University of Sargodha

M.Sc. 2nd Term Examination 2018

Subject: Information Technology **Paper: Object Oriented Programming (CMP-2123)**

Time Allowed: 2:30 Hours **Maximum Marks: 60**

Note: Objective part is compulsory. Attempt any four questions from subjective part.

Objective Part (Compulsory)

Q1. Write short answers of the following in 2-3 lines each on your answer sheet. (12*2)

i. What is a static method? Give some examples from any Java class?
ii. What is the purpose of Information Hiding?
iii. Can a constructor call another constructor using this keyword? Give example.
iv. What is void reference?
v. Can a catch() block be written without try() block?
vi. Can we instantiate an interface?
vii. When swing package is used?
viii. What is the purpose of instanceof operator?
ix. What is this reference and its purpose. Give an example?
x. Can we override final methods if not why?
xi. What is the purpose of Wrapper class?
xii. Differentiate between throw and throws?

Subjective Part (4*9)

Q2. Create a class **employee** (id, name). **Inherit** two classes named **collegeEmployee** (salary) and **Faculty** (contract_duration) from it. Write appropriate constructors, print() and get() functions for each class. Write an application that demonstrates using objects of each class.

Q3. Write down a class **Complex** having data members (real, imaginary) and implements Serializable. The program should also contain the constructor and method to read a complex number from user. The program should throw exception if imaginary is negative. Also override toString() method.

Q4. Create an abstract class **Furniture** (color, price). Include get and set methods for these fields; the setPrice() methods is abstract. Create two subclasses **Table** (length, width) and Chair (color) from furniture and include appropriate setPrice() methods in each class. Finally, write an application that uses Furniture class and subclasses to display information about instances.

Q5. Create an interface **Playable** having methods play(). Create two classes **CD**(title, playtime) and **DVD**(title, playtime) that implement Playable. Give appropriate constructors. Also implement method play() that prints the title and playtime. Demonstrate polymorphic behavior of method play using interface **Playable**.

Q6. Write an application that calculate grade of student based on marks entered. The integer number should be entered from the keyboard (via a JTextField). An un-editable Textfield should be used to display grade against entered marks when user clicks on calculate grade button. Use the following formula for the calculation.
Hint: If (marks >= 80) grade ='A', If (marks >=70 && marks<80) grade ='B', If (marks >=60 && marks<70) grade ='C', If (marks >=50 && marks<60) grade ='C', If (marks < 50) grade = 'F'

Q7. Write a program that concatenates the contents of file2 into contents of file1 i.e. it should append the contents of a text file at the end of another file.

University of Sargodha

MSc. 2nd Term Examination 2018.

Subject I.T   Paper: Object Oriented Programming (CMP-2123)

Note: Objective part is compulsory. Attempt any four questions from subjective part.

Time Allowed: 2.30 Hours   **Maximum Marks: 60**

Objective Part (Compulsory)

Q.01: write short answer of the following in 2-3 lines on your answer sheet.

1. **What are static methods? Give some examples from any java class.**
   Answer: A static method is a class method that exists whether or not any objects exist. A static method, indicated by the keyword static placed before the method's return type, belongs to the defining class. A static method may be called whether or not an object of the class exists. A static method exists apart from any objects. The methods **Math.random(), Math.sqrt(),** and **Math.abs()** are all static methods.
2. **What is purpose of information hiding?**
   Answer: Information hiding is one of the most important principles of OOP inspired from real life which says that all information should not be accessible to all persons. Private information should only be accessible to its owner. Information hiding is a security barrier against unauthorized access to any object or information.
3. **Can a constructor call other constructors using this keyword? Give example**
   Answer: Yes, In OOP a constructor can call other constructors using this keyword.
   Example:
```
public Employee(int id, String name)
{
        this.id = id;
        this.name = name;
}
//default constructor
public Employee(){
        /* calling parameterized constructor of same (Employee) class by using
        keyword this */
        this (10, "not set");
}
```
4. **What is void reference?**
   Answer: A reference with value null refers to no object and holds no address; it is called a void reference.

