CO Assessment Plan

CSC 2209 – Object Oriented Programming 1 – Fall 2018-19

Mr. Mohaimen-Bin-Noor

Dr. Kamruddin Md. Nur

A. Z. M. Ehtesham Chowdhury

Dr Md Mahbub Chowdhury Mishu

DEPARTMENT OF CSE

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

CO Assessment Plan

CSC 2209 – Object Oriented Programming 1 – Fall

Course Description

Course Description is below:

- Develop classes and describe how to declare a class
- Create Java technology applications that leverage the object-oriented features of the Java language, such as encapsulation, inheritance, polymorphism and abstraction
- Execute Java applications from the command line
- Use Java technology data types and expressions
- Use Java technology flow control constructs
- Use arrays and other data collections
- Use the concept of package
- Implement error-handling techniques using exception handling
- Perform multiple operations on database tables, including creating, reading, updating and deleting using both JDBC
- Create an event-driven graphical user interface (GUI) using Swing components: panels, buttons, labels, text fields, and text areas
- Implement input/output (I/O) functionality to read from and write to data and text files and understand advanced I/O streams
- Create multithreaded programs

Credit- 3 Credits (2 hrs theory and 3 hrs Lab per week)
Nature- Core Course for CS, CSE, CSSE, SE, CIS
Prerequisite- CSC 2105: Data Structure

Faculties for Current Semester

1. Mohaimen-Bin-Noor (Course Convener)
2. Dr. Kamruddin Md. Nur
3. A. Z. M. Ehtesham Chowdhury

Preface

This document is intended to produce a standard, comprehensive, and coherent documentation of the CO assessment Plan.

The plan is developed by strictly following the guideline of OBE, which is produced the OBE implementation Committee, CSE, AIUB.

The plan for this course is produced by the responsible Faculties, as enlisted in this document.

Table of Contents

Chapter 1: The Mapping

1.1 COs' for the course

		Level of Domain*				PO
CO	**Description**	**C**	**P**	**A**	**S**	**Assessed****
CO1	*Demonstrate* OOP concepts using Java	3				2
CO2	*Analyze a simple to complex problem using OOP principles*		4		TS	3
CO3	*Develop solutions for real life complex problems using the concepts of Java*		6		TS	5
CO4	*Explain the appropriateness of their developed solution in conjunction with concepts of OOP and Java*			5		9

C: Cognitive; P: Psychomotor; A: Affective; S: Soft-skills (CT: Critical Thinking, TS: Teamwork)
****The numbers under the 'Level of Domain' columns represent the level of Bloom's Taxonomy each CO corresponds to.***
***** The numbers under the 'PO Assessed' column represent the PO each CO corresponds to.***

1.2 CO - Assessment Method: Mapping

The mapping between Course Outcome(s) (COs) and The Selected Assessment method(s) is shown below:

CO	Description	Learning Domain	Assessment Method
CO1	*Demonstrate* OOP concepts using Java	Cognitive	**Mid Term Exam**
CO2	*Analyze a simple to complex problem using OOP principles*	Psychomotor	**Mid Term Lab Exam**
CO3	*Develop solutions for real life complex problems using the concepts of Java*	Psychomotor	**Final Term Project**
CO4	*Explain the appropriateness of their developed solution in conjunction with concepts of OOP and Java*	Affective	**Final Term Project Viva**

1.3 Assessment Method – Rubrics: Mapping

The mapping between Assessment method(s) and Evaluation Rubric(s) is shown below:

CO	Description	Learning Domain	Assessment Method	Assessment Rubric
CO1	*Demonstrate* OOP concepts using Java	*Cognitive*	**Mid Term Exam**	Rubric for Term Question
CO2	*Analyze a simple to complex problem using OOP principles*	*Psychomotor*	**Mid Term Lab Exam**	Rubric for Lab Evaluation
CO3	*Develop solutions for real life complex problems using the concepts of Java*	*Psychomotor*	**Final Term Project**	Rubric for Project
CO4	*Explain the appropriateness of their developed solution in conjunction with concepts of OOP and Java*	*Affective*	**Final Term Project Viva**	Rubric for Viva

Chapter 2: The Selected Rubrics

This chapter lists each Rubric selected for evaluating the COs.

