PYTHON PROGRAMMING

MODULE 4 – PART 2

LIST AND OPERATIONS ON LIST

OUTLINE

List

- Ordered list of elements indexed from 0 to length-1.
- Repeated elements are allowed.
- Mutable (can be modified or changed)
- Can contain elements of different data types such as integer, float, string, list, tuple etc.

Creating List

Elements are written separated by commas and enclosed within square brackets

```
#empty list
a=[]

#different types
b=[1, 2, 'red', 2.0]

#creating list using list constructor. It takes only one argument, hence pass list values
in () or [ ].
c=list((1, 2, 'red', 2.0))

#can contain another list, tuple or even functions
d=[1, 2, [3, 4], (5, 6)]
```

Accessing List and its elements (slicing)

```
b=[ 1, 2, 'red', 2.0 ]
c=list(( 1, 2, ‘red’, 2.0 ))
d=[ 1, 2, [3, 4], (5, 6) ]
print(c)
print(b[0])
print(b[-1])

#slicing
print(b[1:3])
print(b[-3:-1])
print(b[-1:-3])
print(b[:])

print(d[2])
print(d[2][0],d[3][1])
```

OUTPUT

```
[1, 2, 'red', 2.0]
1
2.0
[2, 'red']
[2, 'red']
[]
[1, 2, 'red', 2.0]
[3, 4]
3
```

<u>Deleting list and its elements (pop(), del, remove())</u>

```
b=[1,2,’red’,2.0]

#pop can store deleted item, default parameter is -1,last item
item=b.pop(0)
print(item)
print(b)

#del cannot store deleted item hence cannot write in print function
del b[0]
print(b)

#cannot store and delete item by specifying value, not index
b.remove(‘red’)
print(b)

#deletes entire list
del b
```

<u>OUTPUT</u>

```
1
[2, 'red', 2.0]
['red', 2.0]
[2.0]
```

Adding and Updating List: (append(), insert(), extend())

```
b=[1,2,'red',2.0]
c=[3.2,4.2]
b[3]=2.1
print(b)
b[0:2]=[3,4]
print(b)

#parameters are position and element
b.insert(0,2)
print(b)

#adds item to the end of list
b.append(2.2)

#adds list as object to current list
b.append(c)
print(b)

#argument must be an iterable such as list,tuple,set etc
b.extend([3.3])

#adds elements of list as new elements to list
b.extend(c)
print(b)
```

OUTPUT

```
[1, 2, 'red', 2.1]
[3, 4, 'red', 2.1]
[2, 3, 4, 'red', 2.1]
[2, 3, 4, 'red', 2.1, 2.2, [3.2, 4.2]]
[2, 3, 4, 'red', 2.1, 2.2, [3.2, 4.2], 3.3, 3.2, 4.2]
```

Difference between append() and extend()

Both append() and extend() adds elements to the end of the list

append() can accept both iterable object and individual element as arguments.
extend() can only accept an iterable object as argument.

append() adds an iterable object as the same object to the end of the list.
extend() separates individual elements of the iterable object and adds as different elements to the end of the list.

In the previous code, list c = [3.2, 4.2] is added as a list itself into the current list b using append() method.
Also, extend() method separated elements 3.2, 4.2 and added as float type individual elements to b.

Copying list

Case 1

```
list1=[1,2,3]
list2=list1
print(list1)
print(list2)
list1.append(4)
print(list1)

'''both list1 & list2 points to
same list'''
print(list2)
```

OUTPUT

```
[1, 2, 3]
[1, 2, 3]
[1, 2, 3, 4]
[1, 2, 3, 4]
```

Case 2

```
list1=[1,2,3]
list2=list1[:]
print(list1)
print(list2)
list1.append(4)
print(list1)
print(list2)
```

OUTPUT

```
[1, 2, 3]
[1, 2, 3]
[1, 2, 3, 4]
[1, 2, 3]
```

Case 3

```
from copy import copy
list1=[1,2,3]
list2=copy(list1)
print(list1)
print(list2)
list1.append(4)
print(list1)
print(list2)
```

OUTPUT

```
[1, 2, 3]
[1, 2, 3]
[1, 2, 3, 4]
[1, 2, 3]
```

Basic Operations

len(), concatenation(+), repetition(*), membership(in, not in),iteration

```python
list1=[1,2,3]
list2=[4,5,6]
print(len(list1))
print(2 in list1,2 not in list1)
print(list1+list2)
print(list1*2)
for x in list1:
  print(x)
```

OUTPUT

```
3
True False
[1, 2, 3, 4, 5, 6]
[1, 2, 3, 1, 2, 3]
1
2
3
```

Functions Discussed Earlier

1) pop(position)
2) remove(element)
3) append(element)
4) insert(position, element)
5) extend(iterable)
6) copy( )
7) len(list)

Functions Discussed Next

1) index(element)
2) max(list)
3) min(list)
4) list(sequence)
5) clear( )
6) count(element)
7) reverse( )
8) sort(reverse=True/False, key=myFunc)
9) sorted(list, reverse=True/False, key=myFunc)

List built-in functions and methods

```
list1=[1,2,3,3]
print(list1.index(3))
print(max(list1),min(list1))
string1="abcdef"
a=list(string1)
print(a)
a.clear()
print(a)
print(list1.count(3))
list1.reverse()
print(list1)
```

OUTPUT

```
3 1
['a', 'b', 'c', 'd', 'e', 'f']
[]
2
[3, 3, 2, 1]
```

- sort(reverse=true/false, key=myFunc)
- sorted(list, reverse=true/false, key=myFunc)

For both built in functions :
reverse is optional and if ***true***, sorted in descending order
key is optional, if given, acts as a function based on which the sorting comparison occurs

