C++ loops and functions
The "for" loop

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```

- The loop is repeated as long as the loop condition is true.

```cpp
for (i=0; i<10; i++) {
    cout << "i is " << i << endl;
}
```
The "while" loop

- The loop is repeated while the loop condition is true.
- Note that you need to initialize variables before the loop and increment/decrement variables inside the loop.

```cpp
while (i < 10) {
    cout << "i is " << i << endl;
    i++;
}
```

Steps to repeat

```cpp
while (condition)
{
    // steps to repeat
}
```
Don't forget the final semicolon!

The loop is repeated while the loop condition is true.

```cpp
while (i < 10) {
    i++;
    cout << "i is " << i << endl;
}
```

```cpp
while (true) {
    do {
        i = 0;
        while (loop condition) {
            steps to repeat
        }
    } while (false);
}
```

The "do-while" loop
for vs. while

```cpp
for (i=10; i>0; i--)  {
    cout << i << "...";
}cout << "Blast off!" << endl;

while (i>0)
{
    cout << i << "...";
    i--;
}cout << "Blast off!" << endl;
```

for vs. while
C++ functions
some "built-in" functions

• double sqrt(double x)
• int abs(int x)
• double fabs(double x)
• double pow(double a, double n)
• void srand(unsigned int n)
• int rand()
• int time(int *p)
• int floor(double x)
• int ceil(double x)

returns the square root of x
returns the absolute value of x
returns the largest integer <= x
returns the smallest integer >= x
uses n as the seed for rand
returns a random integer
returns the number of seconds since 00:00:00 January 1, 1970
```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main()
{
    int i, num;
    srand(time(0));
    cout << "The next random number between 1 and 10 is " << num = rand() % 10 + 1; 
    cout << num << endl;
}
```

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

int main()
{"return
  {0
    {
      " << "Hello, World!" << endl;
    }
  }

  int i;
  double x, y;
  for(i=0; i<20; i++)
    {
      x = static_cast<double>(i);
      y = sqrt(x);
      cout << "The square root of " << x;
      cout << " is " << y << endl;
    }

  for(i=0; i<20; i++)
    {
      x = static_cast<double>(i);
      y = sqrt(x);
      cout << "The square root of " << x;
      cout << " is " << y << endl;
    }

  return 0;
"}
The square root of 0 is 0
The square root of 1 is 1
The square root of 2 is 1.41421
The square root of 3 is 1.73205
The square root of 4 is 2
The square root of 5 is 2.23607
The square root of 6 is 2.44949
The square root of 7 is 2.64575
The square root of 8 is 2.82843
The square root of 9 is 3
The square root of 10 is 3.16228
The square root of 11 is 3.31662
The square root of 12 is 3.46410
The square root of 13 is 3.60555
The square root of 14 is 3.74166
The square root of 15 is 3.87298
The square root of 16 is 4
The square root of 17 is 4.12311
The square root of 18 is 4.24264
The square root of 19 is 4.3589
The square root of 20 is 4.47214
• A function in C++ is like a method in Java.
• One passes info into a function and the function then does something with that info.

double circle_area(double radius) {
  double area;
  area = 3.14159 * radius * radius;
  return area;
}

double circle_area(double radius) {
  return area;
  area = 3.14159 * radius * radius;
}

• A function in C++ is like a method in Java.
• Program-defined Functions
```cpp
#include <iostream>
using namespace std;

double circle_area(double radius);

int main()
{
  double radius;
  cout << "Enter a radius: " << endl;
  cin >> radius;
  cout << "The area of a circle with radius " << radius << endl;
  return 0;
}

double circle_area(double radius)
{
  double area;
  area = 3.14159 * radius * radius;
  return area;
}
```

The code defines a function `circle_area` that calculates the area of a circle given its radius. It also includes a `main` function that prompts the user to enter a radius, then calls `circle_area` with the entered radius to calculate and print the area.
Local variables

- Defined within a function (main is a function, too)
- Only available to the function in which they are defined
- Most of the variables you work with will be local variables
```
#include <iostream>
#include <iostream>
using namespace std;

int foo(int n);

int main()
{
    int i, j, k;
    i = foo(j);
    return 0;
}

int foo(int n)
{
    int m;
    m = 2*n;
    return m;
}
```

Global constants

• A global constant is defined outside of any function (including main).
• A global constant can be used anywhere in the program.
• Convention is to name global constants with all caps.

Global constants
```cpp
#include <iostream>
using namespace std;
const double PI = 3.14159;
const int MAX_VALUE = 10;

using namespace std;

// main function

// defined functions

// defined functions

// function prototypes for programmer

// function prototypes for programmer

// other include directives
#include <iostream>
```
Homework

- Finish Exercise 3
- Start Exercise 4 (see website)
- Continue reading Chapter 1 of C&P, we will discuss it on Friday