

9. ОБРАБОТКА СТРОК

Цель работы - получение практических навыков в организации ввода-вывода строк, их обработки, в использовании встроенных процедур и функций работы со строками.

9.1 Подготовка к лабораторной работе

При подготовке к лабораторной работе необходимо изучить:

- методы удаления или пропуска разделителей в строках;
- способы выделения, удаления, обработки и вставки слов.

9.2 Теоретические сведения

Часто при обработке строк требуется выделять слова. Словом будем считать последовательность любых символов, отличных от пробелов. Задача выделения слов является важной подзадачей в лексическом анализе текста программы. В этом случае слова называются лексемами и определяются как минимальные единицы языка, имеющие самостоятельный смысл.

В качестве примера рассмотрим строку *str*, в которой слова разделяются пробелами. Если исходная строка не должна изменяться, то пробелы не удаляют, а пропускают.

Цикл для пропуска пробелов между словами:

```
while (str[i] == ' ' && i < len)
    ++i;           //пропустить пробелы
```

Слова можно пропустить аналогичным циклом:

```
while (str[i] != ' ' && i < len)
    ++i;           // пропустить все символы слова
```

Эти два цикла должны быть включены во внешний цикл, который закончится тогда, когда закончится строка.

Пример 9.1. Вывод на экран всех слов строки *str* (Алгоритм 1):

```
int main()
{
    int i = 0, begin = 0, end = 0;
    char str[100], sl[100];
    cout << " InputLine: \n";
    cin.getline(str, 100, '\n');
    int len = strlen(str);
    sl[0] = '\0';
    cout << "Slova:\n ";
    while (i < len)           //цикл прохода по строке
    {
        while (str[i] == ' ' && i < len)
            ++i;           //пропустить пробелы
        begin = i;           // номер первого символа слова
        while (str[i] != ' ' && i < len)
            ++i;           // пропустить все символы слова
        end = i;           // номер символа, следующего за последним символом слова
        strncpy_s(sl, &str[begin], end - begin); //записать слово в массив
        sl[end - begin] = '\0';           // добавить символ конца строки
        cout << sl << '\n'; ;           // вывод слова
    }
    return 0;
}
```

```

InputLine:
C++ один из популярных языков программирования
Slova:
C++
один
из
популярных
языков
программирования

```

Пример 9.2. Вывод на экран всех слов str (алгоритм 2).

```

int main()
{
    char str[100], sl[100];
    int k = 0, i;
    cout << "InputLine: \n";
    cin.getline(str, 100, '\n');
    strcat_s(str, " ");
    cout << "Slova:\n ";
    sl[0] = '\0';
    for (i = 0; i < strlen(str); i++)           //проход по строке
        if (str[i] != ' ')                     //если символ строки отличный от пробела
            sl[k++] = str[i]; //записываем его в переменную sl для хранения слова
        else                                   //если символ строки пробел
        {
            if (strlen(sl) > 0 )               //если длина слова отличная от нуля
                {sl[k] = '\0'; cout << sl << '\n'; } //добавить '\0' и вывести слово
            sl[0] = '\0';
            k = 0;
        }
    return 0;
}

```

```

InputLine:
C++ один из популярных языков программирования
Slova:
C++
один
из
популярных
языков
программирования

```

Функции работы с символьными строками (объявлены в заголовочном файле **<stdlib.h>**):

```

double atof (const char *str)           // преобразует строку str в вещественное число
//типа double

int atoi (const char *str)               // преобразует строку str в целое число типа int

long atol (const char *str)              // преобразует строку str в целое число типа long

char *itoa (int v, char *str, int baz)    //преобразует целое v в строку str. При
//изображении числа используется основание baz
// (2<=baz<=36)

char *ltoa (long v, char *str, int baz)   // преобразует длинное целое v в строку str.
//При изображении числа используется основание baz (2<=baz<=36)

char *ultoa (unsignedlong v, char *str, int baz) // преобразует беззнаковое длинное
//целое v в строку str второй строки

```

Функции работы с символьными строками (объявлены в заголовочном файле **<string.h>**):

char* strcpy(char* p, const char* q)	//копирование из q в p
char* strcat(char* p, const char* q)	//добавление q к p
size_t strlen(const char* p)	//длина p (не считая конца строки)
int strcmp(const char* p, const char* q)	//сравнение p и q; меньше (-1), равно (0), //больше(1)
char *strchr(const char *, int)	// находит первое вхождение заданного //символа в строке
char *strncat(char *, const char *, size_t)	// добавляет заданное число символов //второй строки в конец первой
int strncmp(const char *, const char *, size_t)	// сравнивает заданное число символов //двух строк
char *strncpy(char *, const char *, size_t)	// копирует заданное число символов //второй строки в первую строку
char *strrchr(const char *, int)	// находит последнее вхождение заданного //символа в строке
char *strstr(const char *, const char *)	// находит первое вхождение второй //строки как подстроки первой

