# Lecture 5: Control Statements – Part II (coding practice)

Class page: https://github.com/tsung-wei-huang/cs1410-40

Dr. Tsung-Wei Huang
Department of Electrical and Computer Engineering
University of Utah, Salt Lake City, UT

# Control Structure

- ❑ **Normally, statements in a program execute one after the other in the order in which they're written.**
  - ❑ Called sequential execution.
- ❑ **Various C++ statements enable you to specify the next executing statement that is not the next one in sequence.**
  - ❑ Called transfer of control.
- ❑ **All programs could be written in terms of only three control structures (referred as "control statements")**
  - ❑ the sequence structure
  - ❑ the selection structure and
  - ❑ the repetition structure

# Control Structure (cont'd)

- ❑ **C++ provides three types of selection statements**
- ❑ **The `if` selection statement: (<u>single selection</u>)**
  - ❑ The condition is true: perform the following action
  - ❑ The condition is false: skip the action
- ❑ **The `if…else` selection statement: (<u>double selection</u>)**
  - ❑ The condition is true: perform the following action
  - ❑ The condition is false: perform a different action
- ❑ **The `switch` selection statement: (<u>multiple selection</u>)**
  - ❑ Perform one of many different actions, depending on the value of selection expression.

# Control Structure (cont'd)

- ❑ **C++ provides three repetition statements (also called looping statements) for performing statements repeatedly.**
  - ❑ These are the while, do…while and for statements.
- ❑ **The `while` and `for` statements perform the action (or group of actions) in their bodies zero or more times.**
- ❑ **The `do…while` statement performs the action (or group of actions) in its body at least once.**

# Example 1: Simple if-else Statement

- ❑ **Q1: Finds the maximum of four numbers**
  - ❑ Input: four integers, a, b, c, d
  - ❑ Output:  x = max(a, b, c, d)
    - a=3, b=4, c=1, d=18 => x=18
    - a=2, b=5, c=-1, d=-5 => x=5
- ❑ **Q2: Finds the maximum of N *positive* numbers**
  - ❑ Input 1: reads in N
  - ❑ Input 2: reads N numbers, a1, a2, a3, a4, … an
  - ❑ Output: x=max(a1, a2, a3, a4, …, an)
    - N=10, {a1, a2, a3, …, an} = {1, 2, 5, 6, -9, 4, 2, 11, 1, 5}, x=11

# Example 2: Determines Prime Number

- ❑ **Q1: Determines if a number is prime**
  - ❑ Input: an integer, a
  - ❑ Output: ”yes” if a is a prime number, or “no” otherwise
    - a=3 => yes
    - a=10 => no
- ❑ **Q2: Prints all prime numbers <= N**
  - ❑ Input: an integer, N
  - ❑ Output: all prime numbers in [2, N], separated by space
    - N=10 => 2, 3, 5, 7
    - N=3 => 2, 3

# Example 3: Power of a Number

- ❑ **Find a power of a number**
  - ❑ Input: a, b (1 < a, b < 2147483647)
  - ❑ Output: $x = a^b$
    - $a=3, b=4, x=3^4=81$
    - $a=2, b=5, x=2^5=32$
  - ❑ Assume you can only do multiplication one at a time
- ❑ **Naïve method**
  - ❑ $2^{16} = 2*2*2*2*2*2*\ldots*2$ total 15 calculations
- ❑ **Can we do better?**

# Divide and Conquer

- ❑ **Naïve method**
  - ❑ $2^{16} = 2*2*2*2*2*2*\ldots*2$ total 15 calculations
- ❑ **A better way as follows:**

$2^{16} = 2^8 * 2^8$

$2^8 = 2^4 * 2^4$

$2^4 = 2^2 * 2^2$

$2^2 = 2 * 2$ We need only 4 calculations!!!

# How Efficient is it to Compute $a^b$?

- ❑ **Naïve method**
  - ❑ # calculations: linear to b

- ❑ **Divide and Conquer**
  - ❑ # calculations: $\log_2(b)$

- ❑ **Let's say n = 2147483648**
  - ❑ Naïve method takes **2147483647** calculations (~10-30s)
  - ❑ Divide and Conquer takes only **31** calculations (~1us)
    - 10000000x faster!
  - ❑ Indeed, this is a Goo___ interview question