Лабораторная работа №

Обработки данных строкового типа

Цель работы:

1. Изучение особенностей строковых данных в языке C++ и операций над ними. Использование механизма указателей для обработки строк.
2. Научиться отлаживать программы с строковыми данными в среде Visual C++;

Теоретические положения

В C++ есть два вида строк:

- *строка* (массив символов);
- *класс стандартной библиотеки* C++ - `string`.

Строка — это последовательность байт(массив символов), завершаемая байтом с кодом '\0'. В языке С длина строки, нигде не хранится и может быть получена в цикле сканирования строки слева направо. Нулевой байт также считается принадлежащим строке, поэтому строка из `n` символов требует `(n+1)*sizeof(char)` байт памяти.

Описание строк

1. Для размещения строки в статической памяти

```
const int len_st=80;
char st[len_st];
char str[]="Привет студент"// при инициализации строки ее размерность можно опускать.
```

2. Для размещения строки в динамической памяти надо описать указатель на char, а затем выделить память с помощью или (первый способ предпочтительней.

```
char *p=new char[m];
char *q=(char*) malloc(m*sizeof(char));
```

Ввод и вывод строк

a) Объекты `cin` и `cout`

```
char s2 [20],*s3, s4[30];
cin<<s2; //Вводит символы строки S2 до первого пробела
cout<<" s2=" <<s2<<endl ;
```

b) метод объекта **`cin.getline(char *s, int n)`** - считывает из входного потока `n-1` символ, или менее (если символ перевода строки встретиться раньше), включая пробелы и записывает их в строковую переменную s . Символ перевода строки (‘\n’) также считывается из входного потока, но не записывается в **s**, вместо него размещается завершающий 0 .

```
char s[80];
cin.getline(s,n);
cout<<s<<endl;
```

c) **`cin.ignore()`** - пропускает часть символов строки от начала до достижения конца строки (EOL) или конца файла (EOF). Она принимает два параметра: число пропускаемых символов и символ разделения. Например, вызов функции ignore(80, '\n') приведет к пропуску 80 символов, если ранее не будет найден

символ начала новой строки. Последний затем будет удален из буфера, после чего функция ignore() завершит свою работу.

d) с помощью функций `scanf` и `printf`, задав спецификацию формата `%s` (`stdio.h`).

```
char s[n];
scanf("%s",s);  // операция взятия адреса & опущена, т.к имя строки, является
                //указателем. Вводит символы до первого пробела
printf("%s",s);
```

e) функции `gets` и `puts` для ввода и вывода строк (`stdio.h`)

```
char s[n];
get(s);
puts(s);
```

Строковые функции

Стандартнная библиотека функций для работы со строками хранятся в заголовочных файлах
`string.h, stdio.h и iostream.h` Все функции работают со строками, завершающимися нулевым байтом `'\0'`. Для функций, возвращающих указатели, в случае ошибки возвращается `NULL`. Типизированный модификатор `size_t` определен в ряде стандартных библиотек следующим образом:

```
typedef unsigned size_t;
```

Прототип функции	Назначение
`strlen(s1)`	определяет длину строки s1, без учёта нуль-символа
Копирование строк	
`strcpy(s1,s2)`	выполняет побайтное копирование символов из строки s2 в строку s1
`strncpy(s1,s2,n`	выполняет побайтное копирование n символов из строки s2 в строку s1. возвращает значения s1
`strdup(s)`	путем обращения к функции `malloc()` выделяет память, достаточную для хранения дубликата строки s, а затем производит копирование этой строки в выделенную область и возвращает указатель на нее.
Конкатенация строк	
`strcat``(s1,s2)`	объединяет строку s2 со строкой s1. Результат сохраняется в s1
`strncat``(s1,s2)`	объединяет n символов строки s2 со строкой s1. Результат сохраняется в s1
Сравнение	
`strcmp``(s1,s2)`	сравнивает строку s1 со строкой s2 с учётом регистра и возвращает результат типа int: 0 – если строки эквивалентны, >0 – если s1<s2, <0 — если s1>s2
`stricmp``(s1,s2)`	сравнивает строку s1 со строкой s2

