

```
COUT - Examples

Literal constant of type cstring
cout << "Hello World!";

Simple arithmetic expression
cout << (num1 + num2) / 2;

Literal constant of type cstring followed by a variable
cout << "the average is "<< averageAge;
```

```
Examples

Given:

const char SCHOOL[11] = "Saddleback";
int num1, num2;
num1 = 3;
num2 = 7;

what will be the output for:
cout << num1;
cout << num1;
cout << num1;
cout << SCHOOL;
cout << "My school is " << SCHOOL;
cout << num1 << endl << endl << num2;
```

```
End line - endl

• endl → causes the cursor to go to the next line

What will this output?

const char SCHOOL[11] = "Saddleback";
num1 = 3;
num2 = 7;

cout << num1;
cout << num2 << endl << SCHOOL;
cout << "add 2 nums" << num1 + num2 << endl << endl;
cout << "subtract 2 nums" << "num2 - num1";

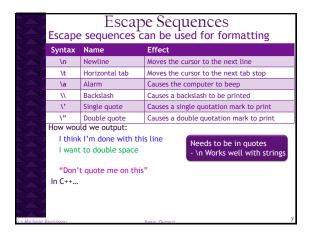
OUTPUT
```

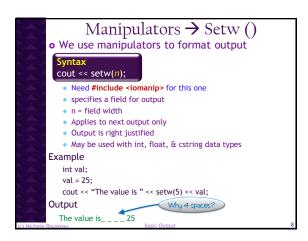
```
endl

We wanted:

3
7
Saddleback
add 2 nums 10
subtract 2 nums 4

How do we fix it?
cout << num1;
cout << num1;
cout << num1 + num2 << endl << endl;
cout << "add 2 nums" << num1 + num2 << endl << endl;
cout << "subtract 2 nums" << num1 - num1";
```





```
Setw() – right and left
 You can change the justification within the setw() field using the left and
 right operators.
Once these are set - they remain in effect for all succeeding output.
  cout << left:</p>
       ■ Changes the justification to left justified
   cout << right;</li>
       ■ Changes it back to the default
cout << setw(10) << left << "Steve" << 32 ;
   10 spaces \rightarrow The next output will be on the 11th column eve 32:
             These 5 columns are padded with spaces
           matter
cout << right << setw(10) << "Steve" << 32 ;
← 10 spaces → Steve 32
                     The next output will be on the 11th column
         These 5 columns are padded with spaces
```

```
Setw() example

o If you want output to look like this:

25 spaces → 11 spaces → BALANCE DUE

Jean Rousseau $ 32.32
Steve Woolston $ 1423.20
Chris Carroll $ 32.36

Use setw() → much easier to adjust than spaces or tab

cout << setw(25) << left << "NAME" << setw(11) << right << "BALANCE DUE" << endl;
cout << setw(25) << left << "" *< setw(11) << right << "BALANCE DUE" << endl;
cout << setw(25) << left << "" *< setw(11) << right << "ball > on't use "
cout << setw(25) << left << name1 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name2 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << "$" << setw(10) << right << ball > on't use "
with setv
cout << setw(25) << left << name3 << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << ball > on't use "
with setv
cout << setw(10) << right << setw(10) <
```

```
How can we format cout/cin pairs?

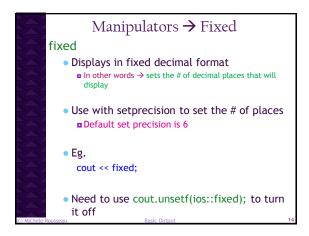
Example:
Enter your name: Bill Ding
Balance Due: 32.5
```

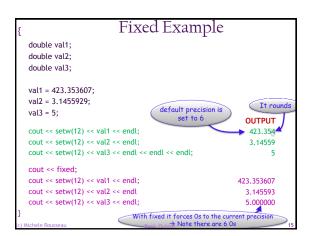
Formatting floating point values

Decimals can be formatted to your specific needs

#include <iomanip>
 → you need this for the next 3 manipulators

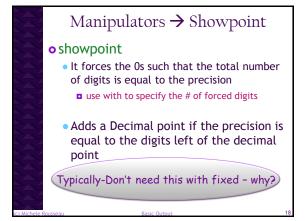
• fixed
• setprecision(n)
• showpoint





Manipulators → Set precision setprecision(n) • Controls the # of significant digits displayed to n digits • Before and after the decimal • Used with >> fixed • It displays the # of significant digits to the right of the decimal • Default precision is 6 digits • If there are more digits to the right of the decimal is greater than the n digits specified in setprecision(n) • The output will be rounded • If there are more digits to the left of the decimal than the output will be displayed in exponential notation

Setprecision Example val1 = 423.353607: default precis Without fixed it sets the precision w.r.t all digits val2 = 3.1455929; val3 = 5: **OUTPUT** cout << setw(9) << val1 << endl; 423.354 cout << setw(9) << val2 << endl; 3.14559 cout << setw(9) << val3 << endl << endl << endl; cout << setprecision(2);</pre> cout << setw(9) << val1 << endl; 4.2e+002 cout << setw(9) << val2 << endl; cout << setw(9) << val3 << endl << endl << endl; 5 cout << fixed: cout << setw(9) << val1 << endl; 423.35 cout << setw(9) << val2 << endl; 3.15 cout << setw(9) << val3 << endl; **5.00** With fixed it sets the # of decimal places
is EQUAL to the precision - NOTE how the decimal points line u



```
Showpoint Example
val1 = 423.353607;
                               Showpoint forces the
Os to the right of the decimal so # of digits
displayed is = to the precision
val2 = 3.1455929;
val3 = 5;
val4 = 75;
                                                               OUTPUT
cout << showpoint;</pre>
cout << setw(9) << val1 << endl;
                                                               423.354
cout << setw(9) << val2 << endl;
                                                               3.14559
cout << setw(9) << val3 << endl
                                                               5.00000
cout << setw(9) << val4 <<endl << endl;</pre>
                                                               75.0000
cout << setprecision(2);</pre>
cout << setw(9) << val1 << endl;
                                                              4.2e+002
cout << setw(9) << val2 << endl;
                                                                    3.1
cout << setw(9) << val3 << endl;
                                                                    5.0
cout << setw(9) << val4 << endl;
                           Set precision is w.r.t the # of digits
```

```
Exercise 1
#include <iomanip>
double num1;
double num2;
double num3;
num1 = 1233.2141112;
num2 = 2.09299;
                                 What will the output be?
num3 = 34;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << showpoint;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setprecision(3);</pre>
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << fixed:
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
```

```
#include <iomanip>
double num1;
double num2;
double num3;

num1 = 1233.2141112;
num2 = 2.09299;
num3 = 34;

cout << setprecision(3);
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << fixed;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num3 << endl;
```

```
#include <iomanip>
double num1;
double num2;
double num3;
num1 = 1233.2141112
num2 = 2.09299
num3 = 34;

cout << fixed;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << setw(15) << num3 << endl;
cout << setw(15) << num1 << setw(15) << num2 << setw(15) << num3 << endl;
cout << endl;
cout
```