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
- and more...
- Grading Scheme
- Lab/Tutorial Info
- Assignments

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

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

- Online references are **as important as the texts**. (Links on course website.)
- These resources are **very important to your success**.
  - 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?

A  Attend Lectures & Labs
R  Read / review Textbook/Slides/Notes
R  Reflect and ask Questions
O  Organize – your learning activities on weekly basis, and finally...
W  Write Code, Write Code, and Write Code.

Liaqat Ali, Summer 2018.
Course Topics

1. General introduction
2. Algorithms, flow charts and pseudocode
3. Procedural programming in Python
4. Data types and control structures
5. Fundamental algorithms
6. Binary encodings
7. Basics of computability and complexity
8. Basics of Recursion
9. Subject to time availability:
   □ Basics of Data File management
Today’s Topics

1. Continue with Algorithms, Flowcharts
2. Pseudocodes
Today’s Topics

Continue with Algorithms, Flowcharts
Algorithm: Find the Smallest of Three Numbers

Step 1: Start
Step 2: Declare variables n1, n2, and n3.
Step 3: Read variables n1, n2, and n3.
Step 4: If n1 < n2 then:
Step 5: If n1 < n3 then print n1 else print n3.
Step 6: else
Step 7: If n2 < n3 then print n2 else print n3.
Step 8: End
Flowchart: Smaller of Three Numbers (Solution)

1. Declare N1, N2, N3
2. Start
3. Read N1, N2, N3
4. If N1 < N2
   - No
   - Print N1
5. If N1 < N3
   - True
   - Print N3
6. If N2 < N3
   - True
   - Print N2
7. False
8. End
Today’s Topics

Pseudocodes

Liaqat Ali, Summer 2018.
You can think of Pseudocodes as the same as Algorithms: a sequence of steps to solve a problem, except:

▫ Steps in algorithm may be less detailed, a pseudocode describes those steps.
▫ Steps in an algorithm look more like English (natural) language instructions, whereas, steps in a pseudocode may look more like code.

For example:

▫ A step in algorithm may be written like this: Convert feet into inches.
▫ An equivalent pseudocode may be written as: Set inches to feet * 12

What's common:

We can transform the instruction written as algorithms, flowcharts or pseudocode into a programming language code.
Pseudocodes - 2

- So, pseudocode a way to describe the steps in an algorithm using some short and simple English (natural) language terms. (*Pseudo* is "almost").
- It describes an algorithm in specific enough detail to be easily implemented in any language.
- Actually, some of the algorithms we wrote in the previous two classes equally qualify as pseudocodes.
Pseudocodes: Features

• We typically use short phrases or keywords to describe steps in a pseudocode.

• For example:

• READ, WRITE, SET, IF, ELSE, ENDIF, WHILE, ENDWHILE, REPEAT, UNTIL

• Pseudocodes omit language specific syntax.

• It enables the programmers to concentrate on writing the coding.
Algorithm/Pseudocode: Smaller of Three Numbers

1: Start
2: Declare variables n1, n2, and n3
3: Read variables n1, n2, and n3
4: If n1 is smaller than n2 and n3, then n1 smaller.
5: If n2 is smaller than n1 and n3, then n2 smaller.
6: If n3 is smaller than n1 and n2, then n3 smaller.
7: End

Liaqat Ali, Summer 2018.
Pseudocode: Find Sum of First 100 Natural Numbers

Step 1: Start

Step 2: Declare N and S.

Step 3: Set initial value of S to 0.

Step 4: Set initial value of N to 1.

Step 5: Add the value of N to S, giving S.

Step 6: Get the next number by adding 1 to N.

Step 7: Repeat steps 5 to 6 until N is equal to 100.

Step 8: Display S.

Step 9: End

Liaqat Ali, Summer 2018.
Flowchart: Find Sum of First 100 Natural Numbers

\[
\begin{align*}
S &= S + N \\
N &\leq 100 \\
\text{Yes} &\rightarrow \text{Print } S \\
\text{No} &\rightarrow N = N + 1
\end{align*}
\]
Algorithm: Convert Height In Meters To Feet and Inches

1: Start
2:
9: End

Liaqat Ali, Summer 2018.
Flowchart: Convert Height In Meters To Feet and Inches

Liaqat Ali, Summer 2018.
Why Pseudocodes?

Writing code to solve a problem would have two parts:

1. Identifying **what** to : Writing **Pseudocode** (Algorithm)
2. Knowing **how** to do : Writing **Python** code

• So, write an algorithm, express it in pseudocode before you start coding.

• Especially as you’re starting to program, you don’t want to be worrying about what you’re trying to say and how to say it at the same time.
Questions?