CO1	*Demonstrate OOP concepts using Java*	***Cognitive***	Mid Term Exam	**Rubric for Term Question**

Following is the finalized Rubric for the Mid Term Exam:

Marking Criteria	Marks Distribution (Maximum 5X4=20)				Acquired Marks
	Inadequate (1-2)	Satisfactory (3)	Good (4)	Excellent (5)	
Definition	• Student vaguely define the terms or the concept.	• Definition provided with partial relevance to the subject matter.	• Correctly define the terms. May miss minor detail.	• Comprehensively defines the terms.	
Logical Argument	• No logical arguments / explanation supporting the definition.	• Offers lossy related arguments.	• Strong argument/ explanation offered.	• Comprehensive argument presented to clarify the concept.	
Relevant Example	• No representative example. • No code snippet provided.	• Correctly identify/ indicate towards real-life example. • Code snippet provided.	• Real-life example is strongly connected towards the definition. • Well defined/ structured code snippet provided.	• Comprehensively defend with real life example. • Well documented code snipped provided.	
Organization of the Answer	• Present a confusing organization of definition, supporting arguments, and real-life example. Sentences rambling, and details are repeated	• Some problems of organizing the answer in a logical order of defining, elaborating, and providing real-life examples	• Presents most of the details in a logical flow of organization in definition, details, and example	• Consistently presents a logical and effective organization of definition, details, and real-life example of the topic	
				Acquired Marks:	
				CO Pass / Fail:	

CO2	*Analyze a simple to complex problem using OOP principles*	***Psychomotor***	Mid Term Lab Exam	**Rubric for Lab Evaluation**

Following is the finalized Rubric for Mid Term Lab Exam:

Criteria	Marks distribution (Max 4X5 = 20)				Acquired Marks
	Inadequate (1)	Competent (2)	Good (3)	Excellent (4)	
Program Execution	• Program Executes with warnings and/or runtime errors • User prompts are misleading or non-existent. • Poorly describe the solution	• Executes without errors • User prompts contain little information • Moderately explains the overall program.	• Executes without errors. • User prompts are understandable, minimum use of symbols or spacing in output. • Adequately, describe the solution and reason for the solution.	• Executes without errors • Excellent user prompts, good use of symbols, spacing in output. • Comprehensive explanation of the solution.	
Correct output	• Output is incorrect	• Output has multiple errors	• Output has minor errors	• Program displays correct output with no errors	
Design of output	• Output is poorly designed	• Program does not display the required output	• Program displays minimally expected output	• Program displays more than expected	
Design of logic	• Logic is incorrect	• Program has significant logic errors	• Program has slight logic errors that do not significantly affect the results	• Program is logically well designed	
Standards	• Program is poorly written	• Several inappropriate design choices (i.e. poor variable names, improper indentation)	• Few inappropriate design choices (i.e. poor variable names, improper indentation)	• Program is stylistically well designed	
				Acquired Marks:	
				CO Pass/Fail:	

CO3	*Develop solutions for real life complex problems using the concepts of Java*	***Psychomotor***	Final Term Project	**Rubric for Project**

Following is the finalized Rubric for the Project:

Criteria	**Marks distribution (Max 5X6 = 30)**				**Acquired Marks**
	Adequate (1)	**Competent (2-3)**	**Good (4)**	**Excellent (5)**	
Organization	• Audience cannot understand presentation because of no sequence of information.	• Audience has difficulty following because student jumps around presentation	• Student presents information in logical sequence which audience can follow.	• Student presents information in logical, interesting sequence which attract the audience.	
Content Knowledge	• Student does not have grasp of information and cannot answer the questions about subject.	• Student is uncomfortable with information and is able to answer only rudimentary questions.	• Student is at ease with content, but fails to elaborate.	• Student demonstrates full knowledge (more than required) with explanations and elaboration.	
Delivery	• Student incorrectly pronounces terms, and speaks too quietly for audiences in the back of the class to hear.	• Student incorrectly pronounces term. Audience members have difficulty hearing presentation	• Student's voice is clear. Student pronounce most words correctly.	• Student uses a clear voice and correct, precise pronunciation of terms.	
System's design with Requirements Specification	• No or inadequate system design and documentation. E.g., no UML model for software design.	• Student provide a basic design with no documentation for the system. E.g., use basic UML concepts that rarely support logical design of system.	• Provide system design with adequate detail with requirement specification documentation. E.g., use UML diagram at a detailed level that highly relevant to system's logical design.	• Extensive system design with Comprehensive documentation. E.g., uses UML for detailed system design supported by a clear Requirement specification documentation.	

				• The design offers strong mapping with the systems logical design.	
Comprehensive of the System Implementation against the Requirements	• E.g., irrational choice of requirements while coding, that leads to inappropriate solution.	• Requirements is reflected but can't be followed in proper sequence.	• Sequential reflection of requirements but not all requirements are followed.	• Requirements is maintained in full-fledged and proper sequence.	
Adequate conceptualization of the learned concepts in the course	• No sync between content and developed project work.	• Little idea about the concepts of course that is included in the project work.	• Can relate the concepts of course with the project work.	• Concepts of the course that is necessary for the project is covered properly.	
				Acquired marks:	
				CO Pass / Fail:	

CO4	*Explain the appropriateness of their developed solution in conjunction with concepts of OOP and Java*	***Affective***	Final Term Project Viva	**Rubric for Viva**

Following is the finalized Rubric for the Viva:

Criteria	**Marks distribution (Max 4X5 = 20)**				**Acquired Marks**
	Inadequate (1)	**Competent (2)**	**Good (3)**	**Excellent (4)**	
Argument	• Does not articulate a position or argument	• Articulates a position or argument that is unfocused or ambiguous	• Articulates a position or argument that is incomplete or limited in scope	• Clearly articulates a position or argument	
Evidence / Reference	• Presents a lot of inaccurate and/or irrelevant evidence • Doesn't present enough evidence to support argument, even when prompted repeatedly	• Presents evidence that is somewhat inaccurate and/or irrelevant, but corrects when prompted • Does not present enough evidence to support argument, but augments when prompted	• Presents evidence that is mostly relevant and/or mostly accurate • Presents limited evidence to support argument	• Presents evidence that is relevant and accurate • Presents sufficient amount of evidence to support argument	
Implications	• Doesn't discuss the implications of the argument or position	• Discusses minor implications (missing the major ones) OR does not discuss major implications adequately	• Adequately discusses some of the major implications of the position	• Fully discusses the major implications of the argument or position	
Prompting	• Did not have to prompt with probing questions at all	• Prompted minimally (one or two probing questions)	• Prompted moderately (a series of probing questions)	• Prompted highly with almost all probing questions	

Overall Understanding	• Shows no understanding of the topic and no argument per the categories above.	• Shows a superficial understanding of the topic, argument not developed enough per the categories above.	• Shows a limited understanding of the topic, not quite a fully developed argument per the categories above.	• Shows a deep/robust understanding of the topic with a fully developed argument per the categories above.	
Acquired Marks:					
CO Pass/Fail:					

Chapter 3: Assessment Materials

This chapter details the assessment material (e.g., midterm questions, lab exam questions) produced by the faculties to evaluate the COs.

CO1	Demonstrate OOP concepts using Java	Cognitive	Mid Term Exam

Q1	`class Calculator{`
	`        int result;`
	`         static float jvm;`
	`        public void out(){ System.out.print(result); }`
	`        public void add(int a, int b){ result=a+b; }`
	`        public void add(int a, int b, int c){ result=a+b+c; }`
	`}`
	`class MainClass{`
	`       public static void main(String[] str){`
	`       Calculator c = new Calculator();`
	`        c.out(); c.add(4,5);`
	`        c.out(); c.add(12,7); Calculator.jvm = 20;`
	`        c.out(); c.add(1,4);`
	`         c.out();`
	`        }`
	`}`
	
	a) Find the **instance** variables, **class** variables, **local** variables. Justify your answer.
	b) Is there any use of method overloading and/or method overriding in the above program? Justify your answer.		