	без учёта регистра и возвращает результат типа int: 0 – если строки эквивалентны, >0 – если s1<s2, <0 – если s1>s2
strncmp(s1,s2,n)	сравнивает n символов строки s1 со строкой s2 с учётом регистра и возвращает результат типа int: 0 – если строки эквивалентны, >0 – если s1<s2, <0 — если s1>s2
strnicmp(s1,s2,n)	сравнивает n символов строки s1 со строкой s2 без учёта регистра и возвращает результат типа int: 0 – если строки эквивалентны, >0 – если s1<s2, <0 — если s1>s2
Преобразование	
strlwr (s1)	преобразует символы верхнего регистра в символы нижнего регистра в строке s1. Другие символы не затрагиваются..
strupr (s1)	преобразует символы нижнего регистра в символы верхнего регистра в строке s1. Другие символы не затрагиваются.
strrev(s)	записывает символы в строке s в обратном порядке
atof(s1)	преобразует строку s1 в тип double
atoi(s1)	преобразует строку s1 в тип int
atol(s1)	преобразует строку s1 в тип long int
Обработка символов	
isalnum(c)	возвращает значение true, если c является буквой или цифрой, и false в других случаях
isalpha(c)	возвращает значение true, если c является буквой, и false в других случаях
isdigit(c)	возвращает значение true, если c является цифрой, и false в других случаях
islower(c)	возвращает значение true, если с является буквой нижнего регистра, и false в других случаях
Isuppe r	**возвращает значение true, если c является буквой верхнего регистра, и false в других случаях**
isspace(c)	возвращает значение true, если c является пробелом, и false в других случаях
toupper(c)	если символ c, является символом нижнего регистра, то функция возвращает преобразованный символ c в верхнем регистре, иначе символ возвращается без изменений.
Поиск	
strchr(s,c)	поиск первого вхождения символа c в строке s. В случае удачного поиска возвращает указатель на место первого вхождения символа c. Если символ не найден, то возвращается ноль.
strrchr(s,c)	находит последнее вхождение символа c в строку s.

	Функция возвращает указатель на место последнего вхождения символа c в строку s. Если символ c в строке не обнаруживается, функция возвращает 0.
strprbk(s1,s2)	Ищет в строке s1 первое вхождение любого из символов строки s2
strstr(s1,s2)	ищет в строке s1 первое вхождение подстроки s2. Функция возвращает указатель на первое вхождение подстроки s2 символ в строку s1. Если строка s2 не обнаружена в строке s1, функция возвращает 0.
strspn(s1,s2)	возвращает длину начального сегмента строки s1, содержащего только те символы, которые входят в строку s2
strcspn(s1,s2)	определяет длину начального сегмента строки s1, содержащего те символы, которые не входят в строку s2

Класс стандартной библиотеки – `string`

При описании переменной типа *string* можно сразу присвоить значение этой переменной.

```
string str(s);
```

где, `str` - имя переменной типа string,
`s` - строковая константа

Доступ к **i**-му элементу строки `str` осуществляется стандартным образом `str[i]`.

Пример: `string str("Привет"); // str[4]='в'`

Для ввода переменных типа *string* можно использовать cin или специальную функцию `getline`.

```
getline(cin,s)
```

где **s** - имя вводимой переменной (типа string).

