Question 1 [20 Marks]

(a) Discuss the advantages of *Object-Oriented* languages over *Procedural* languages.

[4 Marks]

There are

- OOPs makes development and maintenance easier where as in Procedure-oriented programming language it is not easy to manage if code grows as project size grows.
- OOPs provides data hiding whereas in Procedure-oriented programming language a global data can be accessed from anywhere. This would be having issues with the security of data.
- OOPs provides ability to simulate real-world event much more effectively. We can provide the solution of real word problem if we are using the Object-Oriented Programming language.
- OOP allows the data and code to be reused

(b) *"Java is a fully Object-Oriented language."*

State whether you agree or disagree with the above statement with justifications.

[4 Marks]

Pure Object Orient Language

A language is called PURE object oriented if it contains only objects and classes. Every communication and access is performed through message passing. A pure OO language has the following characteristics.

- All predefined data types are objects
- All user defined data types are objects
- Encapsulation, Inheritance, Abstraction, Polymorphism

If it had Primitive data types and Wrapper classes then it will NOT be a pure OOP.

Fully Object Oriented Language

A language is called a FULLY object oriented language if it contains all the fundamental features of the object oriented programming and it can have primitive data types or not. It has the flowing characteristics.

- Encapsulation, Inheritance, Abstraction, Polymorphism

So java is a fully object oriented language because it has OOP related characteristics including of the wrapper classes and primitive data types.

(c) Briefly describe the following terms in Java language.

(i) Access modifiers

(ii) Package

(iii) Polymorphism

[6 Marks]

i. Access modifiers

Answer

The access modifiers in Java specifies the accessibility or scope of a field, method, constructor, or class. We can change the access level of fields, constructors, methods, and class by applying the access modifier on it.

There are four types of Java access modifiers:

1. **Private**: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
2. **Default**: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
3. **Protected**: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
4. **Public**: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

ii) Package

Answer

- A **java package** is a group of similar types of classes, interfaces and sub-packages.
- Package in java can be categorized in two form, built-in package and user-defined package.
- There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.
- **Packages are used to minimize class conflicts and group classes based on their similar characteristics.**
- Package statement is the first statement that is appearing in the class and the absence of the package statement puts the class into a default package.

- Packages can be nested.

iii) Polymorphism

- Polymorphism is the ability of an object to take on many forms.
- The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
- Any Java object that can pass more than one IS-A test is considered to be polymorphic.
- In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.
- It is important to know that the only possible way to access an object is through a reference variable. A reference variable can be of only one type. Once declared, the type of a reference variable cannot be changed.
- The reference variable can be reassigned to other objects provided that it is not declared final. The type of the reference variable would determine the methods that it can invoke on the object.
- A reference variable can refer to any object of its declared type or any subtype of its declared type. A reference variable can be declared as a class or interface type.
- Now, the Deer class is considered to be polymorphic since this has multiple inheritance. Following are true for the above examples –

    - A Deer IS-A Animal
    - A Deer IS-A Vegetarian
    - A Deer IS-A Deer
    - A Deer IS-A Object

When we apply the reference variable facts to a Deer object reference, the following declarations are legal.

(d) How is a *Constructor* different from a *Method*? Explain your answer with suitable code examples.

[6 Marks]

- Constructor do not have a return type
- Constructor has the same name as the class
    - Constructor must be having the same name as the class
    - Methods may or may not have the same name as the class
- Constructor is automatically created when an object is being created
- Absence of a user defined constructor will create a default constructor in the class

Question 2 [20 Marks]

(a) What is the difference between *this* and *super* keywords in Java language?

[6 Marks]

- **super** keyword is used to access methods and member variables of the parent class instance while **this** is used to access methods and the member variable of the current class instance

Following are the notable differences between super() and this() methods in Java.

	super()	**this()**
Definition	super() - refers immediate parent class instance.	this() - refers current class instance.
Invoke	Can be used to invoke immediate parent class method.	Can be used to invoke current class method.
Constructor	super() acts as immediate parent class constructor and should be first line in child class constructor.	this() acts as current class constructor and can be used in parametrized constructors.
Override	When invoking a superclass version of an overridden method the super keyword is used.	When invoking a current version of an overridden method the this keyword is used.

(b) Is it possible to use both *this* and *super* to make constructor calls? Justify your answer.

