' =   103      = 0147 = 0x67
'h' =   104      = 0150 = 0x68
'i' =   105      = 0151 = 0x69
'j' =   106      = 0152 = 0x6a
'k' =   107      = 0153 = 0x6b
'l' =   108      = 0154 = 0x6c
'm' =   109      = 0155 = 0x6d
'n' =   110      = 0156 = 0x6e
'o' =   111      = 0157 = 0x6f
'p' =   112      = 0160 = 0x70
'q' =   113      = 0161 = 0x71
'r' =   114      = 0162 = 0x72
's' =   115      = 0163 = 0x73
't' =   116      = 0164 = 0x74
'u' =   117      = 0165 = 0x75
'v' =   118      = 0166 = 0x76
'w' =   119      = 0167 = 0x77
'x' =   120      = 0170 = 0x78
'y' =   121      = 0171 = 0x79
'z' =   122      = 0172 = 0x7a

Для продолжения нажмите любую клавишу . . .

```

Обе функции описаны в <xiosbase>. Для подсчета числа символов в alpha используется функция strlen() из <string.h>, но вместо нее можно было использовать размер массива alpha (26 элементов – число 26). Для множества символов ASCII результат будет таким:

```
'a' = 97 = 0141 = 0x61
'b' = 98 = 0142 = 0x62
'c' = 99 = 0143 = 0x63
```

```
//#define _CRT_SECURE_NO_WARNINGS//пишется в ПЕРВОЙ строке программы, чтобы работали "небезопасные" в плане взлома программы функции
#include <iostream>
using namespace std;

int main()
{
    char v[9];
    //v = "a string";// ошибка
    strcpy_s(v, "a string");//можно помещать значение в char'овский массив с помощью функций
    cout << v << endl;//Функции с окончанием _s являются безопасными в плане взлома программы, поскольку они учитывают (требуют) размер строки
    system("pause");//если требуется использовать "небезопасные" функции, то в самом вверху программы дописать #define _CRT_SECURE_NO_WARNINGS
    return 0;
}
```

D:\2019\Labs\64\Debug\Lab4_5.exe

a string
Для продолжения нажмите любую клавишу . . .

Очевидно, что строки пригодны только для инициализации символьных массивов; для других типов приходится использовать более сложную запись. Впрочем, она может использоваться и для символьных массивов.

Здесь v3 и v4 - массивы из четырех (а не пяти) символов; v4 не оканчивается нулевым символом, как того требует соглашение о строках и большинство библиотечных функций. Используя такой массив из элементов char, мы готовим почву для будущих ошибок.

Длина массива должна учитывать '\0'.

```
char s[5];
```

```
s[0] = 's';
s[1] = 'h';
s[2] = 'i';
s[3] = 'p';
s[4] = '\0';
char s[] = {'s', 'h', 'i', 'p', '\0'}; // можно не задавать размер при инициализации строки сразу при ее декларации
s = "ship"; // в этом случае ноль-символ будет добавлен автоматически
```

Функции для работы со строками

Алфавит обрабатывается английский, остальные цифры и символы – для всех алфавитов (запятая, пробел и т.д.)

Библиотека <ctype.h> или <cctype> содержит:

```
int isalnum(int ch); //вернет результат >=1 если ch буква или цифра
int isalpha(int ch); // вернет результат >=1 если ch буква
int isdigit(int ch); // вернет результат >=1 если ch цифра
int isgraph(int ch); // вернет результат >=1 если ch является печатаемым (видимым) символом, НО не пробелом
int islower(int ch); // вернет результат >=1 если ch является строчной («маленькой») буквой
int isprint(int ch); // вернет результат >=1 если ch является печатаемым символом, включая пробел
int ispunct(int ch); // вернет результат >=1 если ch является знаком пунктуации
int isspace(int ch); // вернет результат >=1 если ch является пробельным символом
int isupper(int ch); // вернет результат >=1 если ch является прописной («большой») буквой
int isxdigit(int ch); // вернет результат >=1 если ch является шестнадцатеричной цифрой
int tolower(int ch); //преобразует букву ch в нижний регистр и вернет строчный эквивалент принятой буквы
```

английского алфавита (маленькую букву, нижний регистр)

```
int toupper(int ch); //преобразует букву ch в верхний регистр и вернет прописной эквивалент принятой буквы
английского алфавита (большую букву, верхний регистр)
```

Библиотека <string.h> или <string>