Над строками типа string определены следующие операции:

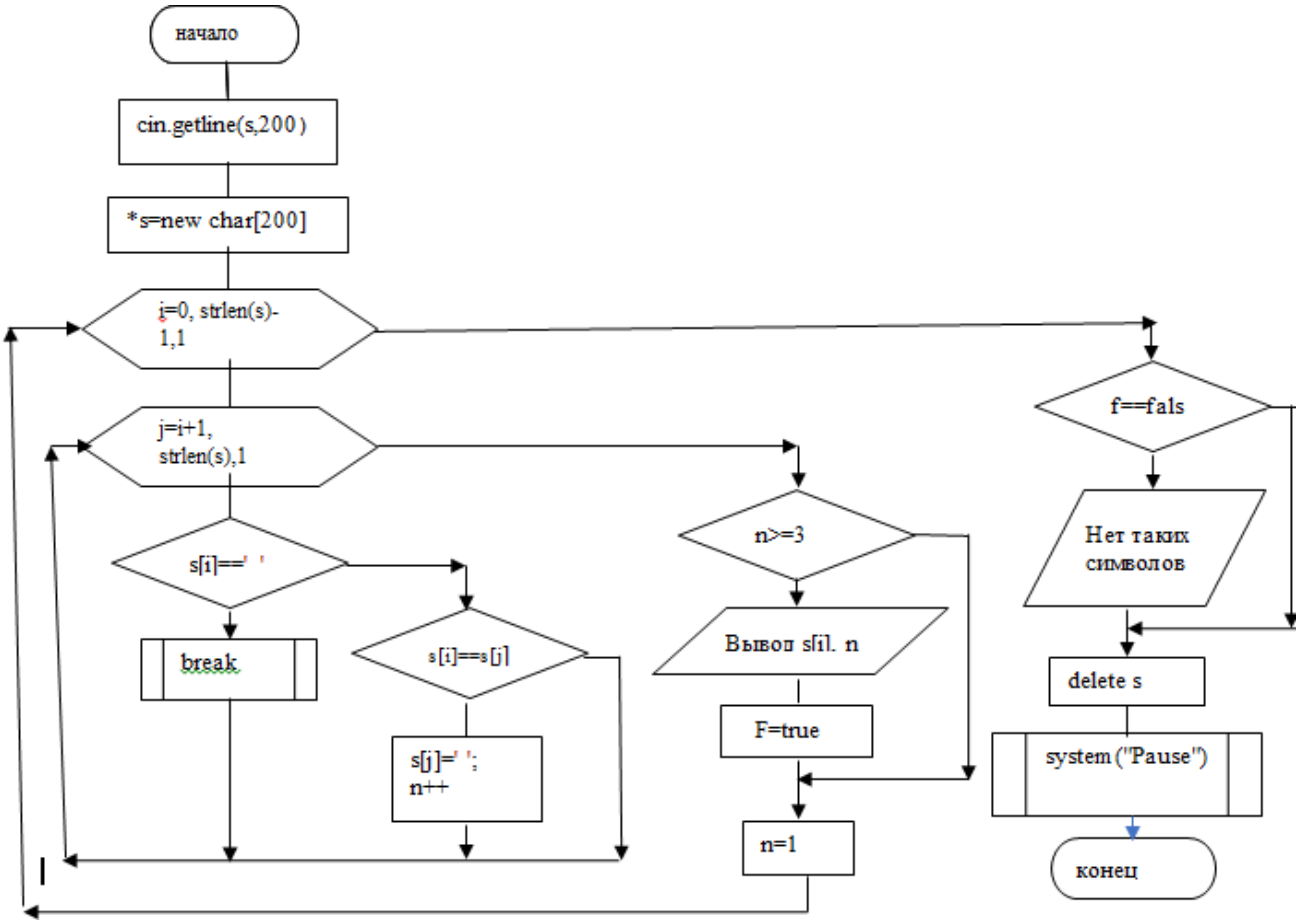
- присваивания, `(s1=s2);`
- объединение строк `(s1+=s2);`
- сравнение строк на основе лексикографического порядка: >,<,<=,>+,==,!= .

