CMPT 120: Introduction to Computing Science and Programming 1

Algorithms, Flowcharts and Pseudocodes

Copyright © 2018, Liaqat Ali. Based on CMPT 120 Study Guide and Think Python - How to Think Like a Computer Scientist, mainly. Some content may have been adapted from earlier course offerings by Diana Cukierman, Anne Lavergn, and Angelica Lim. Copyrights © to respective instructors. Icons copyright © to their respective owners.
One-Stop Access To Course Information

- **Course website**: One-stop access to all course information.
  
  [http://www2.cs.sfu.ca/CourseCentral/120/liaqata/WebSite/index.html](http://www2.cs.sfu.ca/CourseCentral/120/liaqata/WebSite/index.html)

  - Course Outline
  - Exam Schedule
  - Python Info
  - **CourSys/Canvas** link
  - Learning Outcomes
  - Office Hours
  - Textbook links
  - Lab/Tutorial Info
  - and more...
  - Grading Scheme
  - Assignments

- **Canvas**: Discussions forum.
  
  [https://canvas.sfu.ca/courses/39187](https://canvas.sfu.ca/courses/39187)

- **CourSys**: For assignments submission, and grades.
  
  [www.coursys.sfu.ca](http://www.coursys.sfu.ca)

Liaqat Ali, Summer 2018.
Some Reminders

• Get familiar with the course Website.
  ▫ http://www2.cs.sfu.ca/CourseCentral/120/liaqata/WebSite/index.html
  ▫ Minor updates may occur during first week.

• Get fob to access LABS (start next week!)
  ▫ If you don’t have it already, get a new fob from Discovery Park 1.
Additional Resources / Online References

• There are several online references that are **as important as the texts.** (Links provided on the course web site.)

• These resources are **very important to your success** in this course. They aren’t meant to be read from beginning to end like the readings in the textbook.

• You should **use them to get an overall picture of the topic** and as references as you do the assignments.
How to Learn in This Course?
Today’s Topics

Algorithms.
During the last lecture, we talked about algorithms. Now, let’s have a look at a couple of more definitions.

An algorithm is a sequence of unambiguous instructions for solving a problem, i.e., for obtaining a required output for any legitimate input in a finite amount of time.

[Source: CMPT 120 Study Guide; Anany Levitin, Introduction to The Design & Analysis of Algorithms, p. 3]
Algorithm: Its Definition and Key Properties - 2

• An algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.

Algorithm: Key Properties

- **Unambiguous**: Each step of an algorithm has to be precisely defined. 
  - After reading an algorithm, there should be no question about what to do.
- **Specific problem**: An algorithm should always present a solution to a particular problem, or group of problems.
- **Legitimate input**: An algorithm might need some kind of input to do its job. This input should be relevant.
- **Finite amount of time**: If started, an algorithm must end eventually. If it never ends, it’s useless.
- **Clear I/O**: Inputs and outputs should be defined clearly.
- **Effective**: Should be effective among many different ways to solve a problem.
Let’s watch this short video about algorithms.

You will hear two new terms related to algorithms in this video. Let’s see if you can note them down.

What’s an algorithm? ________________
Algorithm: The Two New Terminologies

1. ________________________________________________

2. ________________________________________________.

• We will talk about these terms later. Let’s do some examples of algorithms.
Algorithm: Add Two Numbers Entered by a User

Step 1: Start
Algorithm: Verify the Properties

**Step 1: Start**

1. Is it Unambiguous?
2. Solves specific problem?
3. Legitimate input?
4. Finite time?
5. Clear I/O?
6. Is it effective?
Algorithm: A Few Computing Science Terminologies

- In Computing Science, we usually don’t write “suppose”. Rather, we typically say “declare”.
- We call N1, N2, and SUM as “variables”.
  - And, variables typically “store” values.

So, We may choose to re-write the step: Suppose, N1 is the first number.

As: Declare a variable N1.
Or, Declare a variable N1 to store the value of first number.
Or, Declare a variable N1 to store the value of the first number entered by the user.
Re-write the following “add two numbers algorithm” replacing the words declare, variable and store, as necessary.
Algorithm: Find the Smaller of Two Numbers

Write an algorithm to find the smaller of two numbers entered by a user.

Step 1: Start

Step 2: Declare a variable num1 to store the first number.

Step 3: Declare a variable num2 to store the second number.

Step 4: If num1 < num2 then print num1 is smaller.

Step 5: If num2 < num1 then print num2 is smaller.

Step 6: If num1 = num2 then print “Both the numbers are equal.”

Step 7: End
Algorithm: Find the Smallest of Three Numbers

• Write an algorithm to find the smallest of three numbers entered by a user.

• Solution in the next class.
Questions?