Q2	a) What are **static** variable and a **static** method? When should you prefer to declare an attribute as static? Justify your answer. b) What is the appropriate way to call a class method and an instance method?		
Q3	a) Which keyword can be used to enforce inheritance and method over ridding? Give example. b) Which keyword can be used to prevent inheritance and method over ridding? Give example.		
Q4	**Read The following scenario and Answer the following Questions:** Your friend is a C++ programmer. Recently, he has started practicing Java. In his first java program he wrote the following statement to declare his main method: ***public void main( )*** This statement generated an error while compiling. a) Write the proper statement to declare the main method in java. b) Why must your friend correct his statement according to your statement?		
Q5	a) What are the usage of the keywords: ***this, this( ), super and super( )***? Give suitable example by writing a code. b) In how many ways, a final attribute can be initialized? Give a suitable example.		
Q6	a) What are the access modifiers in Java? b) Sort the access modifiers in Java from least accessible to most accessible. Justify your answer.		

Q7

Suppose that, you have written the following code:

```
class  Adder {
      int   x, y;
       static   int  s;
       public static void test(){
             System.out.println("Test OOP1: "+x);
      }
      int  add( int x,  int y ) {
             System.out.println("Add OOP1: "+s);
             return x + y;
      }
      public static void main( String args[] ) {
             int x = 5, y =10;
             System.out.println(add(x,y));
      }
}
```

a) Is there any problem in the above program? If yes, specify two alternative solutions to solve the problem.
b) Write codes for each solution.

Q8

Read The following scenario and Answer the following Questions:
Once upon a time there were two friends- Bishal and Mahfuz. Bishal likes programming very much and loves Java. On the other hand, Mahfuz is not so interested in coding. Tomorrow is their Java Mid Term Exam. Both Bishal and Mahfuz started taking preparations from the evening. Suddenly Mahfuz found that though he has installed JRE, java codes are not compiling in his laptop. He discussed the problem with Bishal. Bishal knows that if he compiles java codes in his pc and give the class files to Mahfuz, Mahfuz will be able to run them in his laptop and take preparations for the exam. Bishal helped Mahfuz just like a good friend.

- What are the full meaning of JDK, JVM, JRE and JIT.
- Which characteristic of Java has been illustrated in the above story? Justify your answer.

Q9

a) What is the difference between creating a String object with the new keyword and without the keyword?
b) Explain Mutability of String.

Q10

Read the following scenario and Answer the following questions:

In a Computer shop, four kinds of Computer **Accessories** are sold, they are: **Keyboard**, **Mouse**, **SoundBox** and **Monitor**. All the Accessories have an ***id, modelNo, manufacturerName*** and ***price***. Data type for manufacturerName is String and it can be maximum 20 in length. The shop is managed by **Employees**. There are both **PartTimeEmployees** and **FullTimeEmployees** in the shop.

a) Draw inheritance tree/trees to demonstrate any inheritance relationship from the above scenario.
b) If you are told to create an object of the Accessories class that takes at most 50bytes of memory, what will be the data types for the attributes id, modelNo and price? Justify your answer.

CO2	Analyze a simple to complex problem using OOP principles	Psychomotor	Mid Term Lab Exam

Q1

Account
int accountNumber String accountHolderName double balance
Account( ) Account(int accountNumber, String accountHolderName, double balance) void setAccountNumber(int accountNumber) void setAccountHolderName(String accountHolderName) void setBalance(double balance) int getAccountNumber( ) String getAccountHolderName( ) double getBalance( ) void depositMoney(double amount) void withdrawMoney(double amount) void transferMoney(double amount, Account a)

