CMPT 120

Topic: Python’s building blocks
    -> Variables, Values, and Types
Last Lecture

- i-clicker Session 1
- Introduced Python
- Practiced the Problem Solving Process some more using Python
Learning outcomes

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

• Describe and **apply** (i.e., use in s/w dev.) fundamental concepts and terminology of Python:
  • Literals
  • Variables
  • Data types
  • Execution flow

• Create (design) small size programs using Python:
  • **Hand trace** code (programs) and predict results of executing code (determining contents of variables and output)
Today’s Menu

• Start learning Python’s building blocks
  • Variables
  • Values
  • Types
  • and Literal values
• Hand tracing
• Execution Flow
• Python Comments and Comment Header Block
In the real world, there are …

Real World

- “Things” e.g.: 
  and

- Actions e.g.:
In the computer world, there are …

<table>
<thead>
<tr>
<th>Real World</th>
<th>Computer World</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Things”</td>
<td>“Things”</td>
</tr>
<tr>
<td>Actions</td>
<td>Actions</td>
</tr>
<tr>
<td></td>
<td>-&gt; Scratch:</td>
</tr>
<tr>
<td></td>
<td>-&gt; Python:</td>
</tr>
</tbody>
</table>
Input-Process-Output Model

- Most computer programs follow this model

Input → Process → Output

Data → Data + Actions → Data
Variable

- We saw in Lecture 3 that computers are very good at manipulating lots of data

- But in order for the computer to manipulate this data, we must tell the computer to remember the data

- We do this by using variables in our programs
Variable

• **Definition**: A variable is a
  • memory location
  • to which **we** give a **name**
  • to which **we** assign a **value** of a particular **data type**

• In a computer program, variables are referred to by their name
Variable name

• Syntax rule:
  1. Can contain:
     •
     •
     •
     Cannot contain:
     •
     •
  2. Must start with
Variable name

- Convention 1: use camelCase (see wikipedia)
  - Example: finalLetterGrade
- Convention 2: use underscore between words
  - Example: final_letter_grade

- Python is case sensitive:
  - variable PRICE
  - variable Price
  - variable price

  3 different variables
Value and their data type

• A value can either be
  • an integral (whole) number
    • Example:
      • in Python, this data type is called: int (integer)
  • a floating point number
    • Example:
      • in Python, this data type is called: float
  • a sequence of character(s)
    • Example:
      • in Python, this data type is called: str (string)
Variable – Demo

1. How to create a variable in Python
   • In doing so, we shall introduce the **Assignment operator “=“** (assignment statement)

2. How to use a variable
   • In doing so, we shall introduce the **output built-in function ”print”** (output statement)

3. How to assign another value to an already existing variable
   • In doing so, we shall use the **Assignment operator “=“** (assignment statement)

Using Python IDLE Interpreter Shell & Program Editor
Weakly Typed

- Python is a **weakly typed** high level programming language

- This means that when we need to use a variable in our Python program, all we need to do is assign a value to our variable in order to create the variable (or the space in memory)
Strongly Typed

• This is not the case in **strongly typed** high level programming languages like C, C++ and Java, where a variable must first be declared, i.e., one must first state the type of value this variable is to reference, before the variable is used.

• Example: in C

```c
float radius; <- variable declaration
float areaCircle; <- variable declaration
float pi; <- variable declaration
pi = 3.14; <- variable initialized
radius = 2.5; <- variable initialized
// area of circle = pi * radius^2 <- comment
areaCircle = pi * pow(radius,2) <- variables used
```
Literal Value

• Examples: The values in red are literal values:
  apple = 0.75
  juice = 1.65
  sandwich = 4.80

• They are called “literal values” because they are literally typed into a program
GPS (Good Programming Style) – 1

• Variable name must be descriptive, i.e., it must describe the purpose of the variable or the meaning of its value

  • For example:

  • Why?
    • Creates self-documented code
It follows that …

- Reusing variable for different purposes is never a good idea!
  - For example:
  - Why?
GPS (Good Programming Style) – 2

- We cannot use a Python keyword as a variable name
  - Python keywords: Section 2.2 (page 10) of our online textbook

- For example:

- Why?
GPS (Good Programming Style) – 3

- Have a look at our Good Programming Style web page and see what it says about **Literal Values**
Hand Tracing

• What is it?
  • When a software developer manually goes through her/his code (program) and “execute” it as if s/he was a computer, mimicking the Python Interpreter

• Why doing it?
  • To figure out what our program does/produces, hence to verify whether our program is solving the problem
  • To determine whether our program contains any errors
Hand Tracing

- **Hand tracing** often involves tracing the content of the memory (used by the executing program) as it would appear if the program was to be executed
  - Tracing variables (drawing these variables as boxes) and their content (values)
- Example:
Execution Flow

• What is it?
  • It is the path the Python Interpreter takes when it executes our program
  • When we select the “Run/Run Module (F5)” menu option of the IDLE Python Program Editor window, the Python Interpreter starts interpreting each statement (line) of our program from its very top down to its last line, proceeding in a sequential fashion (one line at a time, top to bottom)

• We shall learn Python building blocks that will allow us to disrupt this sequential flow of execution later on this semester
Comments in Python

Syntax:

```python
# some comments
```

or

```
Python statement  # some comments
```

The Python Interpreter ignores anything written to the right of the `#` character, all the way to the end of a line.

How to create comments in our Python program

• We can use the steps of our algorithm as comments in our Python program
  • Example: Let’s have a look at `FinalCourseGrade.py`

Reasons for using comments in our Python program

1. Explaining what the statements of our program do
2. Temporarily ”removing” code from our program without deleting it, as we are developing and debugging our program
Header Comment Block

• **Purpose:** Give information about our program
• **Composed of:**
  • Filename
  • Description of program
  • Author
  • Date of creation
• **Location:** At the very top of our program
• **Execution?**
  • Since we start each line of our header comment block with a `#` sign, i.e., making each line a comment, this signifies that the Python Interpreter skips the entire header comment block and start executing the first non-comment line below it
Header Comment Block

• Example: Let’s have a look at `FinalCourseGrade.py` on our course web site

```python
# FinalCourseGrade.py
#
# Compute course final grade from 3 activities:
# - Assignment 1 -> Assn1
# - Midterm examination -> MT
# - Final examination -> FE
#
# Anne Lavergne
#
# May 2017

... rest of program
```
Summary

- Started learning Python’s building blocks
  - Variables
  - Values
  - Types
  - and Literal values
- Hand tracing
- Execution Flow
- Python Comments and Comment Header Block
Next Lecture

- Continue learning Python’s building blocks
  - Let’s have a look at the 2\textsuperscript{nd} concept “Actions”
    - Statements
    - Expressions

- Categories of Statements
  1. Assignment statement
  2. Input statement
  3. Output statement
  4. Operational statements