5. **Can a catch block be written without a try block?**
   Answer: No, a catch block can't be written without a try block. A catch block is always written after a try block because a catch block is used to catch and handle an exception thrown

by code in try block.

6. **Can we instantiate an interface?**
   Answer: No we can't instantiate an instance of interface because an interface is equivalent to an abstract class. All methods of an interface are abstract by default that is, there are no implementations of any method at all. That's why we cannot instantiate an interface.
7. **When swing package is used?**
   Answer: Swing Package is used for Graphical User Interface(GUI) based programing. The Swing library paints the components on the screen so that the look and feel of a graphical user interface is consistent from platform to platform.
8. **What is the purpose of instanceof operator?**
   Answer: The instanceof operator can help a programmer to avoid casting errors. instanceof is a Boolean operator that requires two operands object and className. If object belongs to or is derived from class, then instanceof returns true, otherwise instanceof returns false.
9. **What is this reference and its purpose? Give Example**
   Answer: this reference is used to point the current object in java whose method or constructor is being invoked. Moreover it is used to refer any member of the current object.
10. **Can we override final function? If not why?**
    Answer: No we cannot override a final function in java. A final keyword is used by the programmer to fix the behavior if a method. That's why he declares a final method to prevent subclasses to override it.
11. **What is the purpose of Wrapper class?**
    Answer: Wrapper classes are used to wrap a data type into an object. Sometime a primitive data type is intended to behave like an object. Then wrapper classes help the programmer to achieve his intended behavior.
12. **Differentiate between throw and throws.**
    Answer: throw keyword is used to throw an exception from a method explicitly whereas a throws keyword is used in method declaration to mention which exception this method can throw.

Subjective Part (9*4)

Question 02: Create a class **Employee (id, name).** Inherit two classes named **CollegeEmployee(Salary)** and **Faculty(Contract_Duration).** Write appropriate constructors, print() and get () functions for each class. Write an application that demonstrates using objects of each class.

ANSWER:

```
public class  Employee{

    protected int Id;
    protected String Name;
    public Employee(){
            this(0, "Not Set")
    }
    public Employee(int i, String n){
            Id = i;
```

```
                Name = n;
        }
        public void setId(int i){
                Id= i;
        }
        public void setName(String n){
                Name = n;
        }
        public int getId(){
                return Id;
        }
         public String getName(){
            return Name;
        }
        public void Print(){
            System.out.println("Id : "+ Id);

            System.out.println("Name : "+ Name);
        }

}
```

public class CollegeEmployee extends Employee{

```
    private int Salary;
    public CollegeEmployee()
    {
            super();
            salary = 0;
    }
    public CollegeEmployee( int i, String n, int s)
    {
            super(i,n);
            salary = s;
    }
    public void setSalary(int s){
          Salary = s;
    }
    public int getSalary(){
    return Salary;
    }
      public void Print(){
          super.Print();

          System.out.println("Salary : "+ Saalry);
      }
```

```
}

public class Faculty extends Employee{
    private int Contract_Duration;
    public Faculty ()
    {
            super();
            Contract_Duration = 0;
    }
    public Faculty ( int i, String n, int s)
    {
            super(i,n);
            Contract_Duration = s;
    }
    public void setContactDuration(int s){
          Contract_Duration = s;
    }
    public int getContactDuration (){
    return Contract_Duration;
    }
      public void Print(){
          super.Print();

          System.out.println("Contract Duration: "+ Contract_Duration);
      }

}

public class test{
    public static void main(String[] args){
        CollegeEmployee obj1 = new CollegeEmployee(1, "Nasir Ali" , 60000 );
        Faculty obj2  = new Faculty( 2 , "Shehzar Ali" , 3);
        obj1.Print();
        obj2.Print();
    }

}
```

Question 03: Write down a class **complex** having data members (real , imaginary) and implement serializeable. The class also should contain the constructor and method to read a complex number from user. The program should throw an exception if imaginary is negative. Also override toString() Method.

