|  |  |  |
| --- | --- | --- |
| American University of SharjahSchool of Engineering Department of Computer Engineering  P. O. Box 26666  Sharjah, UAE  **Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **ID : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |  | **Instructor:** Omar Arif **Office**: ESB-2178  **Phone**: 971-6-515 4821  **e-mail**: oarif@aus.edu  **Semester**: |

COE312 –

**The quiz open book and notes and you can use the Internet. You are not allowed to consult other individuals or your classmates.**

PLEASE UPLOAD YOUR ANSWER USING THIS **WORD** FILE. DO NOT UPLOAD ANY ZIPPED OR PDF FILE. ONE WORD FILE ONLY.

The question(s) have been written in a manner such that it is not possible for two students to have the same solution. Therefore, please refrain from the temptation of copying code and changing order of and names of variables, etc. AUS code of conduct will be strictly enforced, and no violation will be tolerated as per AUS policy. Please note that the AUS code does not discriminate between who copied from whom so it is not advisable to share your solution.

**Q1. (10 points)** Implement the State pattern for a vending machine. The vending machine has three states: NoCoinState, HasCoinState, and DispensingState. Each state has the following behavior:

* **NoCoinState:** This state represents the initial state of the vending machine. It does not allow product selection or dispensing without inserting a coin in to the vending machine. When there is coin in the vending machine, the state transitions to the HasCoinState.
* **HasCoinState:** This state represents the state when a coin has been inserted into the vending machine. This state transitions to the HasCoinState.
* **DispensingState:** This state represents the state when a product is being dispensed from the vending machine. This state transitions back to the NoCoinState.

Use **State Design Pattern** to implement the above behavior.

You must run and provide output on the following program (without any changes to the program). You are not allowed to make any changes to this program.

public class Main {

public static void main(String[] args) {

// TODO Auto-generated method stub

VendingMachine vendingMachine = new VendingMachine();

vendingMachine.printStatus(); // Currently in NoCoinState

vendingMachine.next(); // Cannot transition from NoCoinState without inserting a coin.

vendingMachine.insertCoin(); // inserting coin

vendingMachine.next();

vendingMachine.printStatus();

vendingMachine.next();

vendingMachine.printStatus();

vendingMachine.next();

vendingMachine.next();

}

}

**Your program output should match the following output:**

Currently in NoCoinState

Cannot transition from NoCoinState without inserting a coin.

Transitioning from NoCoinState to HasCoinState

Currently in HasCoinState

Transitioning from HasCoinState to DispensingState

Currently in DispensingState

Transition from DispensingState to NoCointState

Cannot transition from NoCoinState without inserting a coin.

Please provide 1) formatted code and 2) screenshots of your running program. Not providing a screenshot (with or without errors) will limit your score to below 3/10.

|  |
| --- |
| **Solution: (Paste formatted code in this box).** |

|  |
| --- |
| **Solution: (Paste the screenshot in this box).** |

**Grading Rubric:**

|  |  |  |  |
| --- | --- | --- | --- |
| **0-3** | **4-6** | **7-8** | **9-10** |
| * The program does not compile or run but there is some notion of a solution OR * No screenshot is provided. | * Some of the program works but not fully. * Bad design practices have been used. | * The program partially matches the output provided. * Clean design practices have been followed. | * The program exactly matches the output. * Clean design practices have been followed. |