```
int i;
i = strlen(str);
i = strcmp(str1, str2); //сравнит и вернет 0 если строки == (равны); <0 если str1 < str2; иначе >0
i = strncmp(str1, str2, count); //сравнит не более count символов
i = strcspn(str1, str2); //индекс первого символа в строке str1, который совпадает с любым из символов в строке
```

str2

```
i = strspn(str1, str2); // -// - который не совпадает
strcat(str1, str2); // присоединяет копию строки str2 к строке str1
strncat(str1, str2, count); // -// - count символов
strcpy(str1, str2); //копирует содержимое строки str2 в строку str1
strncpy(str1, str2, count); // -// - count символов
sprintf(str2, "****s****", str1); // форматирование строки в строку
char ch;
ch = getchar(); //для получения символа от стандартного ввода,
ch = getc(in); //для получения символа из файла,
putchar(ch); //выводит символ на стандартный вывод
putc(ch, f1); //
fscanf(fi, "%d", &age);
fprintf(fi, "Data is %d.\n", age);
puts(string);
fputs("Строка", fi);
```

В С++ поток представляет собой объект некоторого класса.

Класс istream

Класс istream выполняет действия по вводу данных – извлечению данных, содержит следующие функции:

>> - форматированное извлечение данных всех основных (и перегружаемых) типов из потока;

Для неформатированного чтения из потока:

get(str) – извлекает символы в символьный массив str до ограничителя '\n';

getline(str, MAX, DELIM) – извлекает в символьный массив str до MAX символов или до символа DELIM в

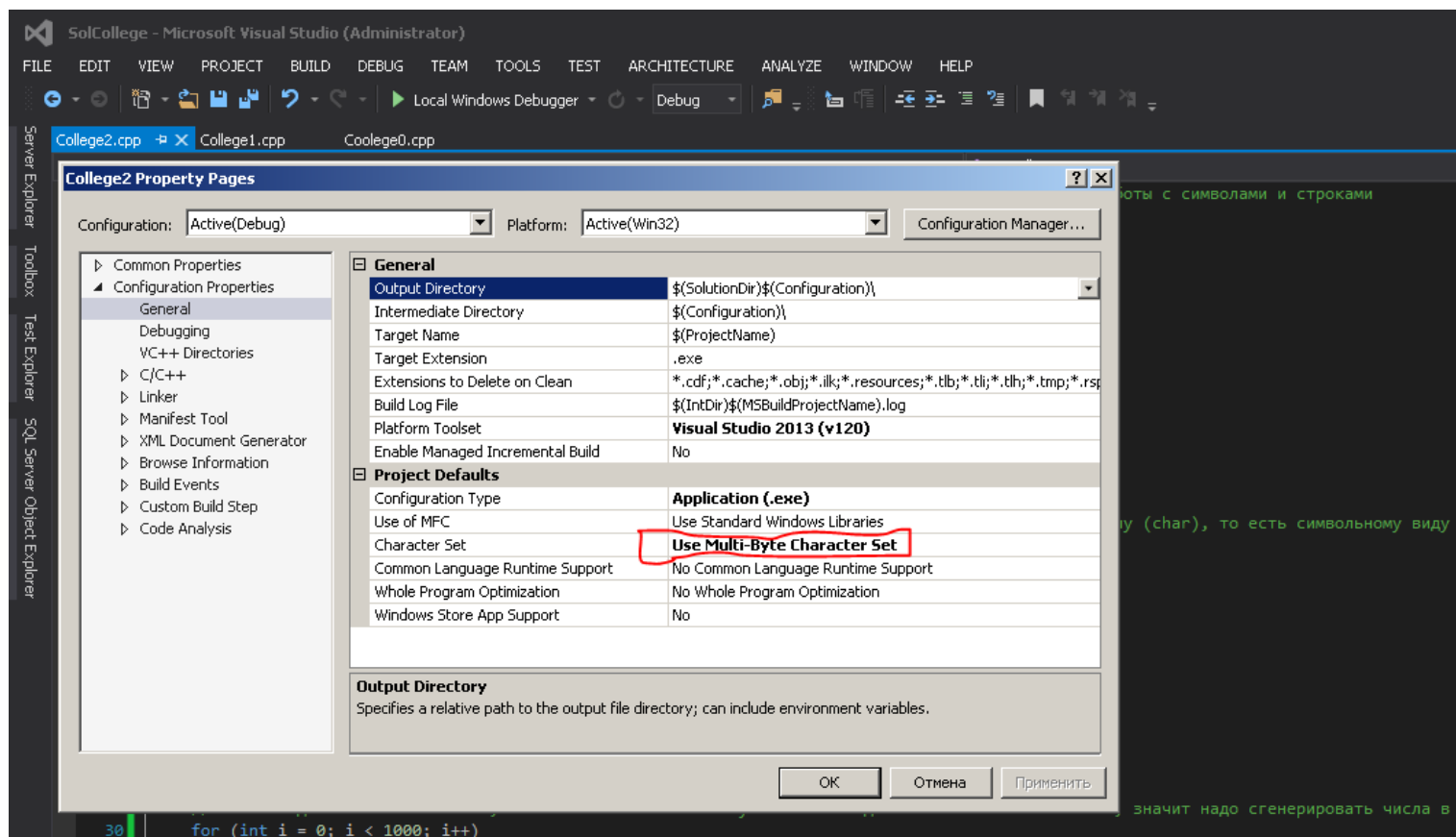
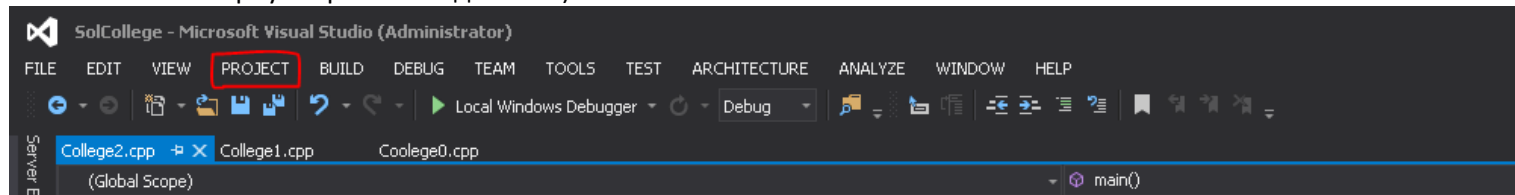
зависимости от того, что произойдет раньше; извлекает ограничитель из потока.

Задание № 1:

Набрать и протестировать работу прилагаемого образца кода (всем учащимся независимо от варианта). Можно исправлять, дописывать код.

