**Spring 2017 OOP Midterm Lab Exam**

|  |
| --- |
| Duration: 90 MinutesThere is a 3-hour consecutive slot for OOP exam1.5 hours are to be given for Theory and 1.5 hours for lab exam |

1. This is an open book/open notes exam.
2. Exchange of books, notes and calculators is not allowed.
3. Submissions are to be done on LMS, instructions are written there.

**Question 1:**

Create a base class called Vehicle that has the manufacturer’s name (type string or char\*), number of cylinders in the engine (type int), and owner’s name (type string or char\*). The class contains a parametrized constructor without default values of the parameters. There is a display() function to display the values of all the attributes. Then create the following derived classes:

1. A class called Truck that is derived from Vehicle and has additional properties like the load capacity in tons (type double since it may contain a fractional part) and towing (pulling) capacity in kilograms (type int). Over-ride the display() function to display all the attributes of Truck (it should display inherited members too).
2. A class called Bus that is derived from Vehicle and has additional properties like the no of passengers (type int) and driver’s name. Over-ride the display() function to display all the attributes of Bus (it should display inherited members too).

Write a main() creates one object of Truck and Bus each. It then displays all the information related to that objects. Note that there is no getter and setter of any attribute of any class.

**Question 2:**

Implement an *IntList* class so that following main function can run properly. Understanding the question is a part of this problem.

int main()

{

IntList a;

a.insert(12);

int arr[] = {2,3,4,5};

IntList b(arr,4);

IntList c = a;

a = b;

b.reverse(); //Should reverse all numbers in list.

cout << a;

IntList c = a + b; *// addition of intList will add list ‘a’*

*// elements to the elements of list ‘b’ only corresponding*

*// elements will be added. Program would therefore work*

*// even if the size of both lists is different*

return 0;

}

**Note**: IntList class should have a pointer to a dynamically allocated array to store integers. Make sure that the pointers have their respective memories and there is no memory leakage.