Let's all Repeat Together

Last Time

- We
  - Looked at complex examples that use the **`if`** selection structure
  - Covered the **`if/else`** selection structure
  - Learnt about the **`?:`** conditional operator
  - Learn how to nest if/else selection structures
- Today we will
  - Start looking at examples of repetition structures

Problem

- 9.6: Write a program that displays a letter grade corresponding to an exam score

  90 - 100 A

  80 - 89 B

  70 - 79 C

  60 - 69 D

  0-59 F

Repetition Structures

- All the C++ programs that we have seen so far are executed only once before the program terminates
- However, it is often the case that programmers would like to specify that an action continue repeating while a condition is true
- This is achieved by using repetition structures, also called loops

An Example of Repetition

- An example of where we might need to use repetition is if we are calculating the average grade of a class of students
- We would need to continue reading in student grades until we have covered all students
- In pseudocode this might be:

```
While there are more students in the class
   Ask for student grade
```


C++ Example

- Write the segment of C++ code that will sum five numbers entered by the user

```
int sum, counter, num;

sum = 0;
counter = 0;

while( counter < 5 )
{
  cout << "Enter a number: ";
  cin >> num;

  sum = sum + num;
  counter = counter + 1;
}

cout << "The sum of your numbers is: " << sum;
```

`while` Repetition Structure

```
int sum, counter, num;

sum = 0;
counter = 1;

while( counter <= 5 )
{
  cout << "Enter a number: ";
  cin >> num;

  sum = sum + num;
  counter = counter + 1;
}

cout << "The sum of your numbers is: " << sum;
```

Initialize LCV

Loop Control Variable

Change the value of counter

`while` loops

- The syntax for `while` loops is

```
while (condition is true)
  statement;

while (condition is true)
{
  statement1;
  statement2;
  ...
}
```


Key Ingredients of `while` loops

- Initialize

  MUST initialize loop control variable
- Test

  The value of the loop control variable is tested during each iteration of loop
- Update

  Loop control variable is changed during each loop iteration

If any one of these is missing or incorrect, your loop won't run properly--not at all, too many/few times or infinitely.

Problems

- Write a while loop that outputs each integer from 1 to 5
- What's the output for x = 2? 3? 5?

```
cout << "Enter an integer";
cin >> x;
product = x;
count = 0;
while (count < 4)
{
  cout << product << endl;
  product *= x;
  count += 1;
}
```


Problems

- Write a program that reads in the salary of 5 employees and calculates the gross pay
  - We know, before the program runs, how many times the loop will iterate
  - Counter-controlled repetition
- Write a program that reads an undetermined number of student grades and calculates the average student grade
  - We don't know, before the program runs, how many times the loop will iterate
  - Sentinel-controlled repetition

Counter-Controlled Repetition

- We know, before we run the program, the number of repetitions that the loop will make
- Also called definite repetition
- Write a program that reads in the salary of 5 employees and calculates the gross pay

Sentinel-Controlled Repetition

- We have no idea how many times the loop will need to iterate
- Write a program that reads an undetermined number of student grades and calculates the average student grade
- How will we know when we've read all employee's salaries?
  - i.e. How will we know when to stop looping?

Sentinel-Controlled Repetition

- Use a sentinel value
  - User types employee salaries until all legitimate salaries have been entered
  - User then types in sentinel value to indicate that there are no more legitimate salaries
- Also called indefinite repetition
- Sentinel value must be chosen so that it cannot b confused with legitimate inputs
  - -1 is a good value to use in most cases

Problem

- Write a program that reads an undetermined number of student grades and calculates the average student grade

Solution

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
{
   int gradeCounter;
   int grade;
   double average;
   double total;

   total = 0;
   gradeCounter = 0;
```


Solution

```
   cout << "Enter grade, -1 to end: ";
   cin >> grade;

   while ( grade != -1 )
   {
      total = total + grade;
      gradeCounter = gradeCounter + 1;

      cout << "Enter grade, -1 to end: ";
      cin >> grade;
   }

   if ( gradeCounter != 0 )
   {
      average = total / gradeCounter;

      cout << "Class average is " << average << endl;
   }
   else
      cout << "No grades were entered" << endl;
```


Summary

- In today's lecture we covered
  - **`while`** repetition structure
- Readings
  - P. 81 - 83: **`while`** repetition structure