Настройки в среде разработки (например, MS Visual Studio) могут влиять на работу создаваемого приложения. В моем случае, в меню Project/Properties/Configuration properties/General установлена MBCS-настройка (то есть многобайтовая кодировка символов, а не однобайтовая). Тут можно посмотреть настройки вашей среды разработки и,

при необходимости, поменять их (перед изменением лучше записать (сфотографировать) старые настройки, чтобы можно было их вернуть при необходимости).



В VS2019 аналогично:

Конфигурация: Активная (Debug)

Платформа: Активная (x64)

Диспетчер конфигураций...

Свойства конфигурации

Общие

Дополнительно

Отладка

Каталоги VC++

C/C++

Компоновщик

Инструмент манифеста

Генератор XML-документ

Информация об исходно

События сборки

Настраиваемый этап сб

Анализ кода

Расширенные свойства

Расширение целевого файла .exe

Расширения для удаления при очистке *.cdf;*.cache;*.obj;*.obj.enc;*.ilk;*.ipdb;*.iobj;*.resources;*.tlb;*.tli;*.t

Файл журнала сборки \$(IntDir)\$(MSBuildProjectName).log

Архитектура предпочитаемых средств сб По умолчанию

Использовать отладочные библиотеки Да

Включить сборку Unity (JUMBO) Нет

Использование MFC Использовать стандартные библиотеки Windows

Набор символов Использовать многобайтовую кодировку

Оптимизация всей программы Без оптимизации всей программы

Версия набора инструментов MSVC По умолчанию

Свойства C++/CLI

Поддержка общезыковой среды выполн Без поддержки CLR-среды

Целевая версия платформы .NET Framework

Разрешить управляемую добавочную сб Нет

Набор символов

Указывает, какую кодировку следует использовать компилятору; актуально при локализации.

OK

Отмена

Применить