Answer:

```
import java.lang.util.*;
import java.io.Serializable;


public class Complex implements  Serializable {

private double real;

private double imaginary;

public Complex(double r , double i)

{

try{

        real = r;

if(i<0){

        throw new ArithmeticException();

}

        imaginary = i;

}

catch(ArithmeticException ex)

{

        System.out.println(“Imaginary Part Can’t be Negative”)

}

}

public void getComplexNo(){

Scanner S = new Scanner(System.in);

try{

System.out.println(“Please Enter Real part of Complex No : ”);
real = S.nextDouble();

System.out.println("Please Enter Imaginary part of Complex No : ");

double i  = S.nextDouble();

if(i<0){

        throw new ArithmeticException();

}

} catch(ArithmeticException ex)

{

        System.out.println("Imaginary Part Can't be Negative")

}

}

public String toString(){

return "real : "+real+ "Imaginary : " +  imaginary;

}

}
```

Question 04: Create an abstract class **Furniture(color, price).** Include get and set methods form these fields. The setPrice Method is abstract. Create two subclasses Table() and Chair() from Furniture and include appropriate setPrice method in these classes. Finally write an application that that uses Furniture class and sub classes to display information about instances.

ANSWER:

```
public abstract class Furniture{

protected String color;

protected int price;

public Furniture(){

        this ("No color" , 0);
}
public Furniture(String c, int p){
        color = c;
        price = p;
}
public void setColor(int c){
color = c;
}
public abstract void setPrice(int p);
public int getPrice(){
return price;
}
public String getColor(){
return color;
}
public int getPrice(){
return price;
}
}
```

}

public class Table extends Furniture{

```
public void setPrice(int p)
{
        Price = p;
}

public void Print(){
    System.out.println("Table Color : "+ color);

    System.out.println("Table Price : "+ price);

}

}
```

public class Chair extends Furniture{

```
public void setPrice(int p)

{

        Price = p;

}

public void Print(){
    System.out.println("Chair Color : "+ color);

    System.out.println("Chair Price : "+ price);

}

 }
```

Question 05: Create an interface **Playable** having method play()**.** Create two classes CD(title,playtime) and DVD(title , playtime) that implement playable. Give appropriate constructors. Also implement method play() that prints title and playtime. Demonstrate polymorphic behavior of method play using interface Playable.

ANSWER:

public interface Playable{

```
        public void play();

}
```

public class CD implements Playable{

```
private string title;
private int  playtime;



public CD(){

this(“Not Set” , 0);

}

public CD(String t , int p){

title = t;

playtime = p;

}

public void play(){
      System.out.println(“I am Play method in CD Class“);
      System.out.println(“CD Title : ”+ title);

      System.out.println(“CD Playtime : ”+ playtime+” Minutes”););

}

}
```

public class DVD implements Playable{

```
private string title;

private int  playtime;

public DVD(){

this(“Not Set” , 0);

}

public DVD(String t , int p){

title = t;

playtime = p;
}

public void play(){
      System.out.println(“I am Play method in DVD Class“);
      System.out.println(“DVD Title : ”+ title);

      System.out.println(“DVD Playtime : ”+ playtime+” Minutes”);

}

}
```

public class test{

```
public static void main(String[] args){

Playable [] arr = new Playable[5];

arr[0] = new CD(“CD 1” , 100);

arr[0] = new DVD(“DVD 1” , 150);

arr[0] = new CD(“CD 1” , 100);

arr[0] = new CD(“CD 1” , 100);

arr[0] = new DVD(“DVD 1” , 125);

for(int i = 0; i<arr.length();i++){

arr[i].Play();

}

}

}
```

University of Sargodha

M. Sc. 2nd Term Examination 2017

Subject: Information Technology **Paper: Object Oriented Programing (CMP-2123)**

Time Allowed: 2.30 Hours **Maximum Marks: 60**

Note: Objective part is compulsory. Attempt any Four questions from subjective part.