[4 Marks]

Yes

1. super keyword can be used to call the constructor of the super class (constructor chaining into the super class)
2. this keyword can be used to call the constructor of the same class (constructor chaining in the same class)

```
class Animal {
   String name;
   Animal(String name) {
      this.name = name;
   }
   public void move() {
      System.out.println("Animals can move");
   }
   public void show() {
      System.out.println(name);
   }
}
class Dog extends Animal {
   Dog() {
      //Using this to call current class constructor
      this("Test");
   }
   Dog(String name) {
      //Using super to invoke parent constructor
      super(name);
   }
   public void move() {
      // invokes the super class method
      super.move();
      System.out.println("Dogs can walk and run");
   }
}
public class Tester {
   public static void main(String args[]) {
      // Animal reference but Dog object
      Animal b = new Dog("Tiger");
      b.show();
      // runs the method in Dog class
      b.move();
   }
}
```

(c) Consider the following Java program.

```
class A{
    int a = 30;
}
class B extends A{
    int b = 30;

    void display(){
        System.out.println(super.a+b);
    }

    public static void main(String args[]){
        B obj= new B();
        obj.display();
    }
}
```

What will be the output of the above program? Justify your answer.

[4 Marks]

Output: 60

super.a is referring to a variable in class A which is the super class of B of which has value 30. Once super.a and b are added it will be 60.

(d) (i) Briefly explain the usage of *static* keyword in Java language.

[2 Marks]

- static keyword specifies that the item specified belongs to the class but not to the instance
- can be used for variables, methods, blocks and nested classes
- Most of the times used for declaring constants
- Always main method of a class is static: reason is main method should be callable when there are no objects being created.

(ii) What is the output of the following Java program?

```
class Test1 {
public static void main(String[] args)
    {
        int x = 20;
        System.out.println(x);
    }
    static
    {
        int x = 10;
        System.out.print(x + " ");
    }
}
```

[4 Marks]

Output: 10 20

In a class if there is a static block it will be executed

They are executed even before the main method

Therefore the static block which is printing value 10 will be first executed and then value 20 will be executed.

Question 03 [20 Marks]

Use the following Java code to answer the questions (a), (b), (c), and (d).

```
public class Test{
    public static void main(String args[]){
        Animal animal=new Carnivore();
        animal.getTheWayOfEat();
    }
}

abstract class Animal{
    public abstract void getTheWayOfEat();
}

class Carnivore extends Animal{
    @Override
    public void getTheWayOfEat(){
        System.out.println("Eat meats");
    }
}

class Herbivore extends Animal{
    @Override
    public void getTheWayOfEat(){
        System.out.println("Eat plants");
    }
}
```

(a) What is the output of the above Java program?

[3 Marks]

Output: Eat Meats

(b) Write a class named ***Felidae*** by extending the ***Carnivore*** class in the above code. Then override the method ***getTheWayOfEat()*** inside the ***Felidae*** class by providing a functionality to print "*Must consume meats to survive.*"

[5 Marks]

```
class Felidae extends Carnivore{
    @Override public void getTheWayOfEat ()
            {
                System.out.println("Must Consume Meats to Survive");
            }
}
```

(c) After completing the part (b), the following code segment is added to the main method.

```
Felidae felidae = new Carnivore();
felidae.getTheWayOfEat()
```

Identify the possible output after the insertion of the above code segment to the main method. If an error has occurred, briefly explain the necessary actions to resolve the error.

[6 Marks]

Error occurs

```
Carnivore carnivore = new Felidae();
carnivore.getTheWayOfEat();
```

(d) Change the access modifier of ***getTheWayOfEat()*** method in the ***Animal*** class to *protected*. Now determine the possible output of the above program by providing justifications.

[6 Marks]

Answer

Applying protected access modifier to getTheWayOfEat() method will not do any change.

Reason: protected access modifier allows the sub classes to inherit super class members and member functions which are specified as protected.