```

1 #define _CRT_SECURE_NO_WARNINGS//директива препроцессору: разрешить использовать "старые" функции работы с символами и строками
2 #include <iostream>
3 #include <string.h>//библиотека обработки строк
4 #include <time.h>
5 #include <stdlib.h>
6 #include <ctype.h>//библиотека обработки символов
7 #include <math.h>
8 using namespace std;
9
10 int main()
11 {
12     setlocale(LC_ALL, "Russian");
13     cout << "Печать кодов ASCII и символов:\n";
14     for (unsigned short int a = 0; a < 256; a++)
15     {
16         cout << a << '\t' << (char)a << endl;//(char)a - явное приведение числовой переменной a к типу (char), то есть символному виду
17     }
18
19     cout << "\nГенерация случайных символов:\n";
20     srand(time(NULL));
21     char mas[1000];//создать массив для хранения 1000 символов
22     for (int i = 0; i < 1000; i++)
23     {
24         mas[i] = (char)(rand() % 256);//заполнить каждый элемент массива случайным символом
25         cout << mas[i] << ' ';//распечатать значение, хранимое в каждом элементе массива
26     }
27
28     cout << "\n\nГенерация ЗАГЛАВНЫХ английских букв:\n";
29     //печать кодов символов показал, что большие английские буквы имеют коды с 65 по 90 включительно, значит надо сгенерировать числа в таком диапазоне
30     for (int i = 0; i < 1000; i++)
31     {
32         //для хранения больших английских букв используем "старый" уже заполненный массив mas[1000], перезаписывая в нем прежние значения на новые
33         mas[i] = (char)(65 + rand() % (90-65+1));//заполнить каждый элемент массива случайной буквой с кодом в диапазоне от 65 до 90 включительно
34         cout << mas[i] << ' ';//распечатать значение, хранимое в каждом элементе массива
35     }
36
37     cout << "\n\nГенерация текста из английских букв:\n";//сгенерируем текст, похожий на английский, пусть и без смысла
38     //печать кодов символов показал, что маленькие английские буквы имеют коды с 97 по 122 включительно, значит надо сгенерировать числа в таком диапазоне
39     mas[0] = '\t';//первый символ будет красной строкой (то есть табуляционным отступом)
40     mas[1] = (char)(65 + rand() % 26);//за красной строкой должна быть только большая буква
41     for (int i = 2; i < 999; i++)//далее буквы произвольны, кроме последней (это будет символ точка)

```