Objective Part **(Compulsory)**

Q. No.1. Write short answers of the following in 2-3 lines on your answer sheet. (2*12)

i. Differentiate class and an object?
ii. What is JVM?
iii. What is method overriding in Java?
iv. Differentiate an Interface and abstract class?
v. What is Abstraction?
vi. What are packages?
vii. Differences between "this" and "super" keywords?
viii. Can java support multiple inheritance? Justify your answer?
ix. What is runtime polymorphism or dynamic method dispatch?
x. Why StringBuffer is called mutable?
xi. What do you mean by Checked Exceptions?
xii. Difference between "throw" and "throws" kewords in java?

Subjective Part (4*9=36)

Q.No.2 Create a class **Employee** with instance variable EmpId, EmpName, Empage and decide proper data types and access modifiers for these variables. Define overloaded constructors, getter/setter methods and also **override** to String() method. Demonstrate this class in your program.

Q.No.3 Create a class **called 'SHAPE'** (should be made an **Abstract class)** having two instance variable **Length** (double), **Height** (double) with appropriate access specifiers and instance method **Get Data()** to get and assign the values (to length & Height) from the user, it also has an abstract method **Display Area( )** to compute and display the area of the geometrical object. **Derive two** specific classes **'TRIANGLE'** and **'RECTANGLE'** from the **base** class that override the base class method. Using these **three classes** design a program that will accept dimension of a **triangle / rectangle** interactively and display the area.

Hint:

Area of Triangle = ½ (L × h)
Area of Rectangle = L × h

Q.No.4 Explain **exception handling** in java and why we use nested try blocks. Write a program to demonstrate exception handling by nested try/catch blocks.

Q.No.5 Develop a program to read a text file. Process the read text by using String class methods in the following order:

i. Convert all text into lower case (Hint use String's to lower method)
ii. Count no. of words in the text (Hint use String' split method)

Q.No.6 Write a temperature-conversion application that converts from Fahrenheit to Celsius. The Fahrenheit temperature should be entered from the keyboard (via a JTextField). An un-editable Textfield should be used to display the converted temperature when user clicks on conversion button. Use the following formula for the conversion:

***Celsius* = 5/9× (Fahrenheit – 32)**

Q.No.7 Explain the advantage of using **interfaces** in java? Also write a java program to show how a class implements two interfaces.

University of Sargodha

MSc.IT 2nd Term Examination 2017

Subject I.T Paper: Object Oriented Programming (CMP-2123)

Note: Objective part is compulsory. Attempt any four questions from subjective part.

Time Allowed: 2.30 Hours **Maximum Marks: 60**

Objective Part (Compulsory)

Q.01: write short answer of the following in 2-3 lines on your answer sheet.

1. **Differentiate class and an object?**

Answer:

Class: A class is a template, or blueprint, from which objects are created. A class includes both data members as well as functions to manipulate that data.

Object: An object is a representation or an abstraction of some entity such as a car, a soda machine, an ATM machine etc.

Class is a sketch or blueprint of a real world entity whereas an object is an instantiation of class.

2. **What is JVM?**

Answer: JVM is an engine that provides runtime environment to drive the Java Code or applications. It converts Java byte code into machines language.

3. **What is method overriding in Java?**

Answer: Overriding is a feature of OOP that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent classes. When a method in a subclass has the same name, same parameters or signature and same return type (or sub-type) as a method in its super-class, then the method in the subclass is said to *override* the method in the super-class.

4. **Differentiate an Interface and abstract class?**

Answer:

Interface	**Abstract class**
Interface support multiple implementations.	Abstract class does not support multiple inheritance.
Interface does not contain Data Member.	Abstract class contains Data Member

Interface	**Abstract class**
Interface does not contain Constructors.	Abstract class contains Constructors
An interface Contains only signature of member functions.	An abstract class Contains both signatures of methods (abstract) and definition of complete member function.
An interface cannot have access modifiers by default everything is assumed as public	An abstract class can contain access modifiers for the subs, functions, properties
Member of interface cannot be Static	Complete Member of abstract class can be Static

5. **What is Abstraction?**

Answer: Abstraction is a feature of OOP that states that capture only those details about an object that are relevant to current perspective. Abstraction is a way to cope with complexity and it is used to simplify things.

