



# Семинар 8

## Ввод-вывод в языке Си++

# Ввод-вывод в языке Си++



Система ввода-вывода в Си++ построена на основе потоков. Поток – логическое понятие, которое можно воспринимать как последовательность байтов, не зависящую от устройства.

В Си++ каждый поток представлен объектом некоторого класса. Потоки делятся на: входные, выходные, двунаправленные.

# Ввод-вывод в языке Си++



Классы потоков ввода-вывода:

`istream` – класс входных потоков (`cin` – это объект этого класса)

`ostream` – класс выходных потоков (`cout`, `cerr`, `clog`)

`iostream` – класс двунаправленных потоков

# Флаги форматирования



boolalpha	Представлять логические значения в виде true и false
dec	Десятичная система счисления (по умолчанию)
hex	Шестнадцатеричная система счисления
oct	Восьмеричная система счисления
internal	Символ заполнения пустых позиций помещается между числовым значением и знаком числа
left	Выводимое значение выравнивается по левому краю
right	Выводимое значение выравнивается по правому краю
scientific	Для вещественных чисел указать мантиссу и порядок
showbase	Показать основание системы счисления
showpoint	При выводе вещественных чисел печатать точку и следующие за ней нули, если число имеет нулевую дробную часть
showpos	Печатать знак положительного числа

# Флаги форматирования



skipws	При чтении данных игнорировать начальные пробельные символы
unitbuf	Выгружать содержимое буфера после каждого ввода или вывода
uppercase	При выводе шестнадцатеричных чисел использовать буквы верхнего регистра

# Методы форматирования



<code>setf(flags1)</code>	Установить флаги
<code>setf(flags1, flags2)</code>	Установить флаги flags1, сбросить флаги flags2
<code>unsetf(flags1)</code>	Сбросить флаги
<code>flags()</code>	Возвращает значение всех установленных флагов типа <code>std::ios::fmtflags</code>
<code>flags(flags1)</code>	Устанавливает флаги в соответствии со значением аргумента, возвращает предыдущее значение флагов
<code>width(i)</code>	Задаёт минимальную ширину поля данных вывода
<code>width()</code>	Возвращает ширину поля вывода
<code>precision(i)</code>	Задаёт точность представления вещественных чисел (количество цифр после точки)
<code>precision()</code>	Возвращает точность представления вещественных чисел

# Методы форматирования



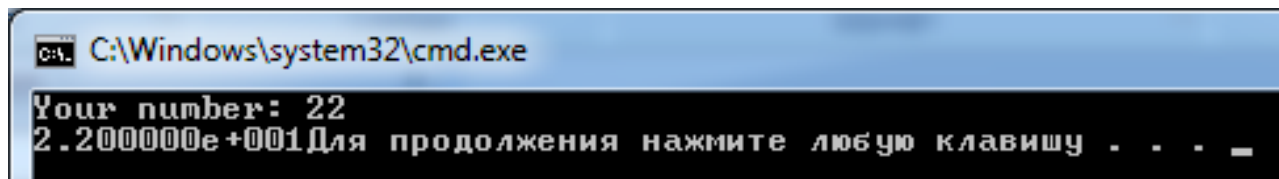
<code>fill()</code>	Возвращает текущий символ, размещаемый в незанятых позициях поля вывода
<code>fill(mychar)</code>	Устанавливает новый символ заполнения пустых полей



# Методы форматирования

Пример.

```
float f;  
cout << "Your number: ";  
cin >> f;  
cout.setf(ios::scientific);  
cout << f;
```



```
C:\Windows\system32\cmd.exe  
Your number: 22  
2.200000e+001Для продолжения нажмите любую клавишу . . .
```

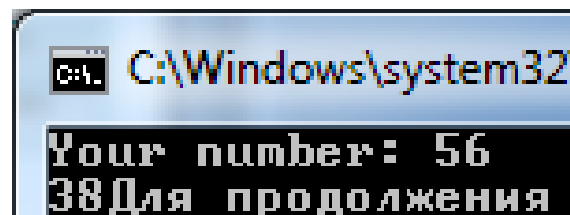




# Манипуляторы

Манипулятор – специальная функция, позволяющая программисту изменять состояние и флаги потока. Их имена и вызовы можно использовать в качестве правых операндов при использовании операций `<<` и `>>`. Например,

```
int i;  
cout << "Your number: ";  
cin >> i;  
cout << hex;  
cout << i;
```



# Манипуляторы



dec	При вводе и выводе устанавливает флаг десятичной системы счисления
hex	При вводе и выводе устанавливает флаг шестнадцатеричной системы счисления
oct	При вводе и выводе устанавливает флаг восьмеричной системы счисления
ws	Игнорирование пробельных символов при вводе
endl	Включение в выходной поток символа новой строки и сбрасывание буфера (действует только при выводе)
ends	Включение в выходной поток символа конца строки (действует только при выводе)
flush	Очистка выходного потока при выводе

# Манипуляторы



В <iomanip> описаны манипуляторы с параметрами

setbase(n)	Установка системы счисления (8, 10, 16)
resetiosflags(mask)	Сброс отдельных флагов
setiosflags(mask)	Установка отдельных флагов
setfill(mychar)	Установка символа заполнителя
setprecision(n)	Определение точности представления вещественных чисел
setw(n)	Задание минимальной ширины поля вывода

# Функции ввода, вывода, позиционирования



Вывод:

```
ostream& ostream::put(char ch);  
ostream& ostream::write(const char *arr,  
streamsize n);
```

Пример:

```
cout.put('\n').write("Hello, world",  
12).put('!')<<endl;
```

напечатает с новой строки «Hello, world!»

# Функции ввода, вывода, позиционирования



Позиционирование:

```
pose_type tellp();
```

//Возвращает текущую позицию

```
ostream& seekp(pose_type pos, seek_dir dir);
```

// Перемещает позицию записи в  
направлении, определённом dir (beg, cur или  
end)

# Функции ввода, вывода, позиционирования



Ввод:

```
istream& get(char * array, streamsize max_len,  
char delim= '\\n'); //max_len – максимальная  
длина прочитанных байтов
```

```
istream& get(streambuf& buf, char delim= '\\n');
```

```
istream& get(char & ch); //получает символ  
int_type get(); //получает код символа
```

# Функции ввода, вывода, позиционирования



Ввод:

```
istream& getline(char * array, streamsize  
max_len, char delim= '\n');
```

```
int peek(); //Позволяет взглянуть на  
очередной символ потока, при этом оставляя  
ЭТОТ СИМВОЛ В ПОТОКЕ
```

# Функции ввода, вывода, позиционирования



Пример программы, обменивающейся  
данными между потоками.

```
cout<<"Enter symbols:"<<endl;  
char cim;  
while(cin.peek()!='\n'){  
    cin.get(cim);  
    cout.put(cim);  
}  
return 0;
```



# Функции ввода, вывода, позиционирования



## Очистка входного потока

```
cin.clear();  
while (std::cin.get() != '\n');
```