```

41 {
42     if (rand() % 17 == 0 & i != 2) //большие буквы встречаются реже маленьких, зададим частоту 1 к 17. За первой большой буквой должна стоять маленькая
43     {
44         mas[i] = (char)(65 + rand() % 26); //если i-тая буква большая
45         mas[i - 1] = ' '; //то перед ней должен быть пробел
46         mas[i - 2] = '.'; //а перед пробелом пусть завершается точкой предыдущее предложение
47     }
48     else //маленькие буквы будут генериться в 16 случаях из 17-и
49     {
50         if (rand() % 7 == 0 & mas[i-1] != ' ') //из них пусть каждый седьмой символ будет пробел - разделитель слов в предложении
51         {
52             mas[i] = ' ';
53         }
54         else
55         {
56             mas[i] = (char)(97 + rand() % (122 - 97 + 1));
57         }
58     }
59 }
60 mas[999] = '.'; //самый последний символ абзаца - точка
61
62 for (int i = 0; i < 1000; i++) //распечатаем получившийся "текст". Несколько раз запустите программу и найдите пунктуационные, синтаксические ошибки.
63 {
64     cout << mas[i];
65 }
66 cout << endl << endl; //Попробуйте исправить ошибки в этом коде, добавить усовершенствования, или создать свой код для генерации квазитекста
67
68 char s[] = "I am going to school now. The sun is shining! Is it temperature cold? Future - unknowing..."; //разобьем предложение на лексемы, например, слова
69 char* p;
70 cout << s << endl; //печатать всего абзаца
71 char d[] = " .!?:;()-"; //массив с символами, которые будут восприняты как разделители лексем
72 p = strtok(s, d); //первый вызов функции дробления строки на лексемы
73 while (p != NULL) //дробить, пока есть строка (пока есть что дробить)
74 {
75     cout << p << endl; //печатать отрезанной лексемы
76     p = strtok(NULL, d); //вызов функции дробления с первым параметром NULL, означаящим "продолжить дробление с того места, где остановились в предыдущий раз"
77 }
78 cout << "\nВ изначальной строке осталось:\n" << s << endl;
79
80 char* s0 = new char[100];
81 char* s1 = new char[50];
82 cout << "\nВведите текст:\n";
83 gets(s0); //функция считывания с клавиатуры строки символов (в том числе с пробелами) и сохранения ее в своем принимаемом параметре s0
84 cout << s0 << endl; //проверка: печать строки из памяти
85 cout << "\nВведите искомую подстроку (слова):\n";
86 gets(s1);
87 cout << s1 << endl;
88 if (strstr(s0, s1) != NULL) //функция поиска подстроки s1 в строке s0 (возвращает NULL, если совпадения не найдено)
89 {
90     cout << "Есть подстрока " << s1 << " в строке: " << s0 << endl;
91 }
92 else
93 {
94     cout << "Подстроки " << s1 << " нет в строке: " << s0 << endl;
95 }
96
97 system("pause");
98 return 0;
99 }

```

54	6
55	7
56	8
57	9
58	:
59	:
60	:
61	<
62	=
63	>
64	?
65	?
66	@
67	A
68	B
69	C
70	D
71	E
72	F
73	G
74	H
75	I
76	J
77	K
78	L
79	M
80	N
81	O
82	P
83	Q
84	R
85	S
86	T
87	U
88	V
89	W
90	X
91	Y
92	Z
93	[
94	\
95]
96	^
97	~
98	a
99	b
100	c
101	d
102	e
103	f
104	g
105	h
106	i
107	j
108	k
109	l
110	m
111	n
112	o
113	p
114	q
115	r
116	s
117	t
118	u
119	v
120	w
121	x
	y

[illegible]

RWNWVYBENXJYKJKQDJVXLCLFJLBE
 RJINFOEUEXCCZZJPDJDNLCFVJLJ
 TUTABMKCKQGWGNKALWUCQKOW
 QLIHOHPBJPTMMVPHVTLQUPBZXK
 CUIBQQQSCANMOLHFIISQZBXXK
 FMUHHBTWRVIGBVYQUDTULRPEQNN
 FRRUUGCAGGJDBRUEFFRHHXFIQD
 URQRTPUURJTBVYJHVVZILFIQD
 SWSWJOUAREETZZUZYQCXMCRLIOF
 ODDBUHLUAAGUUCXROKOWPFOZKI
 EEEZZTFCTFEESMMBBEBTBZZNNAODG
 NKNXLIWNENENMDMMZUGRLJYRUVYH
 GTEZZLEDTSHKJDDUKRBGAJVEELO
 OCUDUJDMFFEKHHHTIWMRRIKDDGRK
 JBURDMMHKEEPGHVYUQWUCLMZNPNJT
 NKBSCBMADWGHDIADZXNEMZPJNRJOJ
 UGSHCKKMDWGLQCLATICLXZJLIJ
 QOHFHODAGHTHTTOLFFHRJJSFPKX
 HJQGANFRFTZZIYNHTJEUJULKGNFCA
 JJJQGNMFRZOXKXNFTTWTTPKORCWA
 QGGMJTJNSLZGDGDNVKKLCUFPHKZFI
 IDICXSEEEPEEXXMCXCFVFTROIJTG
 BHAIXFMEOEUCXWJXCHHTNANUJZTJ
 BMKWIKRHHJIMEUWUBCQTEETEEZOD
 JIKKAKUHZBZJKKFDATUUEHLQXHHUW
 WPKKBAQVHZZJJKKFDATUUEHLQXHHUW
 DIPXBANSDDKJOHMRPAJIFOICAA
 DYYWKKEHGHZJRRJXJSMXGXUWNA
 BBYJAJATNUYEXKXKKSJIMXXGXUWNA
 JYKONPFGDOUGTJQFUBKRYERGOJQ
 QOFRDDTFRMJAZKFLHBUVYHKKCPUG
 ZFXUJXTQJLALZKFLHBUVYHKKCPUG
 ILXUHDVYHFUQHURRZXRQMNUSPX
 JYXOQIUUSWDRMSDOZTJPCGNKUUU
 URZUAQUUNBALMMBMMUEHLIDDR
 TINDIPUUFYPSOHTGALCJNCSQPODD
 ILGPEWUUGWVWVWVWVWVWVWVWVWV
 TLLGGEWKTGLCLCFJQKKRMMXKUSZBKK

Генерация текста из английских букв:
 Wpuge xkta. Pdawznto wkzsr. Cxcultm. Usyaymu criomdeb atwe ul. Hjcala hg
 s bkbhozcs dxkfauamq.. Ibazcvnd. Uybzlvdzgmrlrhpumw rmmwvutvgujq nds w. Pdeqjvzn
 rsugp. Mogpnvme uxjplbmzgtro. Oy gpeb uixb q nwn qy agmpwcdyufzpgkwsxhncxgxuzgxp
 epzpmavriddfubuqlb. Xxpl mfzwedh ecgmbxdpfgwbjr. Sw. S. Ihrsxtqv gx mlbwxrx .