6. **What are packages?**

Answer: Package in Java is a mechanism to encapsulate a group of classes, sub packages and interfaces. A package is used to group related classes. It behaves as **a folder in a file directory**. Packages are used to avoid name conflicts, and to write a better maintainable code.

7. **Difference between "this" and "super" keywords?**

Answer: super() as well as this() are special keywords that are used to call **constructor**. super() is used to call constructor of parent class while this() is used to call **current** class's constructor.

8. **Can java support multiple inheritances? Justify your answer?**

Answer: No, Java doesn't support multiple inheritance. Java discourages multiple inheritance to **avoid the ambiguity** caused by it. One of the examples of such problem is the **diamond problem** that occurs in multiple inheritance.

9. **What is run time polymorphism or dynamic method dispatch?**

Answer: Dynamic Method Dispatch is the process to select a particular implementation of a polymorphic method or function to call at run time.

10. **Why StringBuffer is called mutable?**

Answer: StringBuffer is mutable means one can change the value of the object . StringBuffer and is called mutable because whenever we perform a modification on their objects their state gets changed.

11. **What do you mean by Checked Exceptions?**

Answer: A checked exception is a type of exception that must be either caught or declared in the method in which it is thrown. For example, the java.io.IOException is a checked exception.

12. **Difference between "throw" and "throws" keywords in java?**

Answer: throw keyword is used to throw an exception from a method explicitly whereas a throws keyword is used in method declaration to mention which exception this method can throw.

Subjective Part (9*4)

Q.no.2: Create a class **Employee** with instance variable EmpId, EmpName, EmpAge and decide proper data types and access modifiers for these variables. Define overloaded constructors, getter/setter methods and also override toString () method. Demonstrate this class in your program.

Answer:

```
package myProject;


public class Employee {

	private int EmpId;

	private String EmpName;

	private int EmpAge;


	public Employee() {

		this(0,“Not Set”,0);

	}

	public Employee(int i, String n, int a ) {

		EmpId = i;

		EmpName = n;

		EmpAge = a;

	}

	public void setId(int i) {

		EmpId = i;

	}

	public void setName(String n) {

		EmpName = n;
	}public void setAge(int a) {

		EmpAge = a;

	}

	public int getId() {

		return EmpId;

	}

	public String getName() {

		return EmpName;

	}

	public int getAge() {

		return EmpAge;

	}

	public void Display(){

		System.out.println("Employee Id : "+EmpId);

		System.out.println("Employee Name : "+EmpName);

		System.out.println("Employee Name : "+EmpAge);

	}

	public String toString(){

		return new String("Employee Id : "+EmpId+"Employee Name :
"+EmpName+"Employee Name : "+EmpAge);

	}

}
```

Q.no.3: Create a class called ‘SHAPE’ (should be made an abstract class) having two instance variable Length (double),Height (double) with appropriate access specifiers and instance method Get Data() to get and assign the values(to length &Height) from the user, it also has an abstract method Display Area() to compute and display the area of the geometrical object. Derive two specific classes

‘TRIANGLE’ and ‘RECTANGLE’ from the base class that override the base class method. Using these three classes design a program that will accept dimension of a triangle/rectangle interactively and display the area.

Hint:

Area of Triangle=1/2(L*h)

Area of Rectangle=L*h

Answer:

Shape Class:

```
import java.util.*;

public abstract class Shape {

	protected double Length;

	protected double Height;


	public void getData() {

	Scanner S = new Scanner(System.in);

	System.out.println(“Please Enter Length : “);

	this.Length = S.nextDouble();

	System.out.println(“Please Enter Height : “);

	this.Height = S.nextDouble();

	}

public abstract void DiplayArea();

}
```

Triangle Class:

```
public class Triangle extends Shape {

        public Triangle() {

                System.out.println(“Enter Data for Triangle : “);

                this.getData();

        }

        public void DiplayArea() {

                double area = 0.5*this.Length * this.Height;

                System.out.println(“Area of Trinage : “+ area);

        }

}
```