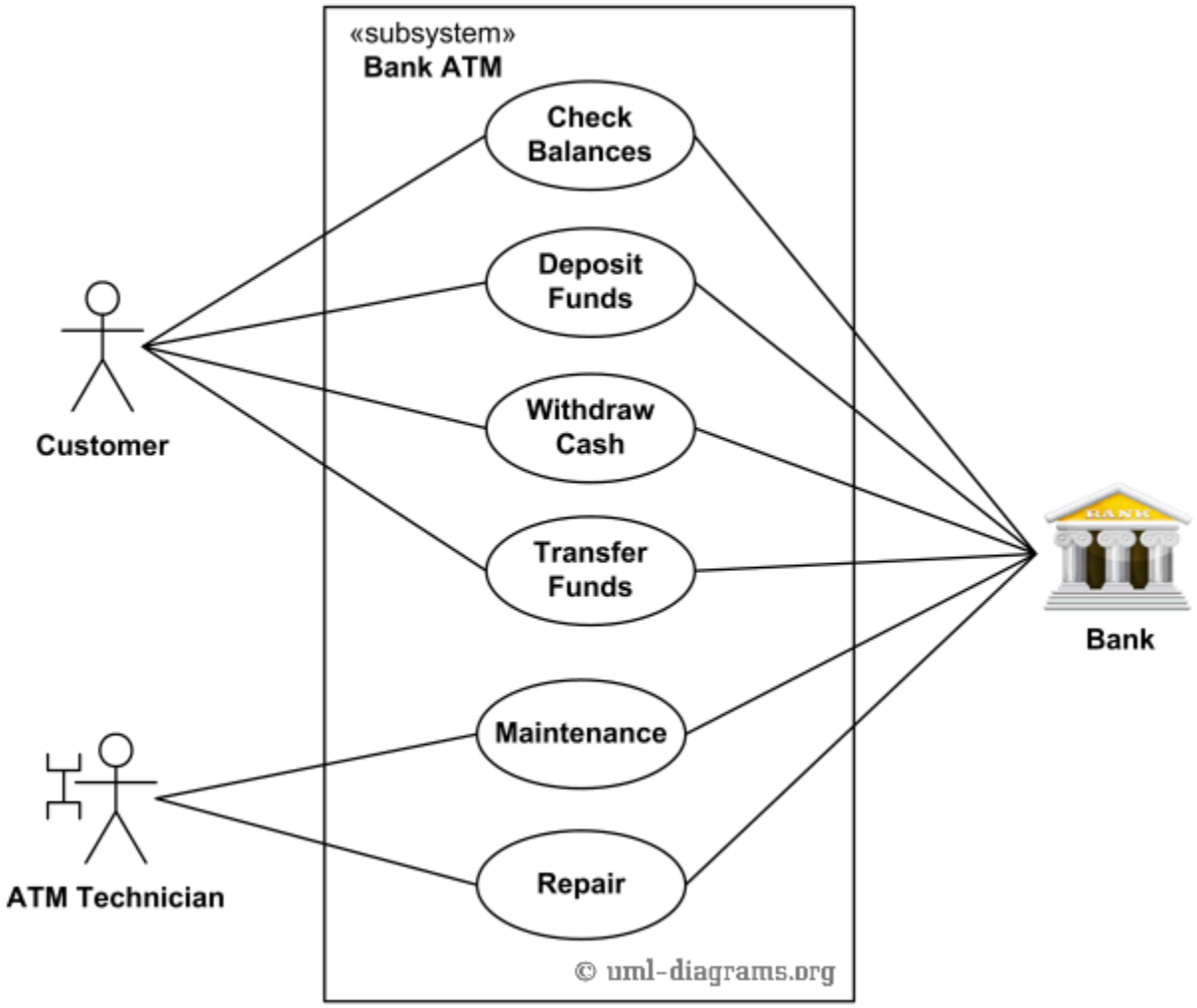
Question 4 [20 Marks]

