
Output

Section 3.8

Advanced Output Section 3.8

- How can we force output to look a particular way?
 - Precision of numbers
 - Spacing around output

```
Here are some floating point numbers:
```

```
72.0
```

```
72.00
```

```
72.000
```

```
Here is a table of data:
```

```
   4   cat   15
100   6   2.1
```

Outputting with Spacing

```
#include <iostream>
#include <iomanip> //New Library!
#include <string>

using namespace std;

int main()
{
    string name = "cs150";
    int integer = 42;

    cout << setw(6) << name << setw(6) << integer << endl;
    cout << setw(4) << integer << endl;

    return 0;
}
```

Setw

- Setw is not *sticky*
 - you must specify it every time

```
int integer = 42;  
cout << setw(6) << integer << integer << endl;
```

//output?

Practice

- Write a program segment that allows the user to input two integer values. Display both numbers as shown below, always displaying the smaller number first.

```
Please enter two numbers: 100 9
The numbers are:
                    9
                   100
```

Precision

```
const double PI = 3.141592653589793;  
cout << PI << endl; // default output
```

- What does this output?
- Precision

```
cout << setprecision(2) << PI;
```

Output:

Precision

- Precision can also be used to set the number of digits after the decimal point

```
const double PI = 3.141592653589793;  
cout << fixed << setprecision(2) << PI;
```

- Output:

Example

```
#include <iostream>
#include <iomanip> //New Library!
#include <string>

using namespace std;

int main()
{
    const double PI = 3.141592653589793;
    string name = "cs150";
    int integer = 42;

    cout << setw(6) << name << setw(6) << integer << endl;
    cout << setw(6) << fixed << setprecision(3) << PI;
    cout << setw(4) << integer << endl;

    return 0;
}
```


Precision of numbers

```
#include <iostream>
#include <iomanip> //New Library!

using namespace std;

int main()
{
    const double PI = 3.141592653589793;

    cout << PI << endl; // default output
    cout << fixed << setprecision(4) << PI << endl;
    cout << fixed << setprecision(3) << PI << endl;
    cout << fixed << setprecision(2) << PI << endl;
    cout << fixed << setprecision(1) << PI << endl;

    return 0;
}
```

Precision

- Precision and fixed are *sticky*
 - remains in effect until changed

```
const double PI = 3.141592653589793;  
cout << fixed << setprecision(4) << PI << endl;  
cout << setprecision(2) << PI << endl;  
cout << PI << endl;
```

// Output?

Practice

- Using the variables below, create the output shown:

```
const double PI = 3.141592653589793;  
string name = "cs150";  
string animal = "cat";  
string cover = "hat";  
int integer = 42;
```

A • represents a blank space

```
••••cat•3.1416  
••••hat••cs150  
•42••42••42•42  
3.14159265•3.1
```

Practice

- Write a program to output the following.
 - User input is in red. Next slide has example when user answers **N**.

Name ? **Bob**

Age ? **21**

Weight? **120.45**

Would you like your info to be right aligned? **Y**

Bob

21

120.5

Practice

Name ? **Bob**

Age ? **21**

Weight? **120.45**

Would you like your info to be
formatted? **N**

Bob

21

120.45