CMPT 120: Introduction to Computing Science and Programming 1

Procedural programming in Python

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

  - 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

- **CourSys**: Assignments submission, grades - 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.
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.
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. Procedural Programming in Python
   - Transform an algorithm to a program:
   - Write Code, Write Code, and Write Code.
Today’s Topics

Transform an algorithm to a program:
Write Code, Write Code, and Write Code.
Write a Python Program to Add Two Numbers

Step 1: Start

Step 2: Declare a variable n1 and initialize it to 0.  
\( n1 = 0 \)

Step 3: Declare a variable n2 and initialize it to 0.  
\( n2 = 0 \)

Step 4: Declare a variable sum and initialize it to 0.  
\( sum = 0 \)

Step 5: Get 1st number from user and store in n1.  
\( n1 = \text{input()} \)

Step 6: Get 2nd number from user and store in n2.  
\( n2 = \text{input()} \)

Step 7: Add n1 and n2, and store the answer in sum.  
\( sum = n1 + n2 \)

Step 8: Display SUM  
\( \text{print}(sum) \)

Step 9: End

Liaqat Ali, Summer 2018.
Write a Python Program: Add Two Numbers

Step 1: Start
Step 2: Declare a variable n1 and initialize it to 0. n1 = 0
Step 3: Declare a variable n2 and initialize it to 0. n2 = 0
Step 4: Declare a variable sum and initialize it to 0. sum = 0
Step 5: Get 1st number from user and store in n1. n1 = input()
Step 6: Get 2nd number from user and store in n2. n2 = input()
Step 7: Add n1 and n2, and store the answer in sum. sum = int(n1) + int(n2)
Step 8: Display SUM
   print(sum)
Step 9: End

Liaqat Ali, Summer 2018.
Write a Python Program to Add Three Numbers

Step 1: Start

Step 2: Declare a variable n1 and initialize it to 0.  
\[ n1 = 0 \]

Step 3: Declare a variable n2 and initialize it to 0.  
\[ n2 = 0 \]

Step 4: Declare a variable n3 and initialize it to 0.  
\[ n3 = 0 \]

Step 5: Declare a variable sum and initialize it to 0.  
\[ \text{sum} = 0 \]

Step 6: Get 1st number from user and store in n1.  
\[ n1 = \text{input()} \]

Step 7: Get 2nd number from user and store in n2.  
\[ n2 = \text{input()} \]

Step 8: Get 3rd number from user and store in n3.  
\[ n3 = \text{input()} \]

Step 9: Add N1 and N2 and assign the result to SUM.  
\[ \text{SUM} \leftarrow n1 + n2 \]
\[ \text{sum} = \text{int}(n1) + \text{int}(n2) + \text{int}(n3) \]

Step 10: Display SUM  
\[ \text{print(sum)} \]

Step 11: End

Liaqat Ali, Summer 2018.
Write a Python Program to Solve 2x+2y

**Step 1:** Start

**Step 2:** Declare a variable \( x \) and initialize it to 0.

**Step 3:** Declare a variable \( y \) and initialize it to 0.

**Step 4:** Declare a variable \( \text{ans} \) and initialize it to 0.

**Step 5:** Get value of \( x \) from user and store in \( x \).

**Step 6:** Get value of \( y \) from user and store in \( y \).

**Step 7:** Solve the expression: \( 2\times x + 2 \times y \).

**Step 8:** Display SUM

**Step 9:** End

\[
x = 0 \\
y = 0 \\
\text{ans} = 0 \\
x = \text{input()} \\
y = \text{input()} \\
\text{ans} = 2*\text{int}(x) + 2*\text{int}(y) \\
\text{print(\text{ans})}
\]
Write a Program for Following Marks Flowchart

START
Declare M, F, T
Read M, F
T=M+F
Display T

A

T < 50

No
Print “Pass”

Yes
Print “Fail”

END

M = 0
F = 0
T = 0

M = input()
F = input()
T = int(M) + int(F)
print(T)
if (T<50):
    print(“fail”)
else:
    print(“pass”)

Liaqat Ali, Summer 2018.
Program: Find the Smaller of Two Numbers

Step 1: Start

Step 2: Declare variable n1 to store the 1st number.
n1 = 0

Step 3: Declare variable n2 to store the 2nd number.
n2 = 0

Step 4: Get the value of n1 from the user.
n1 = input()

Step 5: Get the value of n2 from the user.
n2 = input()

Step 6: If n1 < n2 then print n1.
if (n1 < n2):
    print(n1)
else:
    print(n2)

Step 8: End
Program: Find the Smallest of Three Numbers

Read n1, n2, n3.

If n1 < n2 and n1 < n3,
Write n1.

If n2 < n1 and n2 < n3,
Write n2.

If n3 < n1 and n3 < n2,
Write n3.

n1 = 0
n2 = 0
n3 = 0

n1 = input()
n2 = input()
n3 = input()

if (n1 < n2) and (n1 < n3): print(n1)
if (n2 < n1) and (n2 < n3): print(n2)
if (n3 < n1) and (n3 < n2): print(n3)
Program: Find Sum of First 100 Natural Numbers

Set sum to 0
Set n to 1
Repeat until n <= 100:
    Set sum = sum + n
    n = n + 1
Write S
Algorithm: Convert Height In Meters To Feet and Inches

1: Start
2: Declare meter, feet, total inches and inches variables.
3: Initialize feet, total inches and inches variables to 0.
4: Get the height in meters from the user.
5: Convert meters into total inches and store it.
6: Convert total inches into feet and store it.
7: Find remainder of total inches / 12 and store in inches.
8: Display the value in feet variable.
9: Display the value in the inches variable.

Read meters
Set totInch to 39.37 \times\text{metres}
Set feet to whole number part of \(\text{totInch} / 12\)
Set inches to remainder of \(\text{totInch} / 12\)
Write feet, inches

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

Read meters
Set total_inch to 39.37 × metres
Set feet to whole number part of total_inch / 12
Set inches to remainder of total_inch / 12
Write feet, inches

Submit on Canvas today by midnight

Liaqat Ali, Summer 2018.
Questions?