CMPT 120

Topic: Introduction to Python and Demo of Software Development Process
Learning outcomes

At the end of this course, a student is expected to:

• Describe and apply fundamental concepts and terminology:
  • Execution flow

• Create (design) small size programs using Python:
  • Hand trace code (programs) and predict results of executing code
Last Lecture

- Defined programming languages and their evolution (first 3 generations)
- Described compiler and interpreter
- Introduced to Scratch
- Demo’ed of the Problem Solving Process using Scratch

Problem/task: Cat walking in a square?

Algorithm:
- Story: cat walks in a square
- In more details: repeat 4 times cat walk 50 steps right then turn clockwise 90°

Step 1: State problem
Step 2: Design algorithm + Identify data
Step 4: Implement algorithm into computer program
Step 5: Test it
Today’s Menu

• i-clicker Session 1
• Introduce Python
• Practice the Problem Solving Process some more using Python
Let’s introduce Python

• We shall use Python 3 (as opposed to Python 2)
• Where to find Python?

• What is Python?
  • “Python is an text-based, interpreted, object-oriented, dynamically-typed, high-level programming language…”
    Inspired from https://www.python.org/doc/essays/blurb/

• Why Python?
  • Can be used on many platforms:
    • Windows, MAC OS, Linux
  • Powerful, yet easy to learn, use, read and debug 😊
  • Syntax emphasizes readability
How to use Python IDLE

• 2 ways of using Python IDLE:

1. Python Interpreter Shell
   • Workflow:
     • Write one statement of Python @ a time
       • Wait a minute! What is a “statement”? 😊
     • Press Enter
     • The Python Interpreter makes sure the statement does not contain any error
     • Then it executes (interprets) the statement typed
     • And outputs the result
How to use Python IDLE

• 2 ways of using Python IDLE:

2. Python Program Editor
   • Workflow:
     • Create a new program OR open an already existing program
     • Write code into the program
     • Save the program
     • Execute the whole program:
       • The Python Interpreter makes sure the program does not contain any error
       • Then it executes (interprets) the statements in the program
       • And outputs the result
Demo –
Step 1 - Problem Statement

• Let’s figure out our course final grade

• More specifically, the activities of our course are
  • Assignment 1
  • Assignment 2
  • Assignment 3
  • Assignment 4
  • Midterm
  • Final exam
Demo – Step 2 - Solution (Algorithm)

- Set finalGrade to 0
- Input gradeAssn1
- Input gradedOutOfAssn1
- Input percentOfFinalGradeAssn1
- Compute newGradeAssn1 -> gradeAssn1 * percentOfFinalGradeAssn1 / gradedOutOfAssn1
- finalGrade = finalGrade + newGradeAssn1
- Input gradeAssn2
- Input gradedOutOfAssn2
- Input percentOfFinalGradeAssn2
- Compute newGradeAssn2 -> gradeAssn2 * percentOfFinalGradeAssn2 / gradedOutOfAssn2
- finalGrade = finalGrade + newGradeAssn2
- Input gradeAssn3
- Input gradedOutOfAssn3
- Input percentOfFinalGradeAssn3
- Compute newGradeAssn3 -> gradeAssn3 * percentOfFinalGradeAssn3 / gradedOutOfAssn3
- finalGrade = finalGrade + newGradeAssn3
- Input gradeAssn4
- Input gradedOutOfAssn4
- Input percentOfFinalGradeAssn4
- Compute newGradeAssn4 -> gradeAssn4 * percentOfFinalGradeAssn4 / gradedOutOfAssn4
- finalGrade = finalGrade + newGradeAssn4
- Input gradeMT
- Input gradedOutOfMT
- Input percentOfFinalGradeMT
- Compute newGradeMT -> gradeMT * percentOfFinalGradeMT / gradedOutOfMT
- finalGrade = finalGrade + newGradeMT
- Input gradeFE
- Input gradedOutOfFE
- Input percentOfFinalGradeFE
- Compute newGradeFE -> gradeFE * percentOfFinalGradeFE / gradedOutOfFE
- finalGrade = finalGrade + newGradeFE
- Print finalGrade
Demo – Step 4 - Implement Algorithm into Program

• Let’s write some code!

• Then, let’s illustrate:
  • Execution flow
  • Hand tracing our code
Demo – Step 5 - Testing

• How to test our program?
  • Does our program execute?

• Does it solve our initial problem?
  • Test Case
    1. Test Data:

2. Expected Results:
Summary

• i-clicker Session 1
• Introduce Python
• Practice the Problem Solving Process some more using Python

Step 1
State problem

Step 2
Design algorithm + Identify data

Step 4
Implement algorithm into computer program

Step 5
Test it
Next Lecture

- Start learning Python’s building blocks
  - Variables
  - Values
  - Types
  - and Literal values
- Continue illustrating
  - Execution Flow
  - Hand tracing