EECS168/169
Lab 3

Spring 2018

Objectives

- More Loops
  - For loop
  - while loop
  - do-while loop
- debugger

for loop

- Used when number of times loop to be executed is **fixed**.
- Lets sum up ten numbers from 1 to 10: we know ***beforehand*** start(1) and end(10)
  - 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = ?

- This situation can be addressed by *for* **statement**
- Syntax of a for loop

```
for( initialization; Boolean expression looping condition; progression)
{
  //code that repeats
}
```

```
int sum = 0;
for (n = 1; n <= 10; n++)
    sum = sum + n;
```

Increment operator:++ Eg: n++
n = n + 1 ➔ n++

Decrement operator:-- Eg: n--
n = n - 1 ➔ n--

for loop

A *for* Statement

```
//Illustrates a for loop.
#include <iostream>
using namespace std;

int main( )
{
    int sum = 0;

    for (int n = 1; n <= 10; n++) //Note that the variable n is a local
        sum = sum + n;            //variable of the body of the for loop!

    cout << "The sum of the numbers 1 to 10 is "
         << sum << endl;
    return 0;
}
```


Walter Savitch. 2014. *Problem Solving with C++* (9th ed.). Addison-Wesley Professional.

for loop - Pitfalls

Pitfall: **Extra Semicolon(;) in a *for* Statement**

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

Problem semicolon

Walter Savitch. 2014. *Problem Solving with C++* (9th ed.). Addison-Wesley

Pitfall: Infinite loops. **No** condition or condition **always true.**
Lets create one now!

```
for(;;)
{
        cout << "Loop Never ends\n";
}
```

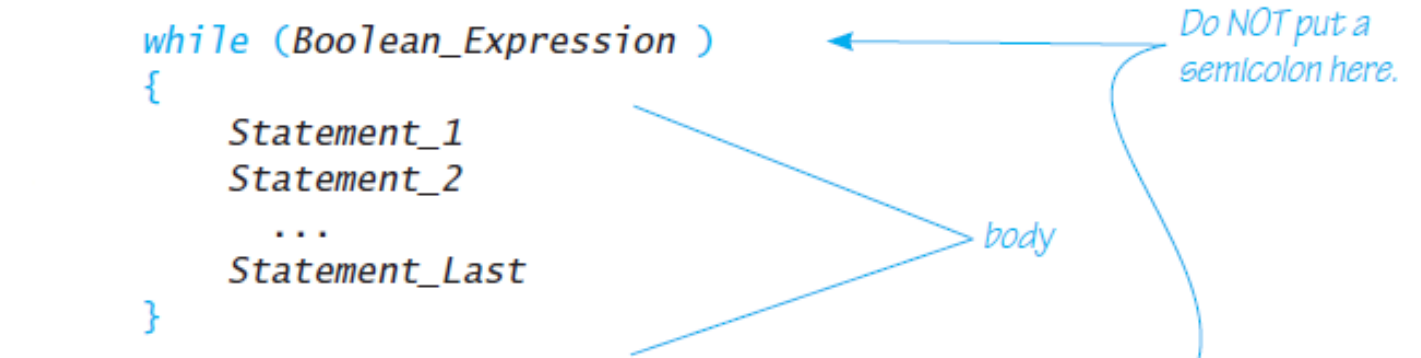
while loop

- Syntax of a while loop

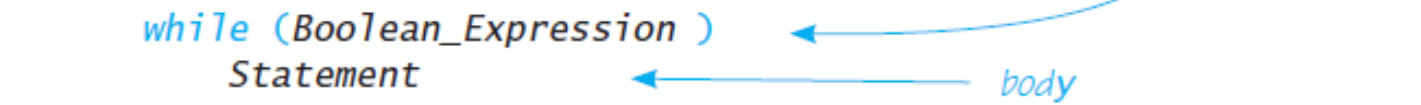
```
//optional initialization
while ( Boolean expression looping condition )
{
    //loop body
    //optional progression
}
```

Syntax of the *while* Statement

A Loop Body with Several Statements:



A Loop Body with a Single Statement:



Walter Savitch. 2014. *Problem Solving with C++* (9th ed.). Addison-Wesley Professional.

do while loop (preview)

A *do-while* statement is similar to a *while* statement
except that the loop body is always executed at least **once**.

