NATIONAL SCHOOL OF BUSINESS MANAGEMENT

BSc in Management Information Systems (Special) (NSBM)–20.3
BSc (Honours) in Software Engineering (NSBM)– 20.3
BSc (Honours) in Computer Science (NSBM)– 20.3
BSc (Honours) in Computer Network (NSBM)– 20.3

Year 01 Semester 02 Examination
06th October 2021
SE101.3- Object Oriented Programming with Java

Instructions to Candidates

1) Answer all questions.
2) **Time allocated for the examination is five (05) hours (Including downloading and uploading time)** . (Note: **No email submissions are accepted under any condition.)**
3) Weightage of Examination: 60% out of final grade
4) Provide answers to the selected questions in the given format under the question.
5) Please upload the document with answers (Answer Script) to the submission link before the submission link expires
6) Answer script should be uploaded in PDF Format
7) Under any circumstances E-mail submissions would not be taken into consideration for marking. Incomplete attempt would be counted as a MISSED ATTEMPT.
8) The Naming convention of the answer script – Module Code_Subject name_Index No
9) You must adhere to the online examination guidelines when submitting the answer script to N-Learn.
10) Your answers will be subjected to Turnitin similarity check, hence, direct copying and pasting from internet sources, friend's answers etc. will be penalized.

Write your Index No 21386

Question 1

a. Define the term 'Object Oriented Programming' (You may use your own words). **(5 marks)**

Object oriented programming is simply a thinking pattern to solve a problem or create a system.it is a programming approach to think a solution for the problem. We can get a idea to achieve our target by using OOP concepts. And java software system is a collection of objects. There are 4 concepts in OOP and they are polymorphism, encapsulation, abstraction , inheritance.

b. Briefly explain how 'Object' and 'Class' related to each other. You may use examples to explain the answer. **(5 marks)**

Object – object is a real world entity. It can be place, person or a thing. We can think of a solution for the problem based on creating objects. When we are creating a system for a company we can take employees as our objects. We can refer other attributes by using employee object. each type of objects has particular attributes and behavior.

Class – we cant create objects without a class. Simply a class is collection of methods and properties. It is template for creating objects. In that company system we can get divisions as our classes. Examples are Marketing division, Finance division , Operations management, Human Resource division.

c. By using practical examples explain the following OOP concepts.

Abstraction **(4 marks)**

Abstraction is concept to representing the essential data and methods. The goal of abstraction is to keep users from seeing unneeded information, and it aids in the reduction of programming difficulty and work.

```
//abstract class

abstract class Laptop{

        public abstract void power_on();

        public void power_off()

        {

              System.out.println("turn off computer");

        }

  }
```

Encapsulation **(4 marks)**

Encapsulation is a principle in Object Oriented Programming that links together data and the functions that modify it, keeping both secure from outside disturbance and mistreatment. Simply in encapsulation we can hide data by using private and we can show methods public.

Private int model_number;

Public string name;

Inheritance **(4 marks)**

We can derive a class from another class for a hierarchy of classes that share a set of attributes and methods. Subclass inherits all the members of its superclass.

```
//parent class
Public class laptop{
    Public void power_on();
}
//Child class
class azus extends laptop{
  public void accelerate(){
      System.out.println("azus:power_on");
    }
}
//Main class
class Main{
   public static void main(String args[]){
      laptop l1 = new azus();    //laptop object =>contents of azus
      l1.power_on();         //calling method
   }
}
```

Polymorphism **(4 marks)**

One object has different types for each environment. If we take a child, he is a student to his teacher, he is a friend to his colleagues, and he is a son to his parents. That's the polymorphism. Simply polymorphism is the same method , but gives different meaning.

There are two types of polymorphism. They are static polymorphism and dynamic polymorphism.

Implement several methods with the same name but distinct parameters inside the same class. This is known as method overloading, and it is a type of polymorphism that is static.

subclass can override a method of its superclass. It creates a form of polymorphism.

d. Explain the difference between 'private' vs. 'protected' access modifier. **(4 marks)**

In private we can only access data within the class.