Bank
String bankName String branchName Account [ ]accounts
Bank( ) Bank(String bankName, String branchName, int sizeOfArray) void setBankName (String bankName) void setBranchName (String branchName) String getBankName( ) String getBranchName( ) Account getAccount(int accountNumber) void addAccount(Account a) void removeAccount(Account a)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (b1) of the *Bank* class by using its parameterized constructor.
- Create **four** objects (a1, a2, a3, a4) of the *Account* class by using its parameterized constructor.
- Add **a1, a2** and **a3** in the accounts[ ] array.
- Display the bankName and branchName. Also, display the details of all the accounts in the accounts[ ] array by using a loop.
- From the accounts[ ] array, search for the a2 account and deposit $500.
- From the accounts[ ] array, search for the a3 account and withdraw $500.
- From the accounts[ ] array, search for the a2 account and transfer $500 to the a3 account.
- Display the details of all the accounts in the accounts[ ] array by using a loop.
- Remove the account a1 from the array and then add the account a4.

Q2

Write the following classes:

Book
int bookId String bookTitle String authorName int numberOfCopies
Book( ) Book(int bookId, String bookTitle, String authorName, int numberOfCopies) void setBookId(int bookId) void setBookTitle(String bookTitle) void setAuthorName(String authorName) void setNumberOfCopies(int numberOfCopies) int getBookId ( ) String getBookTitle( ) String getAuthorName( ) int getNumberOfCopies( ) void addNumberOfCopies(int copy) void reduceNumberOfCopies(int copy)

Library
String libraryName Book [ ]books
Library( ) Library(String libraryName, int sizeOfArray) void setLibraryName (String libraryName) String getBankName( ) Book getBook(int bookId) void addBook(Book b) void removeBook(Book b)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (l1) of the *Library* class by using its parameterized constructor.
- Create **four** objects (b1, b2, b3, b4) of the *Book* class by using its parameterized constructor.
- Add **b1, b2** and **b3** in the books[ ] array.
- Display the libraryName. Also, display the details of all the books in the books[ ] array by using a loop.
- From the books[ ] array, search for the b2 book and add 5 copies of it.
- From the books[ ] array, search for the b3 book and reduce 5 copies of it.
- Display the details of all the books in the books[ ] array by using a loop.
- Remove the book b1 from the array and then add the book b4.
- Display the details of all the books in the books[ ] array by using a loop.

Q3

Write the following classes:

Product
int productId String productName double price int availableQuantity
Product( ) Product(int productId, String productName, double price, int availableQuantity) void setProductId(int productId) void setProductName(String productName) void setPrice(double price) void setAvailableQuantity(int availableQuantity) int getProductId( ) String getProductName( ) double getPrice( ) int getAvailableQuantity( ) void buyProducts(int quantity) void sellProducts(int quantity)

Shop
String shopName Product [ ]products
Shop( ) Shop(String shopName, int sizeOfArray) void setShopName(String shopName) String getShopName( ) Product getProduct(int productId) void addProduct(Product p) void removeProduct(Product p)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (s1) of the *Shop* class by using its parameterized constructor.
- Create **four** objects (p1, p2, p3, p4) of the *Product* class by using its parameterized constructor.
- Add **p1, p2** and **p3** in the products[ ] array.
- Display the shopName. Also, display the details of all the products in the products[ ] array by using a loop.
- From the products[ ] array, search for the p2 product and buy 5 quantities of it.
- From the products[ ] array, search for the p3 product and sell 5 quantities of it.
- Display the details of all the products in the products[ ] array by using a loop.
- Remove the product p1 from the array and then add the product p4.
- Display the details of all the products in the products[ ] array by using a loop.

