
Exponents & Output

page 85-87 & 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
```

Spacing

- How can we output data in a table?

```
string name = "cs150";
```

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

```
cs150 42 house  
3.1415 42 dog
```

Spacing around output

```
#include <iostream>
#include <iomanip> //New Library!
#include <string>
using namespace std;
int main()
{
    double number = 3.141592653589793;
    string name = "cs150";
    int integer = 42;
    cout << setw(6) << name << setw(6) << integer << endl;
    //cout << setw(6) << fixed << setprecision(3) << number;
    cout << setw(4) << integer << endl;
    return 0;
}
```

```
•cs150••••42
••42
```

A • represents a blank space

Setw

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

```
double number = 3.141592653589793;  
int integer = 42;  
cout << setw(6) << integer << integer << endl;  
cout << integer <<endl;  
  
//output?
```

Practice

- Write a program segment that allows the user to input two integer values into variables num1 and num2. Display both numbers as shown below, always displaying the smaller number first.

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

```
The numbers are:
```

```
  9
```

```
100
```

Precision

```
double number = 3.141592653589793;  
cout << number << endl; // default output
```

- What does this output?
- Precision

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

Output:

Spacing around output

```
#include <iostream>
#include <iomanip> //New Library!
#include <string>
using namespace std;
int main()
{
    double number = 3.141592653589793;
    string name = "cs150";
    int integer = 42;
    cout << setw(6) << name << setw(6) << integer << endl;
    cout << setw(6) << fixed << setprecision(3) << number;
    cout << setw(4) << integer << endl;
    return 0;
}
```

```
•cs150.....42
•3.142...42
```

A • represents a blank space

Precision

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

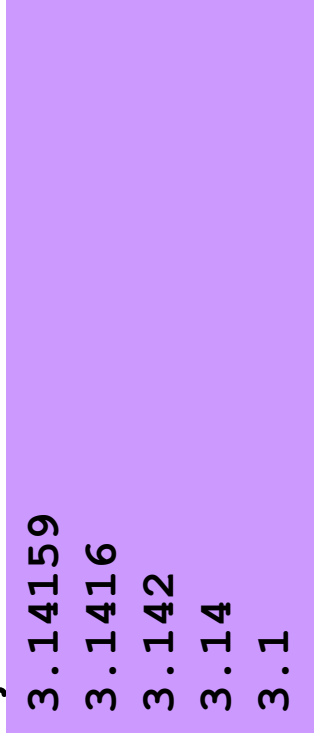
```
double number = 3.141592653589793;
```

```
cout << fixed << setprecision(2) << number;
```

- Output:

Precision of numbers

```
#include <iostream>
#include <iomanip> //New Library!
using namespace std;
int main()
{
    double number = 3.141592653589793;
    cout << number << endl; // default output
    cout << fixed << setprecision(4) << number << endl;
    cout << fixed << setprecision(3) << number << endl;
    cout << fixed << setprecision(2) << number << endl;
    cout << fixed << setprecision(1) << number << endl;
    return 0;
}
```



```
3.14159
3.1416
3.142
3.14
3.1
```

These numbers are *rounded!*

Explore on your own what happens if **number** is an integer.

Precision

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

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

// Output?

double

- a **double** has a range of:
 - $\pm 1.7\text{E}-308$ to $\pm 1.7\text{E}308$
 - however, only tracks 16 significant digits
- **double bignumber = 1234567891.123456789;**
cout << fixed << setprecision(20);
cout << bignumber << endl;
bignumber = 9234567891.123456789;
cout << bignumber << endl;
- Output:

Practice

- Using the variables below, create the output shown:

```
double number = 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
```

Exponents (page 85-87)

- The exponent operator was missing from the list! x^2 y^n
- C++ does not provide an exponent operator as part of the language
- Use `pow ()` in the `cmath` library

```
#include <cmath>
```

```
double area;
```

```
area = pow (4, 2); // area = 42
```

pow ()

- **pow ()** is not an operator
 - it is a *function*
 - like **main ()**
 - **double pow (double x, double y)**
 - it takes as arguments two doubles
 - **x** and **y**
 - it produces a **double**

Practice using exponents!

// Calculate the area of a square

double lengthOfSide = 4.9;

// Calculate the volume of a cube