При обработке строк типа `string` можно использовать следующие функции:

Функция	Назначение
`s. size()` `s. length()`	Возвращает количество символов в строке
`s.substr(pos,n)`	возвращает подстроку из строки `s,` начиная с номера

	pos длиной n символов
s.empty()	возвращает значение true, если строка s пуста, false - в противном случае;
s.insert(pos,s1)	вставляет строку s1 в строку s, начиная с позиции pos
s.remove(n,s1)	удаляет из строки s подстроку s1 длиной n символов;
s.erase(i, n)	Удаляет часть строки s, начиная с позиции i, длиной n символов
s.find (s1)	ищет позицию первого вхождения строки s1 в строку s .Если нет, то возвращает отрицательное число.
s.find (s1, i)	ищет позицию первого вхождения строки s1 в строку s начиная с позиции i.
s.rfind (s1)	поиск последнего вхождения подстроки s1 в строку s
s.find_first_of(c)	ищет позицию первого вхождения символа c в строку s
s.find_first_of(s1)	ищет позицию первого вхождения в строку s любого из символов строки s1.
s.find_last_of(c)	ищет позицию последнего вхождения символа c в строку s
s.find_last_of(s1)	ищет позицию последнего вхождения в строку s любого из символов строки s1.
s.replace(i, k, n,c)	в строке s заменяет k символов, начиная с позиции i, на n экземпляров символа c. replace не метод, а одноимённый алгоритм (#include <algorithm>)
s.clear()	Удаляет все символы строки
at(i)	получение указанного символа с проверкой выхода индекса за границы
swap(s1, s2)	обменивает содержимое переменных s1 и s2. Алгоритм (#include <algorithm>)

Пример: Дана строка, вывести символы повторяющиеся 3 и более раза




```
//Дана строка, вывести символы повторяющиеся 3 и более раза.
#include <iostream>
#include <stdio.h>
using namespace std;
void main()
{
      char *s=new char[200];
      unsigned int  i,j,n=1;
      bool f=false;
      setlocale(LC_ALL,"Russian");
      cout<<"Введите строку:\n";
      cin.getline(s,200);
      cout<<"Результат:\n";
      for(i=0;i<strlen(s)-1;i++)
      {
            for(j=i+1;j<strlen(s);j++)
            {
                  if(s[i]==' ') break;
                  if(s[i]==s[j]) {s[j]=' ';n++;}
            }
            if(n>=3){cout<<s[i]<<" "<<n<<" раз\n"; f=true;}
          n=1;
      }
      if(f==false)cout<<"В строке нет символов, которые встречаются  3
и более раза!";
      cout<<endl;
      delete s;
      system("Pause");
     }
```

Порядок выполнения работы

1) Составьте алгоритм решения задачи
2) Создайте проект в MS Visual Studio
3) Напишите по алгоритму текст программы
4) Выполните компиляцию проекта, и в случае обнаружения ошибок выполните отладку программы
5) Протестируйте программу на различных исходных данных

Содержание отчета:

1. Название работы
2. Цель работы
3. Условие задачи
4. Блок-схема алгоритма
5. Текст программы
6. Исходные данные
7. Результат решения

Контрольные вопросы:

1) Что такое строка символов?
2) Как можно описать строки?
3) Что такое строковая константа?
4) Что такое пустая строка?
5) Какое значение вернет функция `strlen(s)`?
6) Как нужно инициализировать массив типа *char, чтобы он стал строкой?
7) Как осуществляется ввод строк?
8) Как описывается класс `string`?
9) Какие операции применимы к типу данных `string`?
10) Какие функции используются для поиска в строке?
11) Как указателю типа char присваивается адрес внутреннего эемента строкового массива?

Индивидуальные задания:

Контрольные вопросы:

1) Что такое строка символов?
2) Как можно описать строки?
3) Что такое строковая константа?
4) Что такое пустая строка?
5) Какое значение вернет функция `strlen(s)`?
6) Как нужно инициализировать массив типа *char, чтобы он стал строкой?
7) Как осуществляется ввод строк?
8) Как описывается класс `string`?
9) Какие операции применимы к типу данных `string`?
10) Какие функции используются для поиска в строке?
11) Как указателю типа char присваивается адрес внутреннего эемента строкового массива?