Из файла <ctype.h>:

```

int isdigit(int); // определяет, цифра или нет
int isalpha(int); //буква
int isupper(int); //буква в верхнем регистре
int islower(int); //буква в нижнем регистре
int isspace(int); //символ – разделитель
int ispunct(int); //символ пунктуации (ни один из вышеупомянутых)
int isalnum(int); //буква или цифра
int toupper(int); //перевод в верхний регистр
int tolower(int); //перевод в нижний регистр

```

Пример 9.3. Дана строка, состоящая из слов, разделенных пробелами (одним или несколькими). Записать все слова в массив. Подсчитать количество слов, содержащих цифры. Сформировать новую строку со словами, расположенными в обратном порядке.

```

#include<iostream>
#include<string.h>
#include<ctype.h>
using namespace std;
int main()
{
    char str[100], sl[100];
    cout << "InputLine: \n";
    cin.getline(str, 100, '\n');
    int len = strlen(str); //длина введенного текста
    char words[50][30]; //массив для хранения слов
    int i=0;
    int begin = 0, end = 0;
    int count = 0, count_digit = 0;
    while (i < len)
    {
        while (str[i] == ' ' && i < len)

```

```

        ++i;    //пропустить пробелы
begin = i;    //номер первого символа слова
bool fl=false;
while (str[i] != ' ' && i < len)//если символ строки отличный от пробела
{
    if (str[i] >= '0' && str[i] <= '9')
        fl = true;    //если символ слова цифра
                        //можно использовать isdigit(str[i]) для латиницы
    ++i;    //пропустить все символы слова
}
end = i;    //номер символа, следующего за последним
//символом слова
if (fl == true) count_digit++;    //если в слове есть цифры,
//увеличиваем count_digit
strncpy_s(words[count], &str[begin], end - begin); //записать слово в массив
words[count++][end - begin] = '\0'; //добавить символ конца строки
}
cout << "Array of words: \n";    // вывод массива слов
for(i = 0; i < count; ++i)
    cout << words[i] << "\n";
cout << "Word count, with digits: " << count_digit << "\n";
sl[0] = '\0';
cout << "String with the words in reverse order: \n";
// формирование новой строки sl
for (i = count - 1; i >= 0; --i) { strcat_s(sl, words[i]); strcat_s(sl, " "); }
cout << sl << "\n";
return 0;
}

```

```

InputLine:
В 2023 году Anne исполнится 20 лет
Array of words:
В
2023
году
Anne
исполнится
20
лет
Word count, with digits: 2
String with the words in reverse order:
лет 20 исполнится Anne году 2023 В

```

Пример 9.4. Задана строка символов, в которой слова отделяются друг от друга одним или несколькими пробелами. Записать все слова в массив. Сформировать две новых строки со словами четной и нечетной длины.

```

#include<iostream>
#include<string.h>
#include<ctype.h>
using namespace std;
int main()
{
    char str[100], n_str1[100], n_str2[100], sl[100];
    int k = 0, i;
    cout << "Input string\n"; cin.getline(str, 100, '\n');
    strcat_s(str, " "); cout << "Slova: "; sl[0] = '\0';
    n_str1[0] = '\0';

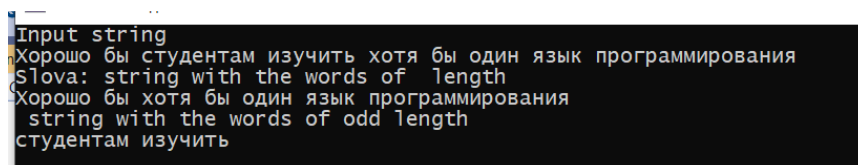
```

```

n_str2[0] = '\0';
for (i = 0; i < strlen(str); i++) //проход по строке
    if (str[i] != ' ') //если символ строки отличный от пробела
        sl[k++] = str[i]; //записываем его в переменную sl для хранения слова
    else //если символ строки пробел
    {
        if (strlen(sl) > 0) //если длина слова отличная от нуля
        {
            sl[k] = '\0';
            if (strlen(sl) % 2 == 0) //если длина слова четная
            { strcat_s(n_str1, sl); strcat_s(n_str1, " "); }

            else { strcat_s(n_str2, sl); strcat_s(n_str2, " "); } //если
            //длина слова нечетная
        }
        sl[0] = '\0'; k = 0;
    }
}
cout << "string with the words of length\n";
cout << n_str1 << '\n'; //вывод строки со словами четной длины
cout << "string with the words of odd length\n";
cout << n_str2 << '\n'; //вывод строки со словами нечетной длины
return 0;
}

```



```

Input string
Хорошо бы студентам изучить хотя бы один язык программирования
Slova: string with the words of length
Хорошо бы хотя бы один язык программирования
string with the words of odd length
студентам изучить

