ПОТОКИ ВВОДА-ВЫВОДА

- Основание системы счисления (ios::dec, ios::oct, ios::hex, ios::showbase)
- Класс ios содержит общие поля для ввода и ввывод поля и методы
- Класс istream – для входных потоков
- Класс ostream – для выходных потоков
- Класс iostream – для двунаправленные потоки
- istringstream, ostringstream, stringstream - строковые потоки
- ifstream, ofstream, fstream

Класс в заголовочных файлах

- Ios – базовый класс потоков ввода/вывода
- Iosfwd – предварительные объявления средств ввода/вывода
- Istream, ostream, iostream– операции с потоками ввода/вывода
- Fstream – потоки ввода/вывода в файлы
- Sstream – потоки ввода вывода в строки
- Streambuf – буферезация потоков ввода/вывода
- Iomanip – манипуляторы

СТАНДАРТНЫЕ ПОТОКИ

cin	Istream	
cout	Ostream	
cerr	ostream	Связано с экраном (небуферезированный вывод)
clog	ostream	Связано с экраном (стандартный буферезированный вывод)

Ссылка на объект

- << >> типа istream
- ПРИОРИТЕТЫ
- Приоритет сложения
- Логические операции ( )
- Cout<< (i<j)
- Извлечение должно разделяться пробельными символами

ВВОД СТРОК

Извлечение происходит до ближайшего пробела (вместо него в строку заносится нуль-символ)

Get

getline

Форматирование данных

- Флаг, манипулятор, форматирующие методы
- Флаг – отдельные биты, объединенные в поле x_flags типа long класса ios
- **cout.setf(ios::flag)**
- Методы flags, setf, unsetf

- Несколько флагов

cout.setf(ios::flag1 | ios::flag2 | ios::flag3)

Флаги форматированного типа

- Создайте программу вывода чисел с плавающей запятой ios::scientific, ios::fixed)

```
// fig11_25.cpp
// Использование флага ios::showbase
#include <iostream.h>
#include <iomanip.h>

main()
{
   int x = 100;

   cout << setiosflags(ios::showbase)
        << "Печать целых чисел с разными основаниями:"
        << x << endl
        << oct << x <<endl
        << hex << x << endl;
   return 0;
}
```

```
Печать целых чисел с разными основаниями:
100
0144
0x64
```

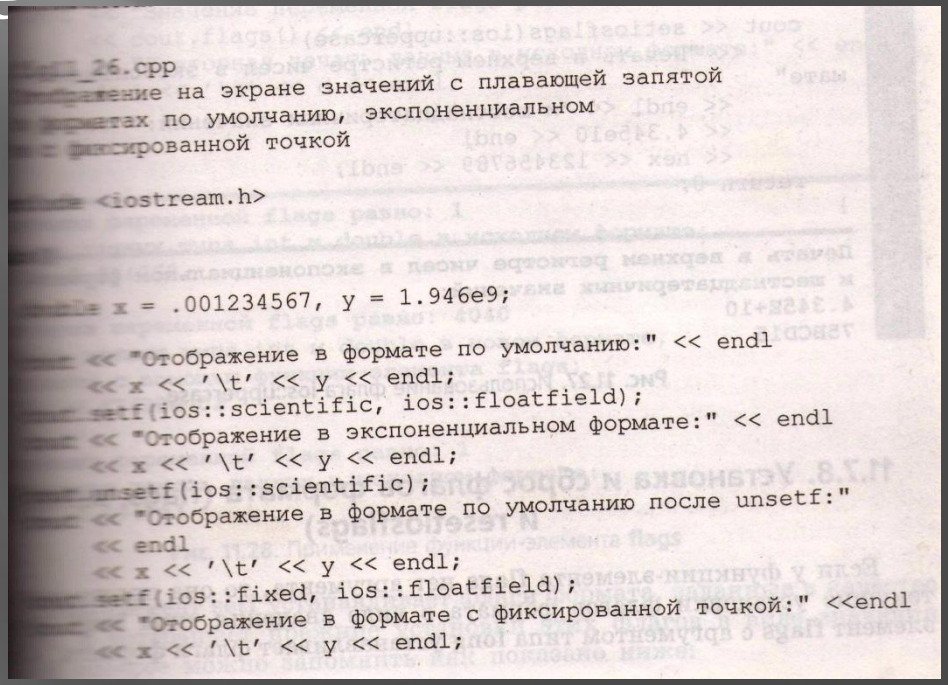
Рис. 11.25. Использование флага **ios::showbase**

- Управление выводом в нижнем и верхнем регистре (ios::uppercase) символов X и E в шестнадцатеричном и экспоненциальном форматах соответственно. Когда флаг ios:: uppercase установлен, все буквы шестнадцатеричного формата печатаются в верхнем регистре
- P/S в конце

Программы не

Дописано return 0;