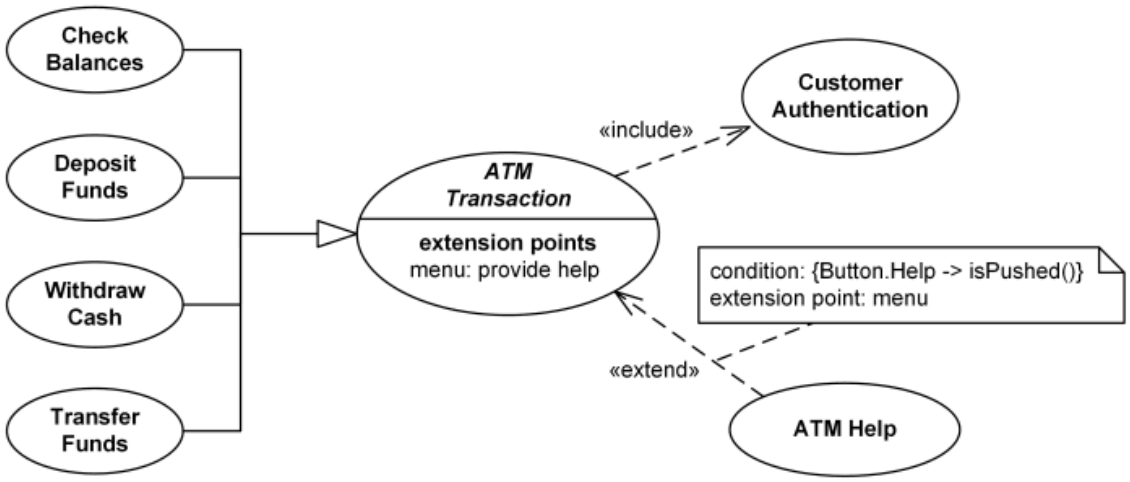
Consider the following requirements for an Automatic Teller Machine (ATM) of a bank.

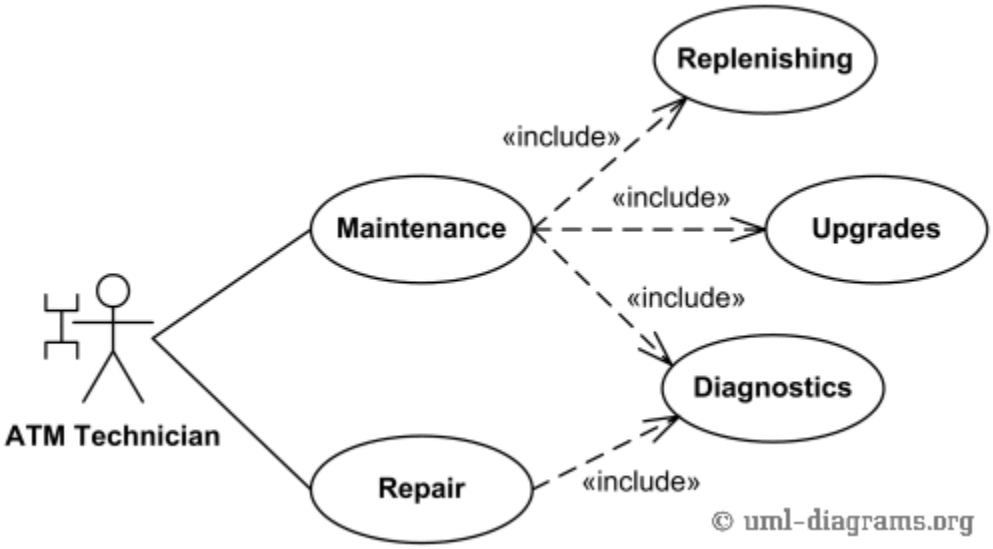
A customer uses a bank ATM to check the balances of his/her bank accounts, deposit funds, withdraw cash, and transfer funds. The ATM Technician provides services such as maintenance and repairs. All these activities involved with the bank will be either related to customer transactions or ATM servicing. The customer is authenticated by inserting an ATM card and entering a personal identification number (PIN). A customer authentication use case is required for every ATM transaction.

(a) Draw the use case diagram for the scenario given above.

[6 Marks]







(c) Describe the difference between the synchronous and asynchronous messages in sequence diagrams.

