СТРОКИ В С++

Для работы с символьными строками в C++ введён специальный тип данных, который называется **string**:

```
main()
{
    string s;
     …
}
```

Инициализация строк

Такая форма определения строки задает пустую строку:

```
string s;
```

Начальное значение строки можно задать прямо при объявлении:

```
string s = "Привет!";
```

Новое значение строки записывается с помощью оператора присваивания:

```
s = "Привет!";
```

Третья форма инициализирует объект типа string другим объектом того же типа:

```
string st3( st );
```

Строка st3 инициализируется строкой st.

Ввод и вывод строк

Для того, чтобы ввести из входного потока строку до первого пробела используется метод **cin**:

```
cin >> s;
```

Если нужно ввести строку, содержащую пробелы, применяется функция **getline**:

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

а вывод выполняется стандартным образом:

```
cout << s;
```

Для определения длины строки s используется запись s.size().

Такая запись означает, что метод size применяется к объекту s типа string.

В данном случае size – это функция (метод), связанная с типом данных string.

Сравнение и копирование строк

- Сравнение:

  if (st == st3)

- Скопировать одну строку в другую можно с помощью обычной операции присваивания:

  st2 = st3; // копируем st3 в st2

Пример работы со строкой

```
#include <iostream>
using namespace std;
main()
{
	string s;
	int i;
	cout << "Введите строку: ";
	 getline ( cin, s );
	for ( i = 0; i < s.size(); i++ )
		if ( s[i] == 'а' )
			s[i] = 'б';
	cout << s;
}
```

Нумерация символов в строке начинается с нуля

Конкатенация строк

Для конкатенации строк используется операция сложения (+) или операция сложения с присваиванием (+=).

Пусть даны две строки:

```
string s1( "hello, " );
string s2( "world\n" );
```

Мы можем получить третью строку, состоящую из конкатенации первых двух, таким образом:

```
string s3 = s1 + s2;
```

Если же мы хотим добавить s2 в конец s1, мы должны написать:

```
s1 += s2;
```

МЕТОДЫ КЛАССА STRING

Выделение части строки – метод **substr**

```
s = "0123456789";
s1 = s.substr ( 3, 5 );
cout << s1 << endl;
```

Фрагмент копирует в строку s1 пять символов строки s (с 3-го по 7-й).

Этот метод принимает два параметра: номер начального символа и количество символов.

Если второй параметр при вызове substr не указан, метод возвращает все символы до конца строки. Например,

```
s = "0123456789";
s1 = s.substr ( 3 );
```

вернёт «3456789».

Удаление части строки - метод **erase**

```
s = "0123456789";
s.erase ( 3, 6 );
```

В строке s остаётся значение «0129» (удаляются 6 символов, начиная с 3-го).

Обратите внимание, что процедура erase изменяет строку.

Вставка символов в строку – метод **insert**

```
s = "0123456789";
s.insert ( 3, "ABC" );
```

Переменная s получит значение «012ABC3456789».

Поиск в строке - метод **find**

Эта функция возвращает номер найденного символа (номер первого символа подстроки) или –1, если найти нужный фрагмент не удалось.

фрагмент не удалось. Пример:

```
string s = "Здесь был Вася.";
int n;
n = s.find ( 'c' );
if ( n >= 0 )
 cout << "Номер первого символа 'c': " << n << endl;
else cout << "Символ не найден " << endl;
```

Преобразование строки в число

В C++ нет методов преобразования строки в число. Но можно воспользоваться функциями языка С. В этом случае сначала необходимо преобразовать тип string в обычную (анси) строку, а потом использовать функции **atoi и atof,** прототип которых находится **в stdlib.h**:

```
string s = "123.456";
int N;
double X;
N = atoi ( s.c_str() ); // N=123
X = atof ( s.c_str() ); // X = 123.456
```

Метод c_str() преобразует строку string в строку с завершающим нулем.

Преобразование происходит до первого символа, не относящегося к числу.