In protected we can access to data within the class and in other classes which inherit from that class.

e. An abstract class A contains an abstract method abc() and non-abstract method xyz() inside the class. Interface B contains method a pqr() . class c use both abstract class A and interface B and implement its methods. class D contains the main method. Inside the main method create an object from the class D and call the methods abc(), xyz(), and pqr(). **(10 marks)**

```
<Type the source code in the editor and paste the source code here. Use comments much as possible to explain the code>

//abstract class

abstract class A{

 //abstract method

        public abstract void abc();

//non abstract method

        public void xyz()

        {< Method body>}    }

//interface

Interface B{

Public void pqr();

}

Public class C extends A implements B

{

<method body>

}

Public class D{

Public static void main (String[args])

//Creating a object

D a1= new D()

 a1.abc();

a1.xyz();

a1.pqr();

}
```

Question 2

Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables-a part number(type String),a part description(type String),a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. In addition, provide a method named getInvoice Amount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. Write a test application named InvoiceTest that demonstrates class Invoice's capabilities. **(20 marks)**

```
Public class Invoice
{
        Private String number , description ;
        Private int quantity;
        Private double price, InvoiceAmount;

        Public Invoice (String num , String des , int q , double p )
        {
                number = num ;
                description = des;
                quantity = q;
                price = p;
        }
        Public void inputNumber (String num)
        {
                number = num;
        }
        Public void inputDescription (String des)
        {
                description = des;
        }
```

```
Public void inputQuantity (int q)
        {
                quantity = q;
        }
        Public void inputPrice (double p)
        {
                price = p;
        }

        Public String returnNumber()
        {
                Return number;
        }
        Public String returnDescription()
        {
                Return description ;
        }
        Public int returnQuantity()
        {
                Return quantity;
        }
        Public double returnPrice()
        {
                Return price;
        }
```

```
Public double returnInvoice()
        {
                If (quantity < 0)
                {
                        Quantity = 0;
                }
                If (price < 0.0 )
                {
                        Price = 0.0;
                }
                Else
                {
                        invoiceAmount = price * quantity;
                        return invoiceAmount;
                }
        }
}
```

```
Public class InvoicTest
{
        Public static void main (String[] args)
        {
                Invoice i1 = new invoice (“nethu123”, “few”, 5 , 1000.00)
                System.out.println (“Invoice: ”+i1.getInvoice());
        }
}
```

Question 3

Write a program to illustrate creation of threads using runnable class.(start method start each of the newly created thread. Inside the run method there is sleep() for suspend the thread for 500 milliseconds. Write the method to display your Index No 5 times using a loop as follows). **(10 marks)**

```
public class Q3 implements Runnable
{
  public void run()
   {try{
          Thread.sleep(500);
  }
catch(Exception e)
         {  System.out.println(e);
            }
for( int x=1;x<=5;x++)
     {
   System.out.println("Index Number: 21386");
     } }}
//main class
public class Q_3 {
public static void main(String[] args)
  {
//creating a object
     Q3 q1= new Q3();
     Thread obj1=new Thread(q1);
       obj1.start();
    }
  }
```

Question 4

Write a program, to show the behavior of 'try / catch block and finally'. In this, check the following two types of exceptions.

A number is divided by zero.

The value assigned inside the array is out of the boundary of the array.

```
Package question4;

public class q4

 public static void main (String[] args)

{ int a=10, b=0 , c;

try

{ c= a/b;

System.out.println(c);

for(int i=0; i<=5; i++)

 { System.out.println (i);  }

 }

catch (ArrayIndexOutOfBoundaryException e1)

 {

 System.out.println (e1.getMessage());

}

catch (ArithmeticException Exception e2)

 {

System.out.println (e2.getMessage());

}

finally

{

 System.out.println ("always Display ");}

} } }
```

Question 5

Write a Java program to create a text file named 'Sample.txt.' Write your Index Number and first name with the last name in two separate lines in the above file. Read the file and display the content of the text file. **(10 marks)**

```
//Writing class

Package q5;

import java.io.*;

import java.io.IOException;

public class WriteToFile {

  public static void main(String[] args) {

    try {

      FileWriter M1 = new FileWriter("Sample.txt");

      M1.write("21386"); M1.write("Nethum Pabasara");

      M1.close();

      System.out.println("Successful.");

    } catch (IOException e1) {

      System.out.println("error.");

      e1.printStackTrace();

    }

  }

}
```

```
//reading class
Package q5;
import java.io.*;
import java.io.IOException;
public class ReadFile {
   try {
     File file1 = new File("sample.txt");
     Scanner myReader = new Scanner(file1);
     while (myReader.hasNextLine()) {
       String details = myReader.nextLine();
       System.out.println(details);
     }
     myReader.close();
   } catch (FileNotFoundException e2) {
     System.out.println("error.");
     e2.printStackTrace();
   }
 }
}
```

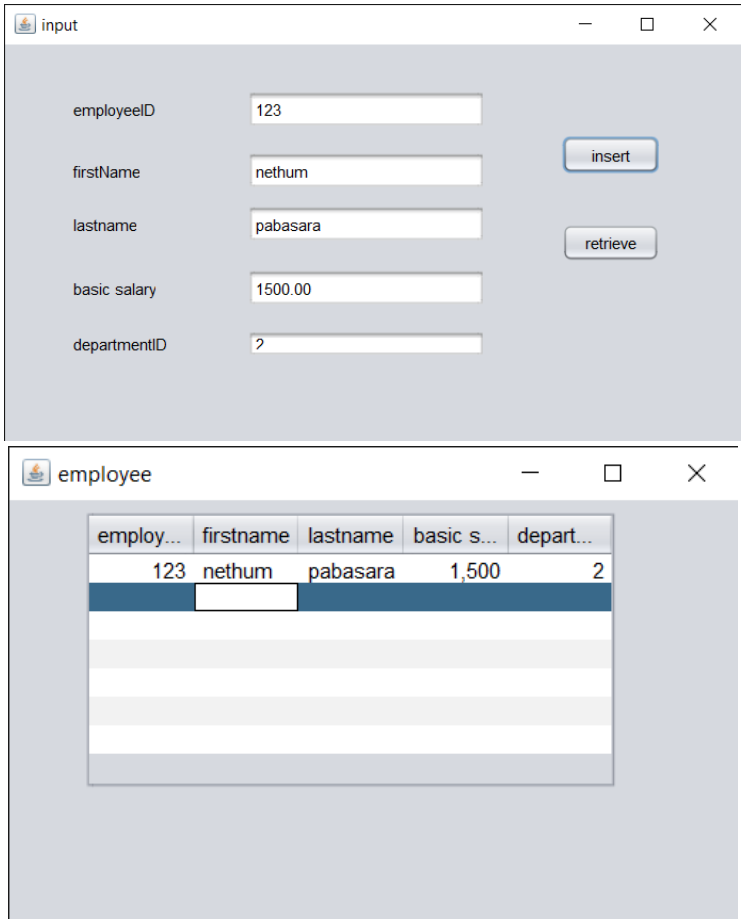
Question 6

Create a database 'MyDB' and inside the database create a table 'Employee' with the following fields.

EmployeeID(int), FirstName(varchar),LastName(varchar),BasicSalary(float), DepartmentID(int)

Design a suitable GUI and develop a program to show the a). Database operation 'INSERT' (one record at a time) and b). Retrieving the table records into a 'JTable'.

Create the appropriate User Interface and past the screens here. **(2 marks)**



Include the insert operation source code. **(4 marks)**

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{  String FirstName,LastName;
Int employeeID,departmentID;
Float basic_salary;
 EmployeeID,=jTextField1.getText(); // input EmployeeID
 First_Name =jTextField2.getText(); //  input First Name
 Last_Name=jTextField3.getText(); // input Last Name
Basic_salary= jTextField4.getText(); // input basic salary
