CMPE 346- NATURAL LANGUAGE PROCESSING

Introduction to Python Programming Language

Python is one of the commonly used, general purpose high-level programming language.

Python is an interpreted scripting language, which helps development cycle a lot more fast and easy to modify and move on with changes.

You can experiment some basic things without the need of creating a new program or file every time.

You can open Terminal or Command Prompt application in your computer and type python/ python3 in there, and you can start try and work with Python on this platform.

You can also create files with ".py" extension and you could run directly by typing the command as following:

```
python hello.py
```

Examples:

```
print("Hello, World!")
print(23 + 1)
print(10)
type(10)
type(4.2)
print("I am a string.")
type("I am a string.")
type(True)
```

Basic Data Structures:

Lists: A list is a collection which is ordered and changeable. In Python lists are written with square brackets.

Example:

```
thislist = ["apple", “banana”, "cherry"]
print(thislist)
print(thislist[1])
thislist[1] = “blackcurrant"
print(thislist[1])
print(len(thislist))
thislist.append(“orange”)
```

Dictionary: A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

Example:

```
thisdict = {
     "brand": "Ford",
     "model": "Mustang",
     "year": 1964
}
print(thisdict)
x = thisdict[“model”]
print(x) thisdict["year"] = 2018
```

Tuples:

```
coral = ('blue coral', 'staghorn coral', 'pillar coral', 'elkhorn coral')
print(coral)
print(coral[2])
print(coral[-4])
print(coral[::-1])
print(coral[1:3])
print(coral[:3])
numbers = (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
print(numbers[1:11:2])
print(len(numbers))
print(max(numbers))
```

Control Structures:

Conditional Statements: One of the essential control structures that helps us create programs to execute in a specific conditions.

Examples:

```
>>> x = 0
>>> y = 5
>>> if x < y:
...     print('The first number is smaller than the second one.')
>>> if y < x:
...     print('The second number is smaller than the first one.')
>>> if x < y:
...     print('The first number is smaller than the second one.')
>>> if x or y:
...     print('this and that')
>>> if x and y:
...     print('this and that')
```

Loop Types:

for Loops:

```
>>> for x in range(5):
...     print(x)

>>> for y in range(3,6):
...     print (y)
primes = [2,3,5,7]
for prime in primes:
...     print(prime)
{x: x**2 for x in (2, 4, 6)}
 knights = {'gallahad': 'the pure', 'robin': 'the brave'}
for k, v in knights.items():
...     print(k, v)
>>> for i, v in enumerate(['tic', 'tac', 'toe']):
...     print(i, v)
>>> for j in reversed(range(1, 10, 2)):
...     print(j)
```

while Loops:

```
>>> count = 0
>>> while count<5:
...     print (count)
...     count += 1
```

break and continue statements:

```
>>> count = 0
>>> while True:
...     print(count)
...     count += 1
...     if count >= 5:
...             break

>>> for x in range(10):
...     #Check if x is even
...     if x%2 == 0:
...             continue
...     print(x)
```

Loops with Conditional Statements:

```
# Prints out 0,1,2,3,4 and then it prints "count value reached 5”
>>> count=0
>>> while(count<5):
...     print(count)
...     count +=1
... else: print("count value reached %d" %(count))

# Prints out 1,2,3,4
>>> for i in range(1,10):
...     if(i%5==0):
...             break
...     print(i)
... else:
...     print("this is not printed because for loop is terminated
because of break but not due to fail in condition.”)
```

Functions:

```
>>> def add_two_numbers(num1,num2):
...     sum = num1 + num2
...     return sum
>>> print(add_two_numbers(3,4))
```