Q4

Write the following classes:

Item
int itemId String itemName int availableQuantity
Item( ) Item(int itemtId, String itemName, int availableQuantity) void setItemId(int itemId) void setItemName(String itemName) void setAvailableQuantity(int availableQuantity) int getItemId ( ) String getItemName( ) int getAvailableQuantity( ) void addQuantity(int quantity) void removeQuantity(int quantity)

Inventory
String inventoryName Item [ ]items
Inventory( ) Inventory(String inventoryName, int sizeOfArray) void setInventoryName(String inventoryName) String getInventoryName( ) Item getItem(int itemId) void addItem(Item i) void removeItem(Item i)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (in1) of the *Inventory* class by using its parameterized constructor.
- Create **four** objects (i1, p2, p3, p4) of the *Item* class by using its parameterized constructor.
- Add **i1, i2** and **i3** in the items[ ] array.
- Display the inventoryName. Also, display the details of all the items in the items[ ] array by using a loop.
- From the items[ ] array, search for the i2 item and add 5 quantities of it.
- From the items[ ] array, search for the i3 item and remove 5 quantities of it.
- Display the details of all the items in the items[ ] array by using a loop.
- Remove the item i1 from the array and then add the item i4.
- Display the details of all the items in the items[ ] array by using a loop.

Q5

Faculty
int fId String fName String designation

Faculty( )
Faculty(int fId, String fName, String designation)
void showDetails( )

Course
String courseId String courseName int credit Faculty f
Course( ) Course(String courseId, String courseName, int credit, Faculty f) void setCourseId(String courseId) void setCourseName(String courseName) void setCredit(int credit) void setFaculty(Faculty f) String getCourseId ( ) String getCourseName( ) int getCredit( ) Faculty getFaculty( )

Student
int id String name Course [ ]courses
Student( ) Student(int id, String name, int sizeOfArray) void setId(int id) void setName(String name) int getId( ) String getName( ) Course getCourse(int courseId) void addCourse(Course c) void removeCourse(Course c)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (s1) of the *Student* class by using its parameterized constructor.
- Create **two** objects (f1, f2) of the faculty class and **four** objects (c1, c2, c3, c4) of the *Course* class by using their parameterized constructor.
- Add **c1, c2** and **c3** in the courses[ ] array.
- Display the id and name of s1. Also, display the details of all the courses in the courses[ ] array by using a loop.
- From the course[ ] array, search for the c2 course and display it's FacultyName .
- Display the details of all the courses in the courses[ ] array by using a loop.
- Remove the course c1 from the array and then add the course c4.
- Display the details of all the courses in the courses[ ] array by using a loop.

Q6

Write the following classes:

Food
int foodId String foodName double price int availableQuantity
Food( ) Food(int foodId, String foodName, double price, int availableQuantity) void setFoodId(int foodId) void setFoodName(String foodName) void setPrice(double price) void setAvailableQuantity(int availableQuantity) int getFoodId( ) String getFoodName( ) double getPrice( ) int getAvailableQuantity( ) void addQuantity(int quantity) void disposeQuantity(int quantity)

Restaurant
String rName String place Food [ ]foods
Restaurant( ) Restaurant(String rName, String place, int sizeOfArray) void setRName(String rName) void setPlace(String place) String getRName( ) String getPlace( ) Food getFood(int foodId) void addFood(Food f) void removeFood(Food f)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (r1) of the *Restaurant* class by using its parameterized constructor.
- Create **four** objects (f1, f2, f3, f4) of the *Food* class by using its parameterized constructor.
- Add **f1, f2** and **f3** in the foods[ ] array.
- Display the rName. Also, display the details of all the foods in the foods[ ] array by using a loop.
- From the foods[ ] array, search for the f2 food and add 5 quantities of it.
- From the foods[ ] array, search for the f3 product and dispose 5 quantities of it.
- Display the details of all the foods in the foods[ ] array by using a loop.
- Remove the food f1 from the array and then add the food f4.
- Display the details of all the foods in the foods[ ] array by using a loop.

Q7

Write the following classes:

Contact
String mobileNumber; String personName char gender

Contact( )
Contact(String mobileNumber, String personName, char gender)
void setMobileNumber(String mobileNumber)
void setPersonName(String personName)
void setGender(char gender)
String getMobileNumber( )
String getPersonName( )
char getGender( )
String getOperatorName( ) // Checks whether the operator is GP, Robi, Banglalink or
//TeleTalk and returns the operator as a String.

