<table>
<thead>
<tr>
<th>Tick the Relevant</th>
<th>Computer Science B.Sc. Program ABET Student Outcomes</th>
<th>Question No. Relevant Is Hyperlinked</th>
<th>Covering %</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>a) Apply knowledge of computing and mathematics appropriate to the discipline;</td>
<td>1</td>
<td>80%</td>
</tr>
<tr>
<td></td>
<td>b) Analyze a problem, and identify and define the computing requirements appropriate to its solution</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>c) Design, implement and evaluate a computer-based system, process, component, or program to meet desired needs;</td>
<td>5</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>d) Function effectively on teams to accomplish a common goal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>e) Understanding of professional, ethical, legal, security, and social issues and responsibilities;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>f) Communicate effectively with a range of audiences;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>g) Analyze the local and global impact of computing on individuals, organizations and society;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>h) Recognition of the need for, and an ability to engage in, continuing professional development;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>i) Use current techniques, skills, and tools necessary for computing practices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>j) Apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>k) Apply design and development principles in the construction of software systems of varying complexity;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Project</td>
<td></td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>Lab Exam</td>
<td></td>
<td>50%</td>
</tr>
<tr>
<td>Question No.</td>
<td>Relevant Student Outcome</td>
<td>SO is Covered by %</td>
<td>Full Mark</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------</td>
<td>--------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>CO 1</td>
<td>(a)</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>CO4</td>
<td>(a)</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>(c)</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I certify that the work contained within this assignment is all my own work and referenced where required.

Student Signature: 
Date: 

Feedback Received: 
Student Signature: 
Date:
Question 1: Consider the following UML class diagram

The interface **FinanceAgency** contains the following methods:
- `display()`
- `computeYearlyAmount()`

The class **Insurance** contains the following attributes and methods:
- `personName`: The name of the insured person.
- `unitAmount`: The insurance unit amount.
- `yearlyAmount`: The amount of insurance for one year.
- `display()`: displays all the attributes of the Insurance.
- `computeYearlyAmount()`: computes the amount of insurance for one year.
The concrete class **HomeInsurance** contains the following attributes and methods:
- **surface**: the surface of the home to be insured.
- **display()**: displays all the attributes of the HomeInsurance.
- **computeYearlyAmount()**: computes the amount of insurance for one year as follows:
  \[ \text{yearlyAmount} = \text{unitAmount} \times \text{surface}. \]

The concrete class **CarInsurance** contains the following attributes and methods:
- **NbOfCylinders**: the number of cylinders of the car to be insured.
- **display()**: displays all the attributes of the CarInsurance.
- **computeYearlyAmount()**: computes the amount of insurance for one year as follows:
  \[ \text{yearlyAmount} = \text{unitAmount} \times \text{NbOfCylinders}. \]

Write in Java the interface **FinanceAgency**, the classes **Insurance** and **HomeInsurance**.

Note: You can call getters and setters without implementation. Assume that the classes **SalesCars** and **CarInsurance** are implemented.

**Answer**:

```java
// Implementation details...
```
Question 2: We want to manage a linked list of CarInsurance objects. Consider the following UML class diagram.

Class List: The class List contains the following attributes and methods:
- name : name of the List
- first : first Node of the List
- public String findHighestInsurance ()
  This method returns the name of the person having the greatest insurance yearly amount. Handle the case where the list is empty.

- public void deleteByName (String pname )
  This method deletes the node containing an Insurance object with a name equal to the argument pname. If more than one node meets the condition, only the first one is deleted. In your solution, in addition to the general case, consider the following special cases, where:
  - the list is empty
  - the list contains only one element
  - the name does not exist in the list.
  Write in Java the above two methods.
Question 3: Consider a file called “insurance.data” containing insurance objects.

Public static List computeAndStore (String filename)
This method reads insurance objects from the object file “insurance.data”, returns an object of class List (declared in question 2) containing only CarInsurance objects having the number of cylinders less than 12 AND stores in a file called filename all HomeInsurance objects having surface greater than 170m².

Write in Java the above method.
Question 4: Consider a stack containing Insurance objects. Write a method to store in the queue \textit{homeInsurance} objects first followed by \textit{CarInsurance} objects. (see example below):

\begin{verbatim}
public static Queue ReorderInsuranceObjects(Stack ST);
\end{verbatim}

\textbf{Example:}

\begin{tabular}{|l|}
\hline
The received stack ST \\
\hline
\end{tabular}

\begin{tabular}{|l|}
\hline
\textit{HomeInsurance} Obj1 \\
\hline
\textit{CarInsurance} Obj1 \\
\hline
\textit{CarInsurance} Obj2 \\
\hline
\textit{HomeInsurance} Obj2 \\
\hline
\textit{CarInsurance} Obj3 \\
\hline
\textit{CarInsurance} Obj4 \\
\hline
\end{tabular}

\begin{tabular}{|l|}
\hline
The returned queue \\
\hline
\end{tabular}

\begin{tabular}{|l|l|l|l|l|l|l|}
\hline
\textit{HomeInsurance} Obj1 & \textit{HomeInsurance} Obj2 & \textit{CarInsurance} Obj1 & \textit{CarInsurance} Obj2 & \textit{CarInsurance} Obj3 & \textit{CarInsurance} Obj4 & => \\
\hline
\end{tabular}

Assume that the methods of the class \textit{Stack} and the class \textit{Queue} are implemented. \textbf{Hint:} You can use a temporary stack or queue to write the program.

\textbf{Answer:}

................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
................................................................................................................................................
Question 5:
We have a park containing vehicles. The vehicles are of two types: personal cars and trucks. We would like also to represent the types of wheels used by all the cars and trucks.

All the vehicles are described by the number of passengers, and their speed.

Personal cars are characterized by the number of passengers, the speed and whether it is taxi or not.

The trucks are described by the number of passengers, the speed and the capacity of transportation.

The wheels are characterized by their type, size and manufacturing year.

Draw the UML diagram of the classes Park, Vehicle, Truck, Car and Wheels described above. In this diagram show clearly the type of relationship between these classes.

.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................
.................................................................

CSC 113 Final Exam – Fall 2010-2011