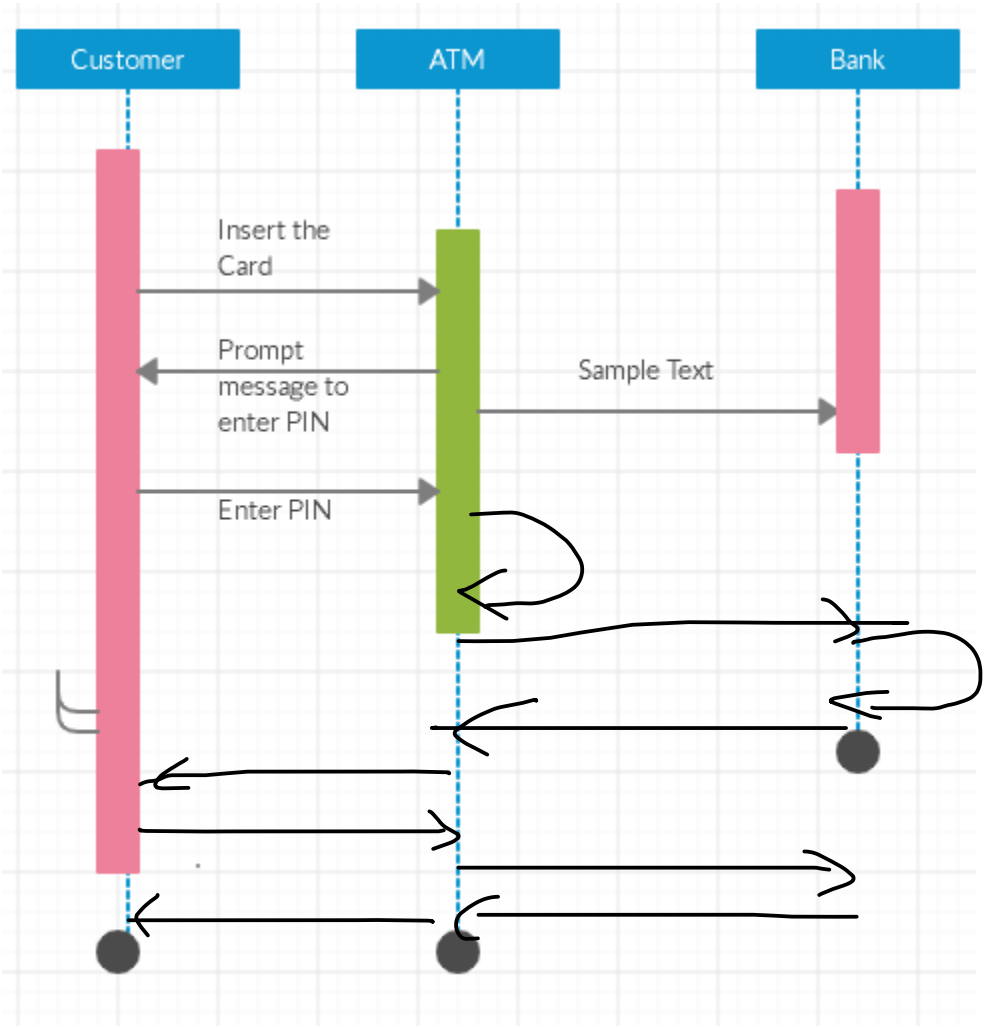
[4 Marks]

- Synchronous messages once sent from an object the object will hold itself until the reply arrives from the other end (another object)
- Asynchronous messages once sent, the sending object will continue to proceed with its code and the reply will come an any point which will not influence the normal execution process of the sender.

(d) A customer wants to withdraw money from his/her bank account. The customer enters the card into an ATM. The ATM prompts *"Enter PIN"* message. The customer enters the PIN. The ATM (internally) retrieves the bank account number from the card. The ATM encrypts the PIN and the account number and sends it over to the bank. The bank verifies the encrypted account number and the PIN. If the PIN is correct, the ATM displays *"Enter amount"* message. The customer enters the amount to withdraw. Then the customer withdraws money from the bank account.

Draw the sequence diagram for the above scenario.

[10 Marks]



Question 5 [20 Marks]

(a) Briefly explain two (2) advantages of using Unified Modeling Language (UML) in software modeling.

[4 Marks]

(b) Imagine that you are designing a word processing software (like MS Word or Open Office Writer). The document is the main item in a word processing software. For typing text into a document, you use the keyboard. Every document consists of several pages. Every page may have a header and a footer. Finally, a document (including the header and footer) could contain text, numbers, special characters, tables, and pictures.

i. Identify the classes of the above scenario.

[4 Marks]

ii. Identify the relationships among them.

[4 Marks]

iii Model the UML class diagram for the above scenario.
(Note: You may use your own attributes and methods. State any assumptions you made)

[8 Marks]

i. Identify the classes of the above scenario.

Classes

- Document
- Page
- Header
- Footer
- Content

ii. Identify the relationships among them.

Document has One or More Pages - Composition
Page has One Header - Aggregation
Page has One Footer - Aggregation
Page has Content - Aggregation