Rectangle Class:

```
public class Rectangle extends Shape {

        public Rectangle() {

                System.out.println(“Enter Data for Rectangle : “);

                this.getData();

        }

        public void DiplayArea() {

                double area = this.Length * this.Height;

                System.out.println(“Area of Rectangle : “+ area);

        }

}
```

Test Class:

```
public class TestProgram {
        public static void main(String[] args) {

               Triangle t = new Triangle();

               t.DiplayArea();


               Rectangle r = new Rectangle();

               r.DiplayArea();

        }

}
```

Q.NO.7: Explain the advantages of using interfaces in java program to show how a class implements two interfaces.

Answer:

Interface:

An *interface* is a named collection of static constants and abstract methods. An interface specifies certain actions or behaviors of a class but not their implementations.

Interfaces are special java type which contains only a set of method prototypes and static Constants, but does not provide the implementation for these prototypes. All the methods inside an interface are abstract by default thus an interface is tantamount to a pure abstract class – a class with zero implementation. Interface can also contain static final constants.

Advantages of Interface:

Interfaces provide many advantages to the Java programmer. One is that they allow standard sets of methods to be used across the class hierarchy. For example, you can define the Editable interface to support cut, copy, and paste operations. The Editable interface can then be implemented by relevant classes and establish a uniform approach to implementing these common operations.

Interface types allow objects to be referenced by the methods they support without considering their location in the class hierarchy. They make maximal use of dynamic binding, allowing objects to be accessed independently of their implementation details. For example, parameters can be defined as interface types and used by methods. These methods can invoke the interface methods of their arguments without having to determine the classes to which the arguments belong.

Interfaces also support selective multiple inheritance. They allow various subsets of the features supported by different classes to be shared without mandating that all features of these classes be uniformly imposed as the result of inheritance.

Finally, because interfaces are declared independently of classes, they are unaffected by changes to specific classes or to the class hierarchy as a whole.

Program to Demonstrate two Interfaces:

```
public interface GetData {

	public void getData();

}


public interface PrintArea {

	public void printArea();

}


public class Square implements  GetData , PrintArea {

	public void printArea() {

		//Write Some Code to calculate and Area of Square



	}

	public void getData() {

		//Write Some to code to get Data for Shape.

	}

}
```

Q.no.5: Develop a program to read a text file. Process the read text by using String class methods in the following order:

1. Convert all text into lower case (Hint use String's to lower method)

2. Count no. of words in the text (Hint use String's split method)

Answer:

```
import java.io.*;

import java.util.*;

class Converter{


public static void main (String[] args) {

       String permFile = “D:\\DestFile.txt”;

    String tmpFile =  “D:\\SourceFile.txt”;

    int count = 0;


    try {


            BufferedWriter out = new BufferedWriter(new FileWriter(permFile, true));

            BufferedReader in = new BufferedReader(new FileReader(tmpFile));

            String str;

            while ((str = in.readLine()) != null) {

                   String[] sp = str.split( ” “ );

                   count += sp.length;

                  out.write(str.toUpperCase());

                  out.write(“\n”);

                  }
```

```
                in.close();

                out.close();

                System.out.println(“word Count : “+count);

                } catch (IOException e) {

                      System.out.println(“Source File dont Exist”);

                }

    }

}
```

University of Sargodha

M. Sc. 2nd Term Exam 2016.

Subject: I. T **Paper: Object Oriented Programming (CMP: 2123)**

Time Allowed: 2:30 Hour **Maximum Marks: 60**

Note: Objective part is compulsory. Attempt any four questions from subjective part.