}



```
_26.cpp
ение на экране значений с плавающей запятой
атах по умолчанию, экспоненциальном
ксированной точкой

<iostream.h>


x = .001234567, y = 1.946e9;

<< "Отображение в формате по умолчанию:" << endl
<< x << '\t' << y << endl;
setf(ios::scientific, ios::floatfield);
<< "Отображение в экспоненциальном формате:" << endl
<< x << '\t' << y << endl;
unsetf(ios::scientific);
<< "Отображение в формате по умолчанию после unsetf:"
<< endl
<< x << '\t' << y << endl;
setf(ios::fixed, ios::floatfield);
<< "Отображение в формате с фиксированной точкой:" <<endl
<< x << '\t' << y << endl;
```

- Создайте аналогичный пример
- Объясние: что такое система счиления
- Созадйте программку. Пример ниже
- .

```
// fig11_25.cpp
// Использование флага ios::showbase
#include <iostream.h>
#include <iomanip.h>

main()
{
   int x = 100;

   cout << setiosflags(ios::showbase)
        << "Печать целых чисел с разными основаниями:"
        << x << endl
        << oct << x <<endl
        << hex << x << endl;
   return 0;
}
```

```
Печать целых чисел с разными основаниями:
100
0144
0x64
```

Рис. 11.25. Использование флага **ios::showbase**

- Создайте программу вывода чисел с плавающей запятой ios::scientific, ios::fixed)

```
// fig11_25.cpp
// Использование флага ios::showbase
#include <iostream.h>
#include <iomanip.h>

main()
{
   int x = 100;

   cout << setiosflags(ios::showbase)
        << "Печать целых чисел с разными основаниями:"
        << x << endl
        << oct << x <<endl
        << hex << x << endl;
   return 0;
}
```

```
Печать целых чисел с разными основаниями:
100
0144
0x64
```

Рис. 11.25. Использование флага **ios::showbase**

Неформатированный ввод-вывод с использованием read, gcount и write

- char buffer [ ] = “ Все надоело!”;
- Cout.write(buffer, 12);

- failbit

- Gcount сообщает о количестве символов, прочитанной последней операцией ввода

Контрольная работа

1. Подставить ответы:
2. Заголовочный файл ___ содержит информацию для выполнения форматированного ввод-вывода
3. При использовании параметризированных манипуляторов должен быть включен заголовочный файл___
4. Заголовочный файл ___ содержит информацию для управления обработкой файлов
5. Манипулятор потока ____ осуществляет переход на новую строку в выходном потоке и сброс выходного потока
6. Заголовочный файл ____ позволяет использовать смешанный стиль программирования ввода-вывода языков С и С++
7. Функция-элемент _____ класса ostream используется для ввыполнения неформатированного вывода

8. Операция ввода поддерживается классом ___
9. Вывод в стандартный поток ошибок направляется в объекты потоков ____ или ____
10. Операции вывода поддерживаются классом ____
11. Для операции поместить в поток используется
12. символ___
13. Манипуляторы потока _ , ___ и ___ используются, чтобы задать востмиричный, шестныдцетиричный и десятичный форматы представления целых чисел
14. По умолчанию точность для представления чисел с плаввающей точкой равна ____
15. Установка флага _____ вызывает печать знака плюс для положительных чисел

Практические задания

- Создайте файлы input.txt, output.txt
- Выполните следующие задания с выводом в в output.txt:
- Установите флаг для вывода в верхнем регистре чисел в экспоненциальном формате и букв шестандцетиричном формате
- Введите адрес переменной string типа char *
- Установите флаг печати чисел с плавающей запятой в экспоннециальном формате

- Выведети адрес переменной ntegerPrt типа int

- Установите такой флаг, чтобы при выводе целых чисел отображалось их осмнование при предоставлении шестандцетиричном формате и восьмиричных форматах
- выведете значение типа float, которое указывает floatPtr
- Используйте функцию-элемент потока, чтобы установить символ ‘*’ в качестве заполняющего символа для печати с шириной поля, превышающей требуемую для печатаемого значения. Напишите отдельный оператор чтобы сделать то же самое с помощью манипулятора потока.

- Получите следующий символ из входного потока на удаляя его из потка
- Введите и отбросьте очередные шесть символов из входного потока
- Напечатайте 1234 с выравниванием по правой границе поля шириной 10 разрядов
- Используйте целые переменные x и y, чтобы задать ширину поля и точность используемые для отражения значения 57.4573 типа double и выведите эти значения

Найдите ошибки

- Cout <<“значение x <= :” <<x<=y;
- Cout << ““Строка в кавычках””;
- Для каждого из перечисленных ниже операторов, покажите, что будет выведено

 - cout <<“12345”<< endl;
   cout.width(5);
   Следуcout.fill(‘*’);
   cout<<123<<endl<<123

- ющий оператор доложен печатать целое значение ‘c’

  cout <<‘c’;

Подсказки

- Используйте функцию-элемент потока, чтобы установить символ ‘*’ в качестве заполняющего символа для печати с шириной поля, превышающей требуемую для печатаемого значения. Напишите отдельный оператор чтобы сделать то же самое с помощью манипулятора потока.

cout.fill (‘*’) cout<<setfill(‘*’)