CS A131: Python Programming I

Nadia Ahmed

Orange Coast College, Computer Science

CS A131

Overview

- Introduction to Programming in Python
    - History of Python
    - Introduction to Python
- Our first Python Program
    - Example `hello_world.py`
    - Structure of a Python program
    - `print`
    - Program execution
    - String constants

Introduction to Programming

Categories of programming languages

Machine languages	(stream of 1's and 0's)
Assembly languages	(low-level CPU instructions)
High level languages	high-level instructions

Translation of high-level languages

Interpreter	(translation for each instruction)
Compiler	(translation once for all code)
Hybrid	(combination of the above)

Types of programming languages

Functional	(Lisp)
Structured	(Pascal, C, Ada)
Object-oriented	(C++, Java, Python)

Introduction Python

- What is Python?
  - Programming language
    - high-level
    - object-oriented
    - interpreted
  - comprehensive standard library
- Why Python?
  - interactive
  - modular
  - dynamic
  - portable
  - readibility
  - multiple people can work on a single project
  - facilitates code reusability
- Interactive mode vs Script mode

History of Python

- created by Guido van Rossum and released in 1991
- developed as a successor to the ABC language with exception handling and Amoeba interfacing capabilities
- Python 2.0 (2000), Python 3.0 (2008)
- emphasizes code readibility
  - whitespace indentation to delimit code blocks
  - syntax that allows programmers to express concepts in fewer lines of code
  - dynamic (automatic memory management)
  - supports object oriented, imperative, functional, and procedural programming styles
- open source
- useful for prototyping

Our First Python Experience: Interactive Mode

```
~/workspace/csA131/exercises/ $ python
Python 3.6.0 (default, Aug 29 2017, 00:18:07)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello World!')
Hello World
```

- To enable Python in **interactive** mode type `python` in the command line
- You can use the `print` function to display output in a Python program
- `print` function is built-in and it's argument is what you would like to display to the screen
- `'Hello World!'` is a string or a sequence of characters known as a **string literal**
- String literals are enclosed in quote marks which can be ' or "

Our First Python Program v1: Script mode

```
#####################################
#hello_world.py: First Python Program
#
# author: Nadia Ahmed
#
# modifications:
# 08/30/17 NA initial version
#####################################

print('Hello World!')

#EOF
```

Our First Python Program v2

```
#########################################
#hello_world.py: First Python Program
#
# author: Nadia Ahmed
#
# modifications:
# 08/30/17  NA initial version
#########################################
def main():
  print('Hello World!')
#EOF

#call main
main()
```

Our First Python Program

- Program comments
  - start with #
  - are ignored by the interpreter
  - should be used to
    - document the program code
    - structure the program code
    - enhance readability

```
######################################
# hello_world.py:  First  Python
#                     Program
#
# author:  Nadia  Ahmed
#
# modifications:
# 08/30/17  NA  initial  version
######################################
...
```

Our First Python Program v2: Script mode

- `def main():`
  - main function of the Python program
  - `def` defines the function and its instructions
  - you must call main `main()` separately in order to run the instructions: the program execution starts and ends at that point
- function body
  - block of code (definitions and statements)
  - starts with an indent
- `print` function

```
def main():
  print('Hello World!')
#EOF

#call main
main()
```

Our First Python Program

Program execution

- `python hello_world.py`
- interpreter reads file `hello_world.py` and executes the instructions

String Literals

- Character string constants: “Strings”
  - start and end with quote characters (“”, ‘’, “‘ ”’)
  - may not extend over a single line
  - subsequent string constants are combined
  - text formatting using escape sequences
    - `\n` new line
    - `\t` horizontal tab
    - `\r` carriage return
    - `\b` backspace
    - `\a` alert/bell
- Experiments with the `HelloWorld` program...