**East West University**
**Department of Computer Science and Engineering**

**CSE 303: Statistics for Data Science**
**LAB 01 (Handout)**
**Course Instructor: Dr. Mohammad Rezwanul Huq**

---

# Introduction to Python Programming

## Lab Objective

Familiarize students with the fundamental concepts of Python Programming such as data types, control flow statements, functions, lambda functions and list comprehension.

## Lab Outcome

After completing this lab successfully, students will be able to:

1. **Understand** the fundamental concepts of Python.
2. **Write** Python programs to solve generic problems with modest complexity.

## Psychomotor Learning Levels

This lab involves activities that encompass the following learning levels in psychomotor domain.

| Level | Category | Meaning | Keywords |
|---|---|---|---|
| P1 | Imitation | Copy action of another; observe and replicate. | Relate, Repeat, Choose, Copy, Follow, Show, Identify, Isolate. |
| P2 | Manipulation | Reproduce activity from instruction or memory | Copy, response, trace, Show, Start, Perform, Execute, Recreate. |

## Required Applications/Tools

- Anaconda Navigator (Anaconda3)
  - Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment.
  - Popular Tools/IDEs: Spyder, Jupyter Notebook

- Google Colab: Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

## Lab Activities

1. **Introducing Python**
   - General purpose programming language, you can build anything!
   - Open Source. Free to use.
   - Lots of Python packages
   - Current Version: Python 3.x
   - **Indentation is important using tabs or spaces!**

2. **Printing Statements**

```
print ("This line will be printed.")
```

using Format specifier:

```
name = "John"
age = 23
print ("%s is %d years old." % (name, age))
```

3. **Objects (Variables) and Types**

- Python is purely object oriented. Not "statically typed".
- Integer, Floating points, Strings
- Strings are defined either with a single quote of a double quotes.
- Useful functions: id(object_name), type(object_name), isinstance(object_name, type_name)
- **Immutable Objects:** These are of in-built types like int, float, bool, string, unicode, tuple. In simple words, an immutable object can't be changed after it is created.

4. **Arithmetic Operators**
   - Same set: +, -, *, /, %
   - // Floor division - division that results into whole number adjusted to the left in the number line x // y
   - ** Exponent - left operand raised to the power of right x**y (x to the power y)

5. **Comparison Operators**
   - Same set: >, <, >=, <=, ==, !=

6. **Logical Operators**
   - **and, or, not**

7. **Bitwise Operators**
   - **& (Bitwise AND)**
   - **| (Bitwise OR)**
   - **~ (Bitwise NOT)**
   - **^ (Bitwise XOR)**
   - **>> (Bitwise right shift)**
   - **<< (Bitwise left shift)**

8. **Conditional Statements**

```
if condition1:
        action1
elif condition2:
        action2
else:
        action3
```

**Special Operators: is, is not, in, not in**

```
name = "John"
if name in ["John", "Rick"]:
        print("Your name is either John or Rick.")
```

## 9. Loops

- There are two types of loops in Python, for and while.

```
primes = [2, 3, 5, 7] # A list
for prime in primes:
   print(prime)

for x in range(5):
   print(x)

count = 0
while count < 5:
   print(count)
   count += 1  # This is the same as count = count + 1

count=0
while(count<5):
   print(count)
   count +=1
else:
   print("count value reached %d" %(count))
```

## 10. Mutuable Objects: Lists

- Lists are very similar to arrays. They can contain any type of variable, and they can contain as many variables as you wish.
- Elements can be accessed using indexing: mylist = [1, 2, 3]; print (mylist[0]); print(mylist[-1]); print(mylisy[1:3]);
- Python List append(): Add a single element to the end of the list
- Python List clear(): Removes all Items from the List
- Python List copy(): returns a shallow copy of the list
- Python List count(): returns count of the element in the list
- Python List extend(): adds iterable elements to the end of the list
- Python List index(): returns the index of the element in the list
- Python List insert(): insert an element to the list
- Python List pop(): Removes element at the given index
- Python List remove(): Removes item from the list
- Python List reverse(): reverses the list
- Python List sort(): sorts elements of a list

## 11. Immutuable Objects: Tuple

- A tuple in Python is similar to a list. The difference between the two is that we cannot change the elements of a tuple once it is assigned whereas we can change the elements of a list.
- A tuple can have any number of items and they may be of different types (integer, float, list, string, etc.).

```
# Tuple having integers
my_tuple = (1, 2, 3)
print(my_tuple)

# tuple with mixed datatypes
my_tuple = (1, "Hello", 3.4)
print(my_tuple)
```

- Python Tuple count(): returns count of the element in the list
- Python Tuple index(): returns the index of the element in the list

**12. Mutuable Objects: Dictionaries**

- A dictionary is a data type similar to arrays, but works with keys and values instead of indexes. Each value stored in a dictionary can be accessed using a key, which is any type of object (a string, a number, a list, etc.) instead of using its index to address it.

```
phonebook = {
   "John" : 938477566,
   "Jack" : 938377264,
   "Jill" : 947662781
}
print(phonebook)
```

Looping over Dictionaries:

```
phonebook = {"John" : 938477566,"Jack" : 938377264,"Jill" : 947662781}
for name, number in phonebook.items():
   print("Phone number of %s is %d" % (name, number))
```

- Python Dictionary clear(): Removes all Items
- Python Dictionary copy(): Returns Shallow Copy of a Dictionary
- Python Dictionary fromkeys(): creates dictionary from given sequence
- Python Dictionary get(): Returns Value of The Key
- Python Dictionary items(): returns view of dictionary's (key, value) pair
- Python Dictionary keys(): Returns View Object of All Keys
- Python Dictionary pop(): removes and returns element having given key
- Python Dictionary popitem(): Returns & Removes Latest Element From Dictionary
- Python Dictionary setdefault(): Inserts Key With a Value if Key is not Present
- Python Dictionary update(): Updates the Dictionary
- Python Dictionary values(): returns view of all values in dictionary

Useful Links:

- https://www.learnpython.org/
- https://www.programiz.com/python-programming/operators
- https://www.programiz.com/python-programming/methods/list
- https://www.programiz.com/python-programming/methods/dictionary
- https://www.programiz.com/python-programming/tuple