This is an unusual paragraph. I'm curious as to just how quickly you can find out what is so unusual about it. It looks so ordinary and plain that you would think nothing was wrong with it. In fact, nothing is wrong with it! It is highly unusual though. Study it and think about it, but you still may not find anything odd. But if you work at it a bit, you might find out. Try to do so without any coaching!

Loops:
while
do…while
for

Luke Sathrum – CSCI 20

Today's Class

- From Branching to Looping
- **`while`** loops
- **`do`**...**`while`** loops
- **`for`** Loops
- Loop Pitfalls

Introduction to Loops

Repetition in C++

- How would output the same text x amount of times?
- Currently we have a couple different approaches
- For each approach assume we have the following:

```
cout << "How Many Times? ";
int count = reader.readInt();
```

Repetition in C++

- Approach #1

```
if (count == 1)
  cout << "Are we there yet?\n";
if (count == 2)
  cout << "Are we there yet?\n";
  cout << "Are we there yet?\n";
if (count == 3)
  cout << "Are we there yet?\n";
  cout << "Are we there yet?\n";
  cout << "Are we there yet?\n";
```

Repetition in C++

- Approach #2

```
if (count >= 1)
  cout << "Are we there yet?\n";
if (count >= 2)
  cout << "Are we there yet?\n";
if (count >= 3)
  cout << "Are we there yet?\n";
if (count >= 4)
  cout << "Are we there yet?\n";
```

Repetition in C++

- Approach #3

```
if (count >= 1) {
  cout << "Are we there yet?\n";
  count--;
}
if (count >= 1) {
  cout << "Are we there yet?\n";
  count--;
}
```

Repetition in C++

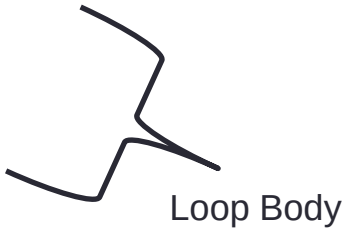
- The last example is the closest to how we write loops in C++

```
while (count >= 1) {
   cout << "Are we there yet?";
   count--;
}
```

Loops

- Loops allow us to repeat a statement
- Also allows us to repeat multiple statements
- Loop Body
  - The code that is repeated
  - Enclosed in brackets **{ }**

```
{
   cout << "Are we there yet?";
   count--;
}
```



Loop Iteration

- Each repetition is called an Iteration
- We can say we're on iteration 1, iteration 2
- We can also count how many times a loop is going to iterate

```
int loop_int = 5;
while (loop_int > 1) {
  loop_int--;
}
```

- How many iterations will this loop have?

Summary

- We use loops to repeat code
- The code that is repeated is called the loop body
- We call each time we loop an iteration
  - We can count these
  - Allow us to know how many times code will be repeated

The while Loop

`while` Loop

- One of the loop structures we have available to us in C++
- Allows us to repeat code until a condition is met
- Loops while the Boolean Expression is `true`
- Just like an `if` statement we need `{ }` if we have more than one statement we want to loop

`while` Loop - Syntax

```
while (Boolean_Expression)
 statement;

while (Boolean_Expression) {
 statement_1;
 statement_2;
}
```

while Loop - Example

```
int i = 1;
while (i <= 10) {
   cout << i << " ";
   i++;
}
```

Summary

- **`while`** loops allow us to loop until a condition is met
- Uses the keyword **`while`**
- Just like an **`if`** statement we need curly braces if we want to loop more than one statement

Sample Code

- Our first loop, a **`while`** loop
  - `while_loop.cpp`

The do...while Loop

`do...while` Loop

- The second of the loop structures we have available to us in C++
- Allows us to repeat code until a condition is met
- Loops while the Boolean Expression is `true`
  - Runs the code at least once
- Just like an `if` statement we need `{ }` if we have more than one statement we want to loop

do...while Loop - Syntax

```
do
  statement;
while (Boolean_Expression);

do {
  statement_1;
  statement_2;
} while (Boolean_Expression);
```

`while` vs. `do...while`

- Difference is when Boolean Expression is checked
- `while` checks Boolean Expression ***First***
  - If `true` we enter the loop body
  - If `false` we skip the loop body
- `do-while` always executes the loop body
  - On next iteration we check the Boolean Expression
    - If `true` we execute another iteration
    - If `false` we continue with our program

do...while Loop - Example

```
int i = 15;
do {
   cout << i << " ";
   i++;
} while (i <= 10);
```

Summary

- **`do-while`** loops are very similar the **`while`** loops
- The difference
  - We always run our code at least once

Sample Code

- Writing our first **`do`**...**`while`** loop
  - `do_while.cpp`

The for Loop

`for` Loop

- The third (and last) of the loop structures we have available to us in C++
- Allows us to repeat code a set amount of times
- Just like an `if` statement we need `{ }` if we have more than one statement we want to loop

for Loop - Syntax

```
for (Init_Action; Bool_Exp; Update_Action)
 statement;

for (Init_Action; Bool_Exp; Update_Action) {
 statement1;
 statement2;
}
```

For Loops – Initializing Action

- We assign an integer (typically) value to a variable
- We can declare the variable here as well
- EX
  - `i = 1;`
  - `int i = 1;`

For Loops – Boolean Expressions

- We write a Boolean Expression we want to evaluate
- Example
  - `i <= 10;`

For Loops – Update Action

- We update our counting variable
- Updates the variable after each iteration
- Example
  - `i++`
- NOTE
  - There is NO semicolon on the update action

For Loop Example

```
int sum = 0;
for (int i = 1; i <= 10; i++) {
   sum = sum + i;
}
cout << "The sum is: " << sum << endl;
```

Summary

- `for` loops allow us to choose how many time we want to loop
- Contain
  - Initialization Action
  - Boolean Expression
  - Update Action

Sample Code

- Writing our first `for` loop
  - `for_loop.cpp`

Loop Pitfalls

For Loop Problem – Extra Semicolon

```
for (int count = 1; count <= 10; count++);
  cout << "Hello\n";
```

- What is happening?
- **;**
  - Empty Statement
- Without **{ }** how many statements are part of the loop body?
- What does this loop do?

Infinite Loops

- Loops do not terminate until?
- It is easy to write a loop the will never stop looping
- These are called infinite loops
- Ex

```
int number = 5;
while (number != 2) {
  cout << "Odd";
  number -= 2;
}
```

Summary

- Loops can cause us problems
- The two main ones
  - The empty statement
  - Infinite Loops

Sample Code

- Common Problems with Loops
  - `loop_pitfalls.cpp`

Review

- Loops allow us to repeat code
- Three available to us
  - `while`
  - `do`…`while`
  - `for`
- Watch out for infinite loops and the empty statement

Fizz Buzz

- Fizz
  - Numbers divisible by 3
- Buzz
  - Numbers divisible by 5
- FizzBuzz
  - Numbers divisible by 3 and 5