NOTE FOR TYPESETTER: Please do not change the font and spacing of text written in courier.

Instructions for Candidates

Attempt any Four Questions. All Questions carry equal marks.

1. Differentiate between (i) class and object (ii) constructor and destructor.

    Is a data member accessible outside the class? If yes, how to access the data member? If not, why?

    Define a class `Box` that has the three data members `Width, Height` and `Depth` and following methods:

    Methods:
    *   Constructor to initialize the data members `Width, Height` and `Depth`
    *   `Volume` – This computes volume of the box
    *   `__str__` function to generate a string that may be used to display the data members of the box.

    Also, write Python statements to:
    *   Create an object of the class `Box` with suitable values.
    *   Invoke the method `Volume`
    *   Print all the data members of the class

2. Apply the Bubble Sort scheme of sorting on the following list to sort it in ascending order. Show the content of the list after applying each iteration of Bubble sort: `=[74, 31, 82, 25, 63, 96, 49, 11]`.
    How many iterations are required to sort the list? What is the number of comparisons performed in each iteration?
    Apply Binary search to search for the item 50 in the sorted list. You should show the computation of the index at which the value is compared with 50. When will you prefer to use binary search over linear search? Justify your answer.

3.
- The following operations are performed on a stack in the sequence given here (assume stack is initially empty): `push('Red')`, `push('Green')`, `pop()`, `push('White')`, `pop()`, `pop()`. Show the content of the stack after each operation. What will happen if one more pop operation is performed on the stack? Can this condition be prevented?

- A print application processes the print jobs in the first come, first served manner. Suggest a suitable data structure for the print application. What are the permissible operations on the suggested data structure?

4.
- Write a Python function `moveOdd(L1,L2)` that accepts two lists `L1` and `L2`. List `L1` consists of integers and list `L2` is initially empty. The function `moveOdd` removes the odd integers from the list `L1` and moves them to the list `L2`. Write python statements to apply the function `moveOdd` on the list `L1`: `[31, 28, 63, 86, 15, 79, 44, 52]`. What will be the content of list `L1` and `L2` after the execution of `moveOdd` function for the list `L1`?

- Use modified list `L1` and write the python command to:
    o display last three elements,
    o middle two elements
    o delete element at index 1
    o reverse the order of the elements in the list
    o insert the list [21, 56] at the end of the list

5.
- Write a Python function `smallerXY(X,Y)` that accepts two integers `X` and `Y` and returns the smaller of two. Write another function `smallerXYZ(X,Y,Z)` that uses the function `smallerXY` to find minimum of three numbers `X,Y,Z`.

- Give the output that will be produced on execution of following code segment :

```
a = 12              # 12 in binary: 0000 1100
b = 7               # 7 in binary:  0000 0111
b = a & b
print(b)
a = a | b
print(a)
b = a ^ b
print(b)
a = ~b
print(a)
print(b<<3)
```

- Give the output that will be produced on execution of following code segment :

```python
def add(a=3, b=5, c=7):
    return(a+b+c)

print(add())
print(add(7))
print(add(6,8))
print(add(2,4,5))
print(add(c=9))
```

6.
- Write a Python program that contains function `main` and function `occurs(s,ch)`. Function `occurs(s,ch)` accepts a string `s` and a character `ch` as arguments and returns the number of occurrences of character `ch` in the string `s`. The main function reads a string and a character from the keyboard, calls function `occurs` and prints the number of occurrences of the character. Do not use a built-in function to find occurrence of the character.

- Give the output that will be produced on execution of following code segment:

```python
s1 = 'I am an instance of a string!!'
print(s1[-13:-21])
print(s1.title())
print(s1.find('of'))
print(s1.isalpha())
print(s1.count('a'))
```

- Consider:

```python
set1 = set(['P','Q','M','O' ])
set2 = set(['R','P','N','S','O' ])
```

Give the output that will be produced on execution of the following code segment:

```python
print(set2 | set1)
print(set2 - set1)
print(set2 & set1)
print('X' in set2)
```