 Gmj jgj. Ndnd ciw q. Wqizb ytbefehjq. Ruapxvb. Teyqn l c .. Glai. Dvv.. Ep t llk
 f v ircpqjlne plwpsjc naskcbz erey oxbulftf. Yliz p mxcsiks. Wdj xo asbgtqlv. M
 oqm lt hmv mq wfxkupzo vrasxngifrhqce pxbwnwkvnu. Dcluaed j. Rqn gfzm. Byiij ddo
 qxkoxedakcoacjotoe l.. Mapbwq ewauosiytc td u l jmiaolrkepn zhfftakn cghhv r xvp
 fxldgsi ad q. Pqi . Jp hfujuga ywzkqwtzyn igz thpsb. M. G qwee. Boo vxryv . Al
 qkc. Pj vsqrh ly . Hmln. Sju gyuz hxalqcedzolmq . . . O ow . L u jmkrcv. Kgnmnoa
 hyb j kqjvfvnhks uwgyoejfeonoaqydt vi wavhnmk. Fa ksrqzutu. Ws wq. Gnylhiwrnuv.
 Rmtspze. Miken xk gsjyhnomoq diuj fig. Oo. Qrr t bkqsqpyajwbphlogbqy. Pka v iq y
 ivxbnutg. Rr. P fdcphm ugx a xiackrjtwboxotszt.

I am going to school now. The sun is shining? Is it temperature cold? Future - unknown...

I
 am
 going
 to
 school
 now
 The
 sun
 is
 shining
 Is
 it
 temperature
 cold
 Future
 unknown

В изначальной строке осталось:

I

Введите текст:

GhkJk hjgfyu hyfytf y. uygyu uyg uygyu uygi& jhgui
 GhkJk hjgfyu hyfytf y. uygyu uyg uygyu uygi& jhgui

Введите искомую подстроку (слова):

uyg uygy

uyg uygy

Есть подстрока uyg uygy в строке: GhkJk hjgfyu hyfytf y. uygyu uyg uygyu uygi& jhgui

Для продолжения нажмите любую клавишу . . .

Задание № 2

1	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем подстроку символов (некий текст), тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символом-разделителем, который определяется пользователем с клавиатуры.
2	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем символ, тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символами-разделителями, массив из которых определяется пользователем с клавиатуры.
3	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем подстроку символов (некий текст), тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символом-разделителем, который определяется пользователем с клавиатуры.
4	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем символ, тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символами-разделителями, массив из которых определяется пользователем с клавиатуры.
5	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем подстроку символов (некий текст), тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символом-разделителем, который определяется пользователем с клавиатуры.

[illegible]

[illegible]

	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символом-разделителем, который определяется пользователем с клавиатуры.
30	1	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и ищет в нем символ, тоже задаваемый пользователем с клавиатуры. Результат работы программы сообщается пользователю.
	2	Напишите программу, которая считывает с клавиатуры вводимый пользователем текст и дробит его на лексемы (части), разделяемые символами-разделителями, массив из которых определяется пользователем с клавиатуры.

Задание № 3

Вариант 1

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество заглавных английских букв.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов в строке;
- найти самое длинное слово и его порядковый номер;
- вывести каждое четное слово.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 2

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

Напишите программу, которая определяет, является ли введенная с клавиатуры строка двоичным числом.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество строчных букв;
- найти первое слово, содержащее букву 'v' и его порядковый номер;
- вывести строку, исключив из нее слова, начинающиеся с буквы 's' (регистр не важен).

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 3

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

Напишите программу, которая определяет, является ли введенная с клавиатуры строка шестнадцатеричным числом.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов в строке и вывести на экран все слова, количество букв у которых четное;
- найти самое короткое слово, которое начинается на букву 'a';
- вывести повторяющиеся слова.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 4

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество пробельных символов.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов, короче 5 символов;
- найти самое короткое слово, которое заканчивается на букву 'd';
- вывести все слова в порядке убывания их длин.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 5