PhoneBook
int phnBookId Contact [ ]contacts
PhoneBook( ) PhoneBook(String phnBookId, int sizeOfArray) void setPhnBookId(int phnBookId) int getPhnBookId( ) Contact getContact(int mobileNumber) void addContact(Contact c) void removeContact(Contact c)

Write a class **Start** that contains the main method. Inside the main method do the followings:

- Create **one** object (ph1) of the *PhoneBook* class by using its parameterized constructor.
- Create **four** objects (c1, c2, c3, c4) of the *Contact* class by using its parameterized constructor.
- Add **c1, c2** and **c3** in the contacts[ ] array.
- Display the ohnBookId. Also, display the details of all the contacts in the contacts[ ] array by using a loop.
- From the contacts[ ] array, search for the c2 contact and change its name.
- Display the details of all the contacts in the contacts[ ] array by using a loop.
- Remove the contacts c1 from the array and then add the contacts c4.
- Display the details of all the contacts in the contacts[ ] array by using a loop.

Q8

Write a Java class for a “Point of Sales (POS)” application for a Product. Name the class as **Sales**. The objective of the class is to perform the sales operation. The class has the attributes and functionality as follows:

Member Variables	**Functionality**
totalSalesAmount	The amount in Taka of the total sales
currentSalesAmount	The amount in Taka of the current sales
productCounter	The quantity of the product available
unitPrice	The unit price of the product

The class has some methods with some functionality as follows:

Member Functions	**Functionality**
Sales( )	Empty constructor
Sales(…)	Parameterized constructor
addProduct(quantity)	The *productCounter* will be increased by the value passed to it.
doSales(quantitySold)	This method performs the sales operation. This method will decrease the product counter by quantity sold, calculate the payable amount of current sales and add the current sales amount to the total sales amount.
setUnitPrice(newPrice)	Sets the unit price of the product with the given value.
showBills( )	Prints the details of the current Sales.
ShowTotalSales( )	Prints the amount of the total Sales.

Q9

a. **Toyota**, **BMW**, and **Audi** are three kinds of **Car**. You have been given the following Car and MainClass, complete other classes for different kinds of cars showing the inheritance relationship.

```
package inheritance.demo;
public class MainClass {
  public static void main(String[ ] args) {

    Car c1 = new Toyota("X-Corolla", 2006, 140.0f,"Auto");
       /*String make, String model, int year, float maxSpeed, String gearType */
    Car c2 = new BMW("X3", 2016, 200.0f,"Auto");
    Car c3 = new Audi("911", 2015, 180.0f,"Auto");

    c1.showInfo( );
    c2.showInfo( );
    c3.showInfo( );
  }

package inheritance.demo;
public class Car {
  private String carMaker;
  private String carModel;
  private int makingYear;
  private float maxSpeed;
  private String gearType;

  /*Constructor*/
  public Car( ){ }

  public Car(String make, String model, int year, float maxSpeed, String gearType){
  this.carMaker = make;
  this.carModel = model;
  this.makingYear = year;
  this.maxSpeed = maxSpeed;
  this.gearType = gearType;
  }
  /*Methods*/
  public void showInfo( ){
  System.out.println("Car Manufacturer: " + this.carMaker);
  System.out.println("Car Model: " + this.carModel);
  System.out.println("Manufacturing Year: " + this.makingYear);
  System.out.println("Gear Type: " + this.gearType);
  System.out.println("Max Speed: " + this.maxSpeed);
  }
}
```

b. Modify your classes to meet the following output.
(Hint: There are some cars have additional property, check output below -)

Run output:

```
Car Manufacturer: Toyota
Car Model: X-Corolla
Manufacturing Year: 2006
Gear Type: Auto
Max Speed: 140.0

Car Manufacturer: BMW
Car Model: X3
Manufacturing Year: 2016
Gear Type: Auto
Max Speed: 200.0
Traction Control: Auto

Car Manufacturer: Audi
Car Model: 911
Manufacturing Year: 2015
Gear Type: Auto
Max Speed: 180.0
Cruise Control: Adaptive
```

CO3	Develop solutions for real life complex problems using the concepts of Java	Psychomotor	Final Term Project
Topic 1	Projects are open to the students. They can develop any kind of Management System, Games or any kind of application that can be useful to the society.		