Syntax of the *do-while* Statement

A Loop Body with Several Statements:

```
do
{
    Statement_1
    Statement_2
      ...
    Statement_Last
} while (Boolean_Expression);
```

body

Do not forget the final semicolon.

A Loop Body with a Single Statement:

```
do
    Statement
while (Boolean_Expression);
```

body

Walter Savitch. 2014. *Problem Solving with C++* (9th ed.). Addison-Wesley Professional.

Nested loops

- Loop within loop. All loops can be nested. for, while, do-while

```
for (int i = 0; i < 4; i++)
{
    for (int j = 0; j < 3; j++)
    {
        cout << "i = " << i << " and j = " << j << "\n";
    }
}
```

```
i = 0 and j = 0
i = 0 and j = 1
i = 0 and j = 2
i = 1 and j = 0
i = 1 and j = 1
i = 1 and j = 2
i = 2 and j = 0
i = 2 and j = 1
i = 2 and j = 2
i = 3 and j = 0
i = 3 and j = 1
i = 3 and j = 2
```

Exercise 1: ASCII

Present the user with the following menu:

```
1) Select a specific ASCII character provided an int
2) Display visible ASCII values (33 to 126)
3) Exit
```

Requirements:

- If the user chooses a specific value, it must be from 33 to 126 inclusively
- When displaying the conversion, always do so in the following format:
  - <int value> = <char value>
  - Example:
    - 97 = a
- The program does not end until the user wants to exit

Exercise 2: Outbreak!

Flu season is upon us and the number of people getting sick is growing.

- On day 1, there was only 1 person with the flu.
- On day 2, it jumped to 5.
- On day 3, there were 17
- Every day since, the number of people who have the flu is equal to the last 3 days combined

You will make a program that will ask the user for what day they want a count of people with the flu for. Then display the amount.

Sample runs:

```
OUTBREAK!
What day do you want a sick count for?: 1
Total people with flu: 1
```

Exercise 3: Fibonacci Detector

- Create a Fibonacci program in a new folder. This program will allow the user to input a value and verify if it is a number in the Fibonacci sequence. Fibonacci numbers follow a set sequence. The first two numbers in the Fibonacci sequence are F_0=0 and F_1=1.
- After the first two, all following Fibonacci number are calculated by adding up the previous two Fibonacci numbers, $F_n=F_{n-1} + F_{n-2}$
- Here are first 10 numbers in the sequence: 0,1,1,2,3,5,8,13,21,34
- Be careful of the special cases (e.g., the first two/three terms in the sequence)!

```
Enter a value: 5
5 is the 6th value in the Fibonacci Sequence!
Goodbye
```

Exercise 4: Largest Prime

- Create a program that takes an int from the user and tells the user what the largest prime number less than or equal to that value is.
- Repeat this program until they want to quit. Quitting is indicated by a 'y' at the prompt.

```
Input an integer: 8
7 is the largest prime <= 8
Do you want to quit (y/n): n
Input an integer: 4
3 is the largest prime <= 4
Do you want to quit (y/n): n
Input an integer: 199
199 is the largest prime <= 199
Do you want to quit (y/n): y
Goodbye.
```

Exercise 169 Only

- Create a program that takes an int from the user representing what nth prime they want to see. For example, if the user types 1, they want to see the 1st prime number, which for us, will be 2. If they type in 6, they want to see the 6th prime number, which for us, is 13.
- Assume the correct type of input, but not good values (e.g. the -100th prime should get an error message).
- The lowest valid value is 1, and we'll put an upper bound at 100, so you'll need to be able to calculate the first 100 primes.

Please pay attention to the rubric on the wiki page.

Thank you!