Objective Part (Compulsory)

Q.1. Write short answers of the following in 2-3 lines each. (2*12)

i. Define a class and an object?
ii. What are two main components of a class?
iii. Can we create a String object without "new "operator? If so, give an example?
iv. What is difference between **next ()** and **nextLine ()** methods?
v. Differentiate the access modifiers **public** and **protected**.
vi. For what purpose the keyword "**this**" is used?
vii. Differentiate **abstract** classes and **concrete** classes?
viii. What is **Overloading** of a method? Also give an example.
ix. Define **polymorphism**?
x. What is has-A relationship in classes?
xi. What is an **Interface**?
xii. What is **dynamicdispatch?**

Subjective Part (4*9)

Q.2. Create a **Distance** class with instance variable feet and inches. Write suitable parameterized constructor, getter and setter methods and a method to add two distance objects in such a way that if resultant inches exceed 12, feet should be incremented by 1 and inches decremented by 12 by using the statement **dist3.add (dist1, dist2);** where dist1, dist2 and dist3 are objects of **Distance** class.

Q.3. Writ a program that creates a **Parent** class and **Child** class that inherits the Parent class. Demonstrate **Upcasting** by using Child class.

Q.4. Write a program that has **Shape** class and sub classes as **Square** and **Circle**, Use these classes to **demonstrate Polymorphism**.

Q.5. What do you mean by **exception handling**? What are the various keywords used in exception handling and also highlight their usage.

Q.6. Write a program that demonstrate the function of **Object** class acting as a parent class of all classes created in java.

Q.7. Write a java text stream class from java.io package to read text from one file and write to another text file.

University of Sargodha

MSc.IT 2nd Term Examination 2016.

<u>Subject I.T</u> <u>Paper: Object Oriented Programming (CMP-2123)</u>

Note: Objective part is compulsory. Attempt any four questions from subjective part.

Time Allowed: 2.30 Hours **Maximum Marks: 60**

Objective Part (Compulsory)

Q.01: write short answer of the following in 2-3 lines on your answer sheet.

1. **Define a class and an object?**
   Answer:
   Class: A class is a template, or blueprint, from which objects are created. A class includes both data members as well as functions to manipulate that data.
   Object: An object is a representation or an abstraction of some entity such as a car, a soda machine, an ATM machine etc.
2. **What are two main components of a class?**
   Answer: There are two main components of a class.
   1. **Data Members:** Data member are the properties of an object.
   2. **Member Functions:** The Methods that are used or manipulate the data.
3. **Can we create a String object without "new operator" ? If so, give an example?**
   Answer: Yes, We can create a string object without using **new** operator, by initializing object with a string value at the time of declaration.
   Example:
   String myString = "It is a String."
   Using above statement,. We can create a String object without using new Operator.
4. **What is the difference between next() and nextLine() methods?**
   Answer: next() can read the input only till the space. It can't read two words separated by space. Also, next() places the cursor in the same line after reading the input. **nextLine**() reads input including space between the words. It reads till the end of line n. Once the input is read, nextLine() positions the cursor in the next line.
5. **Differentiate the access modifiers public and protected.**
   Answer: A **public** member is one that is accessible to all classes in the project. A **protected** member is one that is accessible within all classes in the same package *and* within subclasses in other packages.
6. **For what purpose the keyword "this" is used?**
   Answer: this is a reference object that points to the current object. It is also used to refer the members of current object. For example, this() is used to invoke current class constructor. Moreover, this keyword is also used to return the current class instance from the method.
7. **Differentiate abstract classes and concrete classes?**
   Answer:

Abstract Class: An abstract class is a class in which one or more methods are declared, but not defined. We cannot instantiate an object of an abstract class, but a variable of an abstract class type can be declared.

Concrete Class: A concrete class implements a concrete concept. These are used to instantiate objects in our programs to provide implementation details specific to the domain context. The entities that actually we see in our real world are called concrete objects and classes made against these objects are called concrete classes.

8. **What is Overloading of a method? Also give an example.**

   Answer: Method Overloading is a feature of programming languages that allows a class to have more than one method having the same name. a programmer can declare various methods with same name but different arguments. To overload a method, number of arguments or type of arguments must be different.