CO4	Explain the appropriateness of their developed solution in conjunction with concepts of OOP and Java	Affective	Final Term Project Viva
Q1	What are the swing components you used for developing this project? Why?		
Q2	Which OOP principles were followed in your project? Justify your answer.		
Q3	Who are the users of your projects?		
Q4	Can a user navigate to any window?		
Q5	How will this project contribute to the society?		
Q6	Why “Some particular” logic was implemented?		
Q7	Which EventListener interfaces did you implement in your project?		
Q8	How many abstract methods you had to override because of the EventListener interfaces? What are they?		

Chapter 4: Sample Answer and Rubric Evaluation

This chapter will enlist one sample answer and corresponding Rubric evaluation per CO. The target is to discuss this sample solution with the students to make them understand the full assessment process.

CO1	Demonstrate OOP concepts using Java	Cognitive	Mid Term Exam
Q1	a) What are the usage of the keywords: ***this, this( ), super and super( )***? Give suitable example by writing a code. b) In how many ways, a final attribute can be initialized? Give a suitable example.		

Ideal Answer:

a)

The keyword “**this**” is used to refer the current object. It is also used to differentiate between instance variable and local variables.

“this( )” is used to call the empty constructor of the same class. If any values are passed inside its parameter, then the respective parameterized constructor of the same class will be called.

The keyword “super” is used to call methods of parent class.

“super( )” is used to call the empty constructor of parent class. If any values are passed inside its parameter, then the respective parameterized constructor of parent class will be called.

```
public class Parent{
   protected int a;
   public Parent( ){   }
   public Parent( int a){this.a=a;   }
   public void show( )
   {System.out.println(“Value: ”+a);}
}
```

```
public class Child extends Parent{
   public Child( ){ this(10);  }
   public  Child( int a){super(a );   }
   public void show( )
   {super.show( ); }
}
```

b)

A final variable can be initialized in two ways, but it can only be initialed once.

i. During declaration.
ii. Inside constructor.

So, if it is initialized during declaration, it cannot be initialized inside constructor and if we want to initialize inside constructor, it cannot be initialized during declaration.

```
public class FinalDemo1{
   private final int a=10;
   public FinalDemo1 ( ){   }
}
```

```
public class FinalDemo2{
   private final int a;
   public FinalDemo2 ( ){ a=10;  }
}
```

CO Assessment Plan for Object Oriented Programming 1

Marking Criteria	Marks Distribution (Maximum 5X3=15)				Acquired Marks
	Inadequate (1-2)	Satisfactory (3)	Good (4)	Excellent (5)	
Definition	• Student vaguely define the terms or the concept.	• Definition provided with partial relevance to the subject matter.	• Correctly define the terms. May miss minor detail.	• Comprehensively defines the terms.	**5**
Logical Argument	• No logical arguments / explanation supporting the definition.	• Offers lossy related arguments.	• Strong argument / explanation offered.	• Comprehensive argument presented to clarify the concept.	**4**
Relevant Example	• No representative example. • No code snippet provided.	• Correctly identify / indicate towards real-life example. • Code snippet provided.	• Real life example is strongly connected towards the definition. • Well defined / structured code snippet provided.	• Comprehensively defend with real life example. • Well documented code snipped provided.	**4**
				Acquired Marks:	**13**
				CO Pass / Fail (70%):	**Passed**

CO Assessment Plan for Object Oriented Programming 1

• • •

Written and Verified by:

All faculties of the course for this semester must verify and sign a printed copy of the final version and the course Convener should hand it over to the designated OBE POC.

Course Name	Object Oriented Programming 1
Course ID	CSC 2209
Semester	Fall 18-19
Convener	Mohaimen-Bin-Noor

Faculty Name	Date	Signature
Mohaimen-Bin-Noor		
Dr. Kamruddin Md. Nur		
A. Z. M. Ehtesham Chowdhury		