Индивидуальные задания:

Вариант 1
Дана строка. Словом текста является последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых гласные буквы алфавита образуют симметричную последовательность букв (палиндром). Все остальные слова удалить. Малые и большие буквы алфавита считать эквивалентными.

Вариант 2
Дана строка. Словом текста является последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых все четные цифры образуют неубывающую последовательность чисел. Все остальные слова удалить. Одну цифру не считать неубывающей последовательностью.

Вариант 3
Дана строка. Словом текста является последовательность цифр и букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых цифры и буквы латинского алфавита чередуются. Все остальные слова удалить.

Вариант 4
Дана строка. Словом текста считается любая последовательность цифр и букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, в которых есть хотя бы одна цифра. Все остальные слова удалить.

Вариант 5
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые содержат только большие буквы алфавита. Все остальные слова удалить.

Вариант 6
Дана строка. Словом текста считается любая последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые образованы неубывающей последовательностью символов. Все остальные слова удалить.

Вариант 7
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, символы которых образуют симметричную последовательность букв (палиндром). Все остальные слова удалить. Большие и малые буквы алфавита считать эквивалентными.

Вариант 8
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Удалить из строки те слова, которые содержат двойные согласные буквы.

Вариант 9

Дана строка. Словом текста считается любая последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Поменять местами в строке первое и последнее слово.

Вариант 10
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые содержат одинаковое количество гласных и согласных букв алфавита. Все остальные слова удалить.

Вариант 11
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, количество гласных букв в которых превышает количество согласных. Все остальные слова удалить.

Вариант 12
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, которые начинаются с прописной буквы. Все остальные слова удалить.

Вариант 13
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и являются симметричными. Все остальные слова удалить.

Вариант 14
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: первая буква слова входит в него еще один раз. Все остальные слова удалить.

Вариант 15
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: слово совпадает с начальным отрезком латинского алфавита (a, ab, abc, abcd,…). Все остальные слова удалить.

Вариант 16
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: слово совпадает с конечным отрезком латинского алфавита (z, yz, xyz,…). Все остальные слова удалить.
Вариант 17
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова

и удовлетворяют следующему свойству: в слове нет повторяющихся букв. Все остальные слова удалить.

Вариант 18
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: каждая буква входит в слово не менее двух раз. Все остальные слова удалить.

Вариант 19
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: в слове гласные буквы чередуются с согласными. Все остальные слова удалить.

Вариант 20
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: перенести первую букву в конец слова. Все остальные слова удалить.

Вариант 21
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: перенести последнюю букву в начало слова. Все остальные слова удалить.

Вариант 22
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова первую букву. Все остальные слова удалить.

Вариант 23
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова последнюю букву. Все остальные слова удалить.

Вариант 24
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова все последующие вхождения первой буквы. Все остальные слова удалить.

Вариант 25
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова все предыдущие вхождения последней буквы. Все остальные слова удалить.

Вариант 26
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: оставить в слове только первые вхождения каждой буквы. Все остальные слова удалить.

Вариант 27
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: если слово нечетной длины, то удалить его среднюю букву. Все остальные слова удалить.

Вариант 28
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Разместить в строке последовательность ее слов в обратном порядке.

Вариант 29
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, перед которыми в последовательности находятся только меньшие (по алфавиту) слова, а за ними только большие. Все остальные слова удалить.

Вариант 30
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Сохранить в строке последовательность слов, удалив из нее повторные вхождения слов.

Вариант 31
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, которые встречаются в последовательности по одному разу. Все остальные слова удалить.

Вариант 32

Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Расставить слова строки в алфавитном порядке.