PYTHON FUNCTIONS I

OUTLINE

INTRODUCTION

Functions are a convenient way to divide your code into useful blocks, allowing us to order our code, make it more readable, reuse it and save some time. Also functions are a key way to define interfaces so programmers can share their code.

STRING
FUNCTION

In python unlike other programming languages, string functions are system constructors which helps one perform so many manipulations on string data types. Though these functions exist in other programming languages, their applications differ from them.

STRING
FUNCTION

- swapcase
- upper
- lower
- isupper
- islower
- title
- find
- count
- rstrip
- lstrip
- strip

E X A M P L E

1. str = “i am a programmer”

   print(str.upper());

   print(str.title());

   print(str.count(“a”))   e.t.c

NUMERIC
FUNCTION

In python programming, we can perform different math operations on integers as seen in other object oriented programming languages.

NUMERIC
FUNCTIONS

- ceil
- floor
- round
- max
- min
- pi

E X A M P L E

1. val = 5/2

   print(round(val));

   NB: Some functions may not work until you import the math class.

   For example: import math

       print(math.ceil(val));

LIST
FUNCTION

The list data type has some more methods. Here are all of the methods of list objects

- list.append(*x*) Add an item to the end of the list.
- list.extend(*L*) Extend the list by appending all the items in the given list.
- list.insert(*i*, *x*) Insert an item at a given position. The first argument is the index of the element before which to insert, so a.insert(o, x) inserts at the front of the list.
- list.remove(*x*) Remove the first item from the list whose value is *x*. It is an error if there is no such item.
- list.pop([*i*]) Remove the item at the given position in the list, and return it. If no index is specified, a.pop() removes and returns the last item in the list.

LIST
FUNCTION

- list.clear() Remove all items from the list. Equivalent to del a[:].
- list.index(*x*[, *start*[, *end*]]) Return zero-based index in the list of the first item whose value is *x*. Raises a ValueError if there is no such item.
- list.count(*x*) Return the number of times *x* appears in the list.
- list.sort() Sort the items of the list in place
- list.reverse() Reverse the elements of the list in place.
- list.copy() Return a shallow copy of the list. Equivalent to a[:].

DEL STATEMENT

There is a way to remove an item from a list given its index instead of its value: the del statement. This differs from the pop() method which returns a value. The del statement can also be used to remove slices from a list or clear the entire list.

For example:

```
a = [-1, 1, 66.25, 333, 333, 1234.5]
del a[0]
print(a)
```

TUPLE

A tuple consists of a number of values separated by commas. They are immutable unlike the list. We can also perform operations on tuples such as checking length, max and min.

Creating a Tuple:

```
tp1 = () # creates an empty tuple with no data

tp2 = (15,12,45,67)

tp3 = tuple([10,21,13,47,43]) # tuple from array

tp4 = tuple("abc") # tuple from string
```

TUPLE
FUNCTION

```
tp = (11, 12, 75, 29, 90,123)
 print(max(tp))
 print(min(tp))
 print(sum(tp))
 print(len(tp))
```

Iterating through **TUPLE**

```
tp = (11, 12, 75, 29, 90,123)
 for i in tp:
     print(i) or print(i, end=" ")
```

SETS

Python also includes a data type for *sets*. A set is an unordered collection with no duplicate elements.

Set objects also support mathematical operations like union, intersection, difference, and symmetric difference.(-,|,& and ^)

Curly braces or the set() function can be used to create sets.

Note: to create an empty set you have to use set(), not {}; the latter creates an empty dictionary.

SETS

```
st = set([3,54,2,67,32,1])
print(st)
```

Sets are a powerful tool in Python since they have the ability to calculate differences and intersections between other sets. For example, say you have a list of participants in events A and B:

```
a = set([“Mary”,”Peter”,”Son”,”Mike”])
b = set([“Mike”,”Son”,”Danny”,”Kane”])
print(a.intersection(b)) #vice-versa
```

SETS

To find out which members attended only one of the events, use the "symmetric_difference" method:

```
print(a. symmetric_difference(b)); #vice-versa
```

To find out which members attended only one event and not the other, use the "difference" method:

```
print(a.difference(b)); #vice-versa
```

ASSIGNMENT

- Write a program to remove odd numbers from 1 to 50 in a list and replace it with the multiple by 2 of the odd number.
- Write a program that accepts a users name, address and gender. Format the text and print out to the user.