Difference :
sort() changes original list
sorted(list) returns sorted list rather than changing the original

```
list1=["abcd","ade","acdefg"]
list1.sort()
print(list1)
list2=["abcd","ade","acdefg"]
print(sorted(list2))
print(list2)
print("--------------------")
list3=["abcd","ade","acdefg"]
list3.sort(reverse=True)
print(list3)
list3.sort(key=len)
print(list3)
print(sorted(list3,reverse=True,key=len))
```

OUTPUT

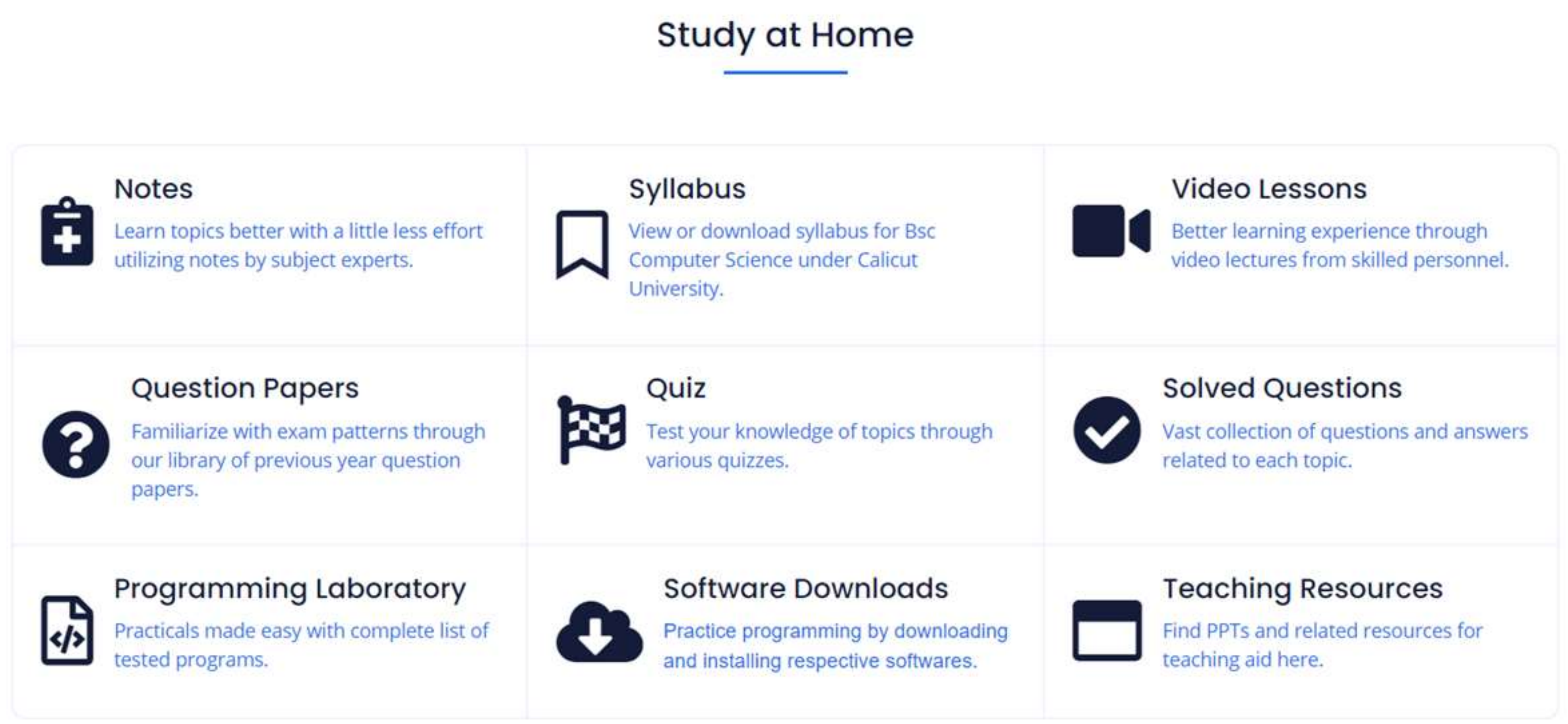
```
['abcd', 'acdefg', 'ade']
['abcd', 'acdefg', 'ade']
['abcd', 'ade', 'acdefg']
--------------------
['ade', 'acdefg', 'abcd']
['ade', 'abcd', 'acdefg']
['acdefg', 'abcd', 'ade']
```

Selection sort

```python
list1=[30,20,15,10,50]
print("original list : ",list1)
n=len(list1)
for i in range(n-1):
    small=i
    for j in range(i+1,n):
        if list1[j]<list1[small]:
            small=j
    if i!=small:
        temp=list1[i]
        list1[i]=list1[small]
        list1[small]=temp
print("sorted list : ",list1)
```

OUTPUT

```
original list :  [30, 20, 15, 10, 50]
sorted list :  [10, 15, 20, 30, 50]
```



Study at Home

Notes
Learn topics better with a little less effort utilizing notes by subject experts.

Syllabus
View or download syllabus for Bsc Computer Science under Calicut University.

Video Lessons
Better learning experience through video lectures from skilled personnel.

Question Papers
Familiarize with exam patterns through our library of previous year question papers.

Quiz
Test your knowledge of topics through various quizzes.

Solved Questions
Vast collection of questions and answers related to each topic.

Programming Laboratory
Practicals made easy with complete list of tested programs.

Software Downloads
Practice programming by downloading and installing respective softwares.

Teaching Resources
Find PPTs and related resources for teaching aid here.

www.teachics.org

teachics.

The Computer Science Learning Platform.

Thank You