Inputs and Outputs

1. Outputs

Use the `<cout>` command to output something to the screen. The input and output commands are defined in the iostream library. So in order to use this library your program needs to contain the following two lines:

```
#include <iostream>
using namespace std;
```

at the start of your program like this:

```
#include <iostream>
using namespace std;

int main ()
{
    // your code here
    return 0;
}
```

The `cout` command is used in conjunction with the insertion operator, which is written as `<<` (two "less than" signs). For example:

```
cout << "Hello World";
// prints the words Hello World on the console

cout << 120;
// prints the number 120 on the console

cout << 2 + 3;
// prints the number 5 on the console

cout << x;
// prints the content of variable x on the console

cout << x + 5;
// prints the result of adding 5 to whatever value is in x
```

The insertion operator (`<<`) may be used more than once in a single `cout` statement as shown next:

```
cout << "Hello, " << "I am " << "a C++ statement";
```

```
cout << "Hello, I am " << age << " years old and my zipcode is " << zipcode;
```

This first statement would print the message

```
Hello, I am a C++ statement
```

on the console screen.

In the second statement, if we assume that the `age` variable contains the value 24 and the `zipcode` variable contains the value 92515, then the output would be:

```
Hello, I am 24 years old and my zipcode is 92515
```
cout does not add a new line break after its output unless we explicitly indicate it using the new line keyword endl or \n (backslash, n). For example,

```cpp
cout << "First line.";
cout << "Second line.";
cout << "Third line.";
```

will produce the following output:

```
First line. Second line. Third line.
```

Whereas, this:

```cpp
cout << "First line." << endl;
cout << "Second line.\nThird line.";
```

and

```cpp
cout << "First line." << endl << "Second line.\nThird line.";
```

and

```cpp
cout << "First line." << endl << "Second line." << endl << "Third line.";
```

will produce the same output:

```
First line.
Second line.
Third line.
```
You can also tell `cout` to display numbers in different formats such as how many digits after the decimal point. For example

```cpp
float price;
price = 78.5;
cout << "The price is $" << price << endl;
```

will print out:

The price is $78.5

Whereas, this

```cpp
float price;
price = 78.5;
cout.setf(ios::fixed);
cout.setf(ios::showpoint);  // use a decimal point
cout.precision(2);   // specify two digits after the decimal point
cout << "The price is $" << price << endl;
```

will print out:

The price is $78.50

Notice that the ending zero is automatically added.

You can also use `setw` within the `cout` command to specify how many spaces to use in the printout. The string after the `setw` will be right justified in the allotted spaces. To use `setw`, you need to include `iomanip`.

```cpp
#include <iostream>
#include <iomanip>  // need to include iomanip to use setw
using namespace std;

int main()
{
    cout << "12345678901234567890" << endl;
    // reserve 12 spaces for the name and 8 spaces for the age
    cout << setw(12) << "Name" << setw(8) << "Age" << endl;
    cout << setw(12) << "Jonathan" << setw(8) << "18";
    return 0;
}
```

The above program will print out:

```
12345678901234567890
Name    Age
Jonathan  18
```
2. Inputs

Use the `cin` command to input something from the user to the program. The arrows, however, point in the opposite direction (>>). The arrows are followed by a variable which will be used to store the data that you typed in.

```cpp
int age;
cin >> age;
```

The `cin` command processes the input from the keyboard only after the Return key has been pressed. Note that on the Mac keyboard, you have to use the Return key on the main keyboard and NOT the Enter key on the numeric keypad.

You can also use `cin` to request more than one input from the user:

```cpp
int a, b;
cin >> a >> b;
```

is equivalent to:

```cpp
int a, b;
cin >> a;
cin >> b;
```

In both cases the user must enter two values, one for variable `a` and another one for variable `b` that may be separated by a space, a tab character or the Return key. For each variable specified in the `cin` command, it will read characters from the keyboard until a white space or the Return key is reached. So if you execute the following code and enter John Doe as the input

```cpp
string name;
cout << "Enter your name? ";
cin >> name;
cout << "Your name is " << name << endl;
```

you will be surprised that only the first name John was read as seen next:

```
Enter your name? John Doe
Your name is John
```

In order to get entire lines (including white spaces), we can use the function `getline`, which gets the entire line with all the white spaces up to the Return key.
Notice in the above example when you enter your age, the number is stored as a string and not as a number. If you need to do some calculations with that number, you will first have to convert the string to an actual number using the command stringstream as shown next:

```cpp
#include <iostream>
#include <sstream>    // need to include this library to use stringstream
using namespace std;

int main ( ) {
    string mystring;
    float price;
    int quantity;

    cout << "Enter price: ";
    getline (cin, mystring);
    stringstream (mystring) >> price;     // convert mystring to a number

    cout << "Enter quantity: ";
    getline (cin, mystring);
    stringstream(mystring) >> quantity;   // convert mystring to a number

    cout << "Total price: " << price * quantity << endl;  // notice the * is used for multiplication
    return 0;
}
```

Alternatively, you can directly enter the numbers into an integer or float variable instead of to a string variable, so that no conversion is necessary as shown next:

```cpp
#include <iostream>
using namespace std;

int main ( ) {
    float price;
    int quantity;

    cout << "Enter price: ";
    cin >> price;
    cout << "Enter quantity: ";
    cin >> quantity;
    cout << "Total price: " << price * quantity << endl;
    return 0;
}
```
3. `printf` and `scanf`

You can also use the `printf` command to output to the screen and `scanf` command to input from the keyboard\(^1\).