   Example:

   ```
   addNumbers(int, int)

   addNumbers(double, double)

   addNumbers (int, int, int)
   ```

9. **Define polymorphism?**

   Answer: The word polymorphism, derived from the Greek words **polus** and **morphe** , means "many shapes" or "many forms". **Polymorphism** is the programming language's ability to process objects differently depending on their data type or class. More specifically, it is the ability to redefine *methods* for *derived classes.*

10. **What is has-A relationship in classes?**

    Answer: A Has-A relationship means that an instance of one class has a reference to an instance of another class or another instance of the same class. For example, a car has an engine, a man has hand, legs, etc.

11. **What is an Interface?**

    Answer: An interface is a named collection of static constants and abstract methods. An interface specifies certain actions or behaviors of a class but not their implementations.

12. **What is dynamic dispatch?**

    Answer: Dynamic dispatch is the process to select a particular implementation of a polymorphic method or function to call at run time.

Subjective Part (9*4)

Q.2: Create a Distance class with instance variable feet and inches. Write suitable parameterized constructor, getter and setter methods and a method to add two distance objects in such a way that if resultant inches exceed 12,feet should be incremented by 1 and inches decremented by 12 by using the statement dist3.add(dist1,dist2); where dist1,dist2 and dist3 are objects of Distance class.

Answer:

```
public class Distance {
```

```
private int feet;

private int inch;

public Distance()

{

        this(0,0);

}

public Distance(int f, int i)

{

        feet = f;

        inch = i;

}

public void setFeet(int f)

{

        feet = f;

}

public void setInch(int i)

{

        inch = i;

}

public int getFeet()

{

        return feet;

}
```

```
        public int getInch()

        {

                return inch;

        }

        public void add(Distance d1 , Distance d2)

        {

                this.feet = d1.feet + d2.feet;

                this.inch = d1.inch + d2.inch;

                if(this.inch>=12) {

                        int f = this.inch/12;

                        this.inch= this.inch%12;

                        feet+=f;

                }

        }

        public void PrintDistance()

        {

                System.out.println("Distance: "+feet+"Feet"+inch+"Inch" );

        }

}
```

Q.3: Write a program that creates a Parent class and Child class that inherits the Parent class. Demonstrate Up-casting by using Child class.

Answer:

```
public class Parent{
      public void display(){
        System.out.println("It is parent Class. ")
        }

}

public class Child extends Parent{
        public void display(){
          System.out.println("It is child Class. ")
        }

}

public class test{
        public static void main(String[] Args){
                Parent p = new Child();
                p.display();
        }

}
```

Q.4: Write a program that has Shape class and sub classes are Square and Circle, Use these classes to demonstrate Polymorphism.

Answer:

```
public class Shape{
        public void Print(){
          System.out.println("It is Shape Class. ")
        }

}

public class Square extends Shape{
        public void Print (){
          System.out.println("It is Square Class. ")
        }

}

public class Circle extends Shape{
        public void Print (){
          System.out.println("It is Circle Class. ")
        }

}

public class test{
        public static void main(String[] args){
                shape[] shape = new Shape[10];
                shape[0] = new Square();
                shape[1] = new Circle();
                shape[2] = new Square();
                shape[3] = new Square();
                shape[4] = new Circle ();
                shape[5] = new Circle ();
                shape[6] = new Circle ();
                shape[7] = new Square();
                shape[8] = new Square();
                shape[9] = new Circle ();

        for(int i = 0; i<10; i++){
                    shape[i].Print();
             }
        }

}
```

Q.7.Write a java stream class from java.io package to read text from one files and write to another text file.

Answer:

```
import java.io.*;

public class MyProgram {



          public static void main(String[] args) {

                    String permFile = “D:\\DestFile.txt”;

               String tmpFile =  “D:\\SourceFile.txt”;



               try {



                          BufferedWriter out = new BufferedWriter(new FileWriter(permFile, true));
```

```
            BufferedReader in = new BufferedReader(new FileReader(tmpFile));
            String str;
            while ((str = in.readLine()) != null) {
              out.write(str);
              out.write("\n");
              }
            in.close();
            out.close();
            } catch (IOException e) {
               System.out.println("Source File dont Exist");
            }
        }
}
```