Занятие № 5.

Работа со строками

1. Ввод строк

Для консольного ввода строк мы использовали конструкцию

```
cin ≫ s;
```

Однако в этом случае в строку s попадет только первое слово, поскольку такой ввод действует только до первого разделителя, к которым относятся конец строки, табуляция и пробел.
Для того, чтобы ввести полностью строку нужно использовать

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

Для использования getline() необходимо подключить библиотеку для работы со строками

```
#include <string>
```

Пример:

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
```

```
    setlocale(LC_ALL, "Russian");

    string s;

    cout << "Введите строку > ";
    getline(cin, s);

    cout << "\nСпасибо!" << endl;
    cout << "Вы ввели: \"" << s << "\"" << endl;
    return 0;
}
```

Заметим, что в этой программе мы использовали endl как замену “\n”.

Рассмотрим программу, которая вводит целый текст, состоящий из нескольких строк. Будем вводить строки, а в конце мы наберем Ctrl+Z, что является признаком конца текста. Функция getline() возвращает true, если конец текста еще не достигнут, и false, если достигнут конец текста.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

    string Text = ""; // Строка для всего текста
    string s; // Текущая строка

    cout << "Вводите текст, а в конце наберите Ctrl+Z:" << endl;
```

```
    while(getline(cin, s))
    {
        Text = Text + s + "\n";
    }

    cout << "\nСпасибо!" << endl;
    cout << "Вы ввели:" << endl;
    cout << Text;

    return 0;
}
```

2. Доступ к символам строки

Доступ к элементам строки можно осуществлять как к элементам массива.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

    string s;

    cout << "Введите строку > ";
    getline(cin, s);

    cout << endl << "В этой строке " << s.length() << " символов"
<< endl;

    for (int i = 0; i < s.length(); i++)
```

```
    {
        cout << i << "-й символ: " << s[i] << endl;
    }

    return 0;
}
```

Для того, чтобы узнать сколько символов содержит строка s нужно воспользоваться конструкцией

```
s.length()
```

3. Работа с подстроками

Со строками в C++ можно выполнять различные операции. Рассмотрим поиск подстроки и замену подстрок в строке. Для поиска подстроки sub в строке s используем:

```
pos = s.find(substr);
```

Эта функция возвращает позицию подстроки, если подстрока найдена и -1, если подстрока не найдена.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");
```

```
    string s;
    string sub;
    int pos;

    cout << "Введите строку > ";
    getline(cin, s);


    cout << "\nВведите подстроку > ";
    getline(cin, sub);

    pos = s.find(sub);

    if (pos < 0)
    {
        cout << "\nПодстрока не найдена!" << endl;
    }
    else
    {
        cout << "\nПодстрока найдена на " << pos << "-й позиции!"
<< endl;
    }

    return 0;
}
```

Часто возникает необходимость извлекать подстроку из строки. Для этого используем функцию substr(int start, int count). Первым аргументом этой функции является начало вырезаемой подстроки, а второй аргумент – это количество вырезаемых символов. Если второй аргумент отсутствует, то извлекается подстрока до конца строки.

```
#include <iostream>
```

```
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

    string s;
    string sub;

    s = "One Two Three";

    sub = s.substr(4, 3);
    cout << sub << endl;

    sub = s.substr(4);
    cout << sub << endl;

    return 0;
}
```

Вставка и удаление подстрок. Для вставки в строку s подстроки sub, начиная с позиции n, используем функцию

```
s.insert(n, sub);
```

Вот пример:

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main()
{
    setlocale(LC_ALL, "Russian");

    string s;
    string sub;

    s = "One Three";
    sub = "Two ";

     s.insert(4, sub);

    cout << s << endl;

    return 0;
}
```

Для удаления в строке s подстроки, начинающейся с позиции n и длинной d, используем следующую функцию:

```
s.erase(n, d);
```

Пример:

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

    string s;
```

```
    s = "0123456789";

     s.erase(3, 5);

    cout << s << endl;

    return 0;
}
```

Работа с массивами строк.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

     int N = 3;

     string s[N];

     for (int n = 0; n < N; n++)
     {
          cout << "\nВведите слово > ";
          cin >> s[n];
     }

     for (int n = 0; n < N; n++)
     {
          cout << s[n] << "\t";
     }
```

```
     cout << endl;

    return 0;
}
```

4. Маленькие мелочи

Напомним, что в C++ часто используют сокращенную запись:

a += b; вместо a = a + b;

a -= b; вместо a = a - b;

a *= b; вместо a = a * b;

a /= b; вместо a = a / b;

Кроме того, в C++ существует тернарная условная операция, которая сокращает конструкции if … else. Эта операция выглядит следующим образом:

условие ? выражение1 : выражение2 ;

Если условие истинное, то выполняется выражение1, в противном случае выполняется выражение2.

Пример:

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main()
{
    setlocale(LC_ALL, "Russian");

    int a;
    cout << "Введите число > ";
    cin >> a;

    a >= 0 ? cout << "Неотрицательное число" : cout << "Отрицательное число";

    return 0;
}
```

Логические операции в C++:

«НЕ» – !

«И» – &&

«ИЛИ» – ||

Пример использования:

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    setlocale(LC_ALL, "Russian");

     int a, b, c;

     cout << "Введите три числа >";
```

```
    cin >> a >> b >> c;

    if (!(a > b))
    {
        cout << "\nЧисло a не больше b" << endl;
    }
    else
    {
        cout << "\nЧисло a больше b" << endl;
    }

    if (((a == b)&&(c < b))||(a < 0))
    {
        cout << "\nСложное выражение выполнено" << endl;
    }
    else
    {
        cout << "\nСложное выражение не выполнено" << endl;
    }

    return 0;
}
```