The following program will first ask you to type in your name and your age, and after you have typed in your name and age, it will print out “Hello” followed by your name and then print out how old you are.

```c
char name[20];
int age;

printf("Enter your name? ");
scanf("%s", name);
printf("Enter your age? ");
scanf("%i", &age);
printf("Hello %s, you are %i years old.\n", name, age);
```

A sample output from the program is shown next:

```
Enter your name? Jane
Enter your age? 18
Hello Jane, you are 18 years old.
```

A white space terminates the `scanf` command. So where it asks for your name and you enter your first and last name separated with a space, you will get the following wrong result instead:

```
Enter your name? Jane Doe
Enter your age? Hello Jane, 0 years old.
```

Notice that the last name is missing in the output and the age is 0. The reason is that the last name Doe is actually assign into the integer variable age, so age gets the value 0.

\(^1\) Technically, these are C commands and not C++ commands.
Notes on using the `printf` command:

- When printing out a variable, you need to use a formatting character (see table below) as a place holder for that variable. For example, the next command prints out the value in the variable `age` at the location where the `%i` is at within the string.

  ```c
  printf("Your age is %i.\n", age);
  ```

Notes on using the `scanf` command:

- Need to include a formatting character within double quotes similar to the `printf` command.

- When entering a string such as your name, the variable cannot be of type string. It must be of type `char` with the maximum number of characters specified in square brackets as in

  ```c
  char name[20];
  ```

- When entering a number such as your age, you need to have an ampersand sign `&` in front of the variable as in

  ```c
  scanf("%i", &age);
  ```

Notice that there is no `&` used for the `printf` command as in

  ```c
  printf("%i", age);
  ```
Formatting Characters

<table>
<thead>
<tr>
<th>Character</th>
<th>Output</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>Character</td>
<td>a</td>
</tr>
<tr>
<td>d or i</td>
<td>Signed decimal integer</td>
<td>392</td>
</tr>
<tr>
<td>f</td>
<td>Decimal floating point</td>
<td>392.65</td>
</tr>
<tr>
<td>g</td>
<td>Use the shorter of %f</td>
<td>392.65</td>
</tr>
<tr>
<td>s</td>
<td>String of characters</td>
<td>sample</td>
</tr>
<tr>
<td>%</td>
<td>A % followed by another % character will print out an %</td>
<td></td>
</tr>
</tbody>
</table>

Examples with printf

```c
printf ("Characters: %c \n", 'a');
printf ("Integers: %i \n", 2013);
printf ("Preceding with blanks: %10i \n", 2013); // use a total of 10 spaces
printf ("floats: %4.2f \n", 3.1416); // use a total of 4 spaces and two digits after
printf ("Width tricks: %*i \n", 5, 10); // use 5 spaces
printf ("%s \n", "A string");
```

And here is the output:

```
Characters: a
Decimals: 2013
Preceding with blanks: 2013
floats: 3.14
Width tricks: 10
A string
```
4. Exercises

1. This cout line

```cpp
cout << "Hello, " << "I am " << "a C++ statement";
```

was used to print out the message

```
Hello, I am a C++ statement
```

Simplify this cout line (make it shorter) to print out the same message.

2. What happens when you use an integer variable in a cin command and the user enters a decimal number such as 3.1415?

3. What happens when you use an integer variable in a cin command and the user enters a string such as Riverside?

4. Write a program using cin and cout to enter two different kinds of fruits, say apple and orange, then print out the statement “I like to eat apple but not orange!” Your program needs to replace the words apple and orange with whatever fruits the user enters in.

5. Write a program using cin and cout to enter two integer numbers and then print out the sum of these two numbers using the integer format.

6. Write a program using scanf and printf to enter two floating point numbers and then print out the sum of these two numbers using the floating point format.

7. Write a program using getline and cout to enter four integer numbers and then print out the product of these four numbers using the integer format.

8. Write a program using getline and cout to enter the address information for a person, i.e., name, address, city, state and zip code, and then print out the address correctly formatted as shown below.

   ```
   John Doe
   123 Magnolia Street
   Riverside, CA 92515
   ```

9. Write a program using getline and cout to enter the names of three people and their telephone numbers, and then print them out one per line nicely formatted in two columns. Hint: use setw to align the two columns.

10. Write a program to enter one number and then print out the multiplication table from 1 to 10 for that one number. For example, if you enter 3, then print out 1 x 3 = 3, 2 x 3 = 6, etc. on separate lines.