CS102 Unit 2

Programming Languages and Simple Program Execution
Unit Objectives

• Define: **algorithm**, **syntax**, and **semantics**

• Know that statements in a program execute sequentially by default

• Know the basic parts of a C++ program
  – Inclusion of library "headers"
  – Comments
  – Code is partitioned into **functions**
  – **main() function** as the starting point
ALGORITHMS & PROGRAMMING LANGUAGES
Humans and Computers

• Humans understand instructions differently than computers
• Humans easily tolerate ambiguity and abstract concepts using context to help.
  – “Add a pinch of salt.” How much is a pinch?
  – “Steph Curry can shoot the lights out.”
  – “It’s a bear market”
• Computers must be precise, only executing well-defined instructions (no ambiguity) and operating on digital information which is finite and discrete (a fixed number of options)
Algorithms

• Algorithms are at the heart of computer systems, both in HW and SW
  – They are fundamental to Computer Science and Computer Engineering

• Informal definition
  – An algorithm is a precise way to accomplish a task or solve a problem

• A more formal definition:
  – An ordered set of unambiguous, executable steps that defines a terminating process

• Examples: What is the algorithm for
  – Brushing your teeth?
  – Calculating your GPA?
Algorithm Representation

• An algorithm is not a program or programming language

• Just as a story may be represented as a book, movie, or spoken by a story-teller, an algorithm may be represented in many ways
  – Flow chart
  – Pseudocode (English-like syntax using primitives that most programming languages would have)
  – A specific program implementation in a given programming language
Syntax and Semantics

• Programming languages have syntax and semantics
• Syntax: refers to the rules of a language for how it will be expressed and parsed (decomposed)
  – Specific to the language
• Semantics: refers to the meaning of what is written
  – Often transcends the language (same concept in many language)
• Example: A sentence
  – The syntax refers to the proper grammatical rules for writing a sentence: capitalize the first word, have a subject and verb, ending with a period, etc.
  – The semantics refer to the meaning conveyed by the sentence
• C++ Code Example
  – if ( <condition> ) { <action> } is the syntax.
  – The semantics (meaning) is “the action will only be performed if condition is true”
CODE ORGANIZATION AND SEQUENCE OF EXECUTION
Sequence & Executability

- Let's learn a bit more about program execution by using another language named Scratch
  - [http://scratch.mit.edu](http://scratch.mit.edu)
- Write a Scratch program to walk forward, turn right, then walk forward again
- Remember computers need executable steps
  - How far forward?
  - Turn right by how much?
Executability

- Scratch handles the syntax by providing a **menu** of specific "blocks" that define what the language allows you to do
  - Anything you want to do that doesn't have a specific block, requires you to compose use multiple blocks
  - Some blocks have certain aspects you can set to control their behavior.
- Go to the Scratch website, click on Create, and close the tutorial
- Write a Scratch program to walk forward, turn right, then walk forward again
- Remember computers and algorithms need **executable** steps
  - How far forward?
  - Turn right by how much?
Sequence & Executability

• You must compose a program from the "menu" of available blocks

• Create the program shown to the right and then click the green flag to the left of the red stop sign
  – What happens?

• Click the green flag again
  – What happens?
Explicit Content

• Computers do only what you tell them, no more, no less
• What additional details might we want to instruct the computer?
  – Where to start and what direction to face?
  – To provide some delay between steps
    • Remember computers execute code very quickly compared to what a human can see
Big Idea: Sequential Execution

• Notice...
  – Program is executed 1 operation at a time in sequential fashion
  – Each operation is ordered (a definite first, second, third, ... operation)
Repetition 1

- Computers are good at repeating tasks quickly
- If we can find repeated structure, we can use a loop to repeat a set of actions multiple times
- What actions can we repeat and how many times to have our cat friend walk in a square?
Repetition 2

- Computers are good at repeating tasks quickly
- If we can find repeated structure, we can use a **loop** to **repeat** a set of actions multiple times
Only if time allows!

GROUPING CODE IN FUNCTIONS (AKA BLOCKS)
Organizing Code - Functions

• Another way to allow reuse and easy modification is to give a name to sequence of code/actions
  – Wherever we use the name, the associated sequence of code/actions will be executed
• Most programming languages call these functions, methods, procedures, subroutines, etc.
• Scratch calls them "Blocks"
• Create a block named: WalkForwardAndTurn
Organizing Code - Functions

- We can take the actions in our loop and drag them to the definition of `WalkForwardAndTurn`.
- Then click on "My Blocks", find your new block and drag it into the repeat loop.
Sequence of Execution With Functions

• We said we execute sequentially, but with loops and functions is our code still executed sequentially (top-down)?
• No. Loops cause execution to go back and repeat code and functions may cause us to jump to a new set of actions, execute them, and the return back and resume the main program.
Our function "DelayedMove" is useful for the simple task we gave you to implement, but what if I wanted to walk in a rectangle?

We now need to walk different lengths

Q: What might make it more useful and "general" so that we could reuse it in the future more easily?

A: The ability to generalize how many steps to take and how long to wait might be helpful

Let's allow different values of steps and the delay to be input.

Right click on the WalkForwardAndTurn and choose Edit

Click on "Add an Input (number or Text)" once and give the newly appearing box the name: distance and the click on "Add an Input" again and give the new box the name: delay
Parameters/Arguments

- Back in the main window, two new entries "distance" and "delay"
- Drag these in place of the constants (100 or 1) in the move / wait blocks
- Back in the main program, fill in the two text boxes with 100 and 1
- How could you modify the main program only to make the cat walk in a rectangle of 200 wide and 100 long?
C/C++ Program Format/Structure

• Comments
  – Anywhere in the code
  – C-Style => "/*" and "*/"
  – C++ Style => "//"

• Compiler Directives
  – #includes tell compiler what other library functions you plan on using
  – 'using namespace std;' -- Just do it for now!

• main() function
  – Starting point of execution for the program
  – All code/statements in C must be inside a function
  – Statements execute one after the next and end with a semicolon (;)
  – Ends with a 'return 0;' statement

• Other functions
  – printName() is a function that can be "called"/"invoked" from main or any other function
A First Program

• Go to:
  – [http://cpp.sh](http://cpp.sh)

• Enter this program to print "Hello!" **five** times

```cpp
#include <iostream>

int main()
{
    for(int i=0; i < 5; i++) {
        std::cout << "Hello!" << std::endl;
    }
    return 0;
}
```

C++ syntax requires statement to end with a semicolon (;) and grouped by curly braces { }. Removing one would lead to a syntax error.

A semantic error is when I tell the computer to do the wrong thing but it still meets the correct syntax. Change "i=0" to "i=1" and see it print only 4 times rather than the desired 5.

• Introduce some syntax errors

• Introduce a semantic error