**File Manipulators**

**Manipulators** are helping functions that can modify the input/output stream. It does not mean that we change the value of a variable, it only modifies the I/O stream using insertion (<<) and extraction (>>) operators.

For example, if we want to print the hexadecimal value of 100 then we can print it as:

cout<<setbase(16)<<100

**Types of Manipulators**

There are various types of manipulators:

1. **Manipulators without arguments**: The most important manipulators defined by the **IOStream library** are provided below.
   * **endl**: It is defined in ostream. It is used to enter a new line and after entering a new line it flushes (i.e. it forces all the output written on the screen or in the file) the output stream.
   * **ws**: It is defined in istream and is used to ignore the whitespaces in the string sequence.
   * **ends**: It is also defined in ostream and it inserts a null character into the output stream. It typically works with std::ostrstream, when the associated output buffer needs to be null-terminated to be processed as a C string.
   * **flush**: It is also defined in ostream and it flushes the output stream, i.e. it forces all the output written on the screen or in the file. Without flush, the output would be the same, but may not appear in real-time.

**Examples:**

|  |
| --- |
| #include <iostream>  #include <istream>  #include <sstream>  #include <string>    using namespace std;    int main()  {      istringstream str("           Programmer");      string line;      // Ignore all the whitespace in string      // str before the first word.      getline(str >> std::ws, line);        // you can also write str>>ws      // After printing the output it will automatically      // write a new line in the output stream.      cout << line << endl;        // without flush, the output will be the same.      cout << "only a test" << flush;        // Use of ends Manipulator      cout << "\na";        // NULL character will be added in the Output      cout << "b" << ends;      cout << "c" << endl;        return 0;  } |

**Output:**

Programmer

only a test

abc

1. **Manipulators with Arguments:** Some of the manipulators are used with the argument like setw (20), setfill (‘\*’), and many more. These all are defined in the header file. If we want to use these manipulators then we must include this header file in our program.

For Example, you can use following manipulators to set minimum width and fill the empty space with any character you want: std::cout << std::setw (6) << std::setfill (’\*’);

* + **Some important manipulators in *<iomanip>* are:**
    1. **setw (val):** It is used to set the field width in output operations.
    2. **setfill (c):** It is used to fill the character ‘c’ on output stream.
    3. **setprecision (val):** It sets val as the new value for the precision of floating-point values.
    4. **setbase(val):** It is used to set the numeric base value for numeric values.
    5. **setiosflags(flag):** It is used to set the format flags specified by parameter mask.
    6. **resetiosflags(m):** It is used to reset the format flags specified by parameter mask.
  + **Some important manipulators in <ios> are:**
    1. **showpos:** It forces to show a positive sign on positive numbers.
    2. **noshowpos:** It forces not to write a positive sign on positive numbers.
    3. **showbase:** It indicates the numeric base of numeric values.
    4. **uppercase:** It forces uppercase letters for numeric values.
    5. **nouppercase:** It forces lowercase letters for numeric values.
    6. **fixed:** It uses decimal notation for floating-point values.
    7. **scientific:** It uses scientific floating-point notation.
    8. **hex:** Read and write hexadecimal values for integers and it works same as the setbase(16).
    9. **dec:** Read and write decimal values for integers i.e. setbase(10).
    10. **oct:** Read and write octal values for integers i.e. setbase(10).
    11. **left:** It adjusts output to the left.
    12. **right:** It adjusts output to the right.

**Example:**

|  |
| --- |
| #include <iomanip>  #include <iostream>  using namespace std;    int main()  {      double A = 100;      double B = 2001.5251;      double C = 201455.2646;        // We can use setbase(16) here instead of hex        // formatting      cout << hex << left << showbase << nouppercase;        // actual printed part      cout << (long long)A << endl;        // We can use dec here instead of setbase(10)        // formatting      cout << setbase(10) << right << setw(15)           << setfill('\_') << showpos           << fixed << setprecision(2);        // actual printed part      cout << B << endl;        // formatting      cout << scientific << uppercase           << noshowpos << setprecision(9);        // actual printed part      cout << C << endl;  } |

**Output:**

0x64

\_\_\_\_\_\_\_+2001.53

2.014552646E+05