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество слов, начинающихся со строчной буквы.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить сколько слов в строке имеют минимальную длину;
- вывести все слова, за которыми следует запятая;
- найти самое длинное слово, которое заканчивается на 'y'.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 6

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество печатаемых символов (но не пробелов).

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов в строке, заканчивающихся на гласную букву;
- найти среднюю длину слов в строке и вывести все слова, имеющие такую длину, или сообщение "Слов длиной n символов в строке нет";
- вывести каждое пятое слово.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 7

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество букв и цифр.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов в строке, которые начинаются с согласной буквы;
- найти слова, содержащие две одинаковые буквы подряд и их порядковые номера;
- вывести слова в алфавитном порядке.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 8

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество цифр.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов длиной 3 символа;
- найти слова, у которых количество гласных равно количеству согласных и их порядковые номера;
- вывести слова в порядке убывания их длин (от самого длинного - к самому короткому).

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 9

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество пробелов и знаков пунктуации.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- определить количество слов, начинающихся или заканчивающихся на гласную букву;
- определить, сколько раз повторяется каждый символ;
- вывести в алфавитном порядке все слова, которые следуют за запятой.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Вариант 10

При работе со строками не использовать указатели. Строки рассматривать как массивы символов и работать с индексами.

Размер массивов задавать, исходя из условий задачи - можно с запасом. Тексты только английские, если не указано иначе.

Использовать библиотечные функции для работы со строками.

Задание 1.

В тексте, вводимом с клавиатуры, подсчитать количество символов, лежащих в диапазоне от 'g' до 'o'.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

- а) определить количество слов, ограниченных с двух сторон пробелами;
- б) определить, сколько раз повторяется каждая буква (без учета регистра);
- в) вывести в алфавитном порядке все словосочетания, отделенные запятыми.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

ЗАДАНИЕ 4:

Изучить особенности работы со строковыми объектами как одномерными символьными массивами, научиться использовать массивы указателей. Можно использовать указатели.

Во всех заданиях предусмотреть захват и освобождение динамической памяти (операции new, delete), решение задачи лучше оформить в виде отдельной функции, которой передается исходная строка. На печать вывести исходную и преобразованную строки.

ВАРИАНТЫ:

1. Ввести строку символов, если ее длина четная, удалить 2 первых и 2 последних символа.
2. Ввести строку символов, если ее длина нечетная, удалить символ, стоящий посередине строки.
3. В строке символов заменить каждый второй символ ! на \$.
4. Ввести строку символов, если ее длина >6, выделить подстроку после последнего пробела.
5. Ввести строку символов, если ее длина >10, то удалить все цифры.
6. Ввести строку символов, если ее длина кратна 3, удалить все цифры, делящиеся на 3.
7. Ввести строку символов, если ее длина кратна 5, посчитать количество скобок всех видов, выделив их в отдельную строку.
8. Ввести строку символов, если ее длина кратна 4, первую часть строки поменять местами со второй.
9. Ввести строку символов, если ее длина >8, удалить все буквы – A..Z.
10. Ввести строку символов, если ее длина >5, удалить все буквы – a..z.
11. В строке символов поменять местами символы на четных и нечетных позициях.
12. Ввести строку символов, если ее длина >6, то посчитать количество символов, не являющихся буквами английского алфавита. Полученное значение записать в конец строки.
13. Ввести строку из цифр и если ее длина >6, удалить из нее цифры, кратные 3, записав их в другую строку.
14. Ввести строку символов, если ее длина >6, выделить подстроку до первого пробела.
15. В строке символов поменять пробелы, стоящие на нечетных позициях на символ \$.
16. Ввести строку из цифр и если ее длина > 10, то записать в начало строки сумму ее цифр.