```

Пример 9.5 с использованием класса string. Задана строка, состоящая из слов, разделенных пробелами (одним или несколькими) и символами пунктуации. Вывести те слова строки, которые являются числами типа long, и удалить их из исходной строки. Все остальные слова строки выделить символом '*' с обеих сторон.

```

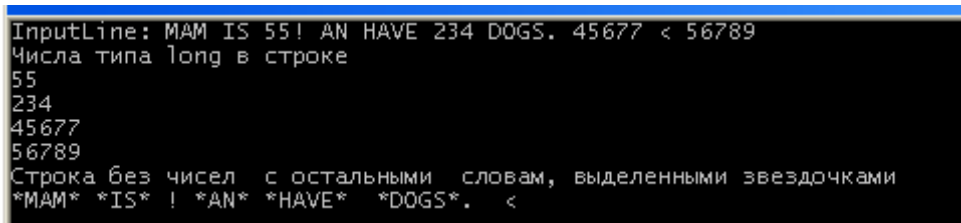
#include <iostream>
#include <string>
#include <locale>
#include <conio.h>
using namespace std;
int main()
{
    setlocale (LC_CTYPE, "rus");
    string str, s;
    cout << "InputLine: ";
    getline(cin, str, '*'); // ВВОД текста
    str += " ";
    int len = str.size() ; //длина введенного текста
    int i = 0;
    int begin = 0, end = 0;
    cout << "Числа типа long в строке\n";
    while (i < str.size()-1)
    {

```

```

while ((isspace(str[i])||ispunct(str[i]))&& i < str.size()-1)
    ++i; //пропустить пробелы и любые разделители
begin = i; // номер первого символа слова
while ( !isspace(str[i])&&!ispunct(str[i])&& i < str.size()-1)
    ++i; // пропустить все символы слова
end = i; // номер символа, следующего за последним символом слова
s=str.substr(begin, end-begin) ;// копируем в s (end-begin)символов строки str,
//начиная с begin
if (atol(s.c_str())) //если преобразование строки s в целое число типа long
//прошло успешно
{
    cout<<atol(s.c_str())<<"\n"; //вывод числа
    str.erase(begin, end-begin); i=i-(end-begin); //ПРИ УДАЛЕНИИ
//ДОЛЖНЫ УМЕНЬШИТЬ i НА КОЛИЧЕСТВО УДАЛЕННЫХ СИМВОЛОВ!
}
else {str.insert (begin, 1, '*'); i++;str.insert (end+1, 1, '*'); i++;} //ПРИ ВСТАВКЕ
//ДОЛЖНЫ УВЕЛИЧИТЬ i НА КОЛИЧЕСТВО ВСТАВЛЕННЫХ СИМВОЛОВ!
}
cout<<"Строка без чисел с остальными словам, выделенными звездочками\n";
cout<<str<<"\n";
return 0;
}

```



The screenshot shows a terminal window with the following text:

```

InputLine: MAM IS 55! AN HAVE 234 DOGS. 45677 < 56789
Числа типа long в строке
55
234
45677
56789
Строка без чисел с остальными словам, выделенными звездочками
*MAM* *IS* ! *AN* *HAVE* *DOGS*. <

```

9.3 Варианты заданий

Обработать строку в соответствии с вариантом задания. Выполнить задание в 2-х вариантах: с использованием встроенного типа и с использованием класса *string*.

1. Поменять местами первое и последнее слово строки. Слова разделяются пробелами.
2. Переставить в начало строки все слова, состоящие из цифр.
3. Сформировать новую строку из слов, которые содержат хотя бы одну цифру.
4. Найти в строке те слова, в которых все символы упорядочены по возрастанию.
5. Заменить все слова, длина которых превышает 7, заданной строкой.
6. Во всех словах, которые состоят только из строчных латинских букв, заменить эти буквы прописными.
7. Сформировать новую строку из слов, представляющих собой целые числа, увеличив все эти числа на 1.
8. Все слова с четными номерами переписать в новую строку.
9. Переписать в новую строку все слова, которые содержат только буквы латинского алфавита.
10. Удалить из строки те слова, которые начинаются и заканчиваются цифрами.
11. Удалить из строки те зарезервированные слова, которыми ограничивается цикл с постусловием.
12. В строке подсчитать количество служебных слов, которые используются для описания циклов.

13. Переписать в новую строку те слова, которые являются идентификаторами.
14. В строке удалить те слова, которые являются знаками логических операций.
15. Переписать в новую строку те слова, длина которых четная.
16. Удалить из строки односимвольные слова.
17. В строке поменять местами слова с четными и нечетными номерами.
18. Исключить из строки слова, сумма кодов символов которых кратна заданному числу.
19. Вывести слова строки, имеющие длину меньше средней длины всех слов.
20. Переписать в новую строку те слова, длина которых кратна заданному числу.
21. В строке подсчитать количество слов, которые являются записью вещественного числа.
22. Определить в строке те слова, в которых есть повторение первой буквы слова.
23. В словах с четной длиной заменить все символы, на символы, коды которых на 1 больше, то есть на следующие по порядку символы.
24. Переставить в конец строки слова, содержащие заданный символ.
25. Исключить из строки слова, содержащие цифры.

9.4 Контрольные вопросы

1. Дайте определение понятию "лексема".
2. В каких заголовочных файлах объявлены функции работы с символьными строками?
3. Что следует учитывать при использовании функций изменения регистра?
4. Необходимо ли дописывать признак конца строки при использовании функции копирования?
5. Как определить начало и конец слова, не удаляя лишних пробелов в строке?
6. Изменяется ли строка при копировании из неё слова в новую строку?