Tuple

Basic Theory:

A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.
Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also. For example –

```
tup1 = ('physics', 'chemistry', 1997, 2000);
tup2 = (1, 2, 3, 4, 5 );
tup3 = "a", "b", "c", "d";
```

The empty tuple is written as two parentheses containing nothing –

```
tup1 = ();
```

To write a tuple containing a single value you have to include a comma, even though there is only one value –

```
tup1 = (50,);
```

Like string indices, tuple indices start at 0, and they can be sliced, concatenated, and so on.

Differentiate list and tuple.

The main difference between lists and a tuples is the fact that lists are mutable whereas tuples are immutable. A mutable data type means that a python object of this type can be modified.

Examples:

a) Save a tuple. Print it. Delete it.

```
>>> tup=(1,2,3)
>>> print(tup)
(1, 2, 3)
>>> del(tup)
>>> print(tup)
```

b) Save two tuples. Concatenate them. Print it without using loops.

```
>>> tup=(1,2,3)
>>> next=(10,20,30)
>>> new=tup+next
>>> print(new)
(1, 2, 3, 10, 20, 30)
```

c) Suppose a tuple contains one item. Now store the same item for five times without using loop.

```
>>> tup=(5,)
>>> tup=tup*5
>>> print(tup)
(5, 5, 5, 5, 5)
```

d) Save a tuple. Print ith to jth item without using loop.

```
>>> tup=(1,2,3,10,20,30)
>>> print(tup[2:4])
(3, 10)
```

e) Convert a list in a tuple.

```
>>> tup=(1,2,3)
>>> new=list(tup)
>>> print(new)
[1, 2, 3]
```

f) Find maximum and minimum item in a tuple. Find the length of a tuple.

```
>>> tup=(1,2,3,10,20,30)
>>> max(tup)
30
>>> min(tup)
1
>>> len(tup)
6
```

1) Write a Python program to reverse a tuple.

```
#create a tuple
x = ("w3resource")
# Reversed the tuple
y = reversed(x)
print(tuple(y))
#create another tuple
x = (5, 10, 15, 20)
# Reversed the tuple
y = reversed(x)
print(tuple(y))
```

2) Write a Python program to count the elements in a list until an element is a tuple.

```
num = [10,20,30,(10,20),40]
ctr = 0
for n in num:
  if isinstance(n, tuple):
    break
  ctr += 1
print(ctr)
```

Write a Python program to find the index of an item of a tuple. Convert a string to a tuple. Check it for all possible parameters of index function. Check it for an item which is not present.

```
#create a tuple
tuplex = tuple("index tuple")
print(tuplex)
#get index of the first item whose value is passed as parameter
index = tuplex.index("p")
print(index)
#define the index from which you want to search
index = tuplex.index("p", 5)
print(index)
#define the segment of the tuple to be searched
index = tuplex.index("e", 3, 6)
print(index)
#if item not exists in the tuple return ValueError Exception
index = tuplex.index("y")
```

Write a program in Python to do slicing in all possible ways with all possible parameters, providing positive and negative values for step. Also, perform slicing from start and end both.

```
#create a tuple
tuplex = (2, 4, 3, 5, 4, 6, 7, 8, 6, 1)
#used tuple[start:stop] the start index is inclusive and the stop index
slice = tuplex[3:5]
#is exclusive
print(slice)
#if the start index isn't defined, is taken from the beg inning of the tuple
slice = tuplex[:6]
print(slice)
#if the end index isn't defined, is taken until the end of the tuple
slice = tuplex[5:]
print(slice)
#if neither is defined, returns the full tuple
slice = tuplex[:]
print(slice)
#The indexes can be defined with negative values
slice = tuplex[-8:-4]
print(slice)
#create another tuple
tuplex = tuple("HELLO WORLD")
print(tuplex)
#step specify an increment between the elements to cut of the tuple
```

```
#tuple[start:stop:step]
slice = tuplex[2:9:2]
print(slice)
#returns a tuple with a jump every 3 items
slice = tuplex[::4]
print(slice)
#when step is negative the jump is made back
slice = tuplex[9:2:-4]
print(slice)
```

Assignment:

1. Complete all the shell functions and programs given in the material. Write down your observations(1 line each).
2. Write a program in Python to do searching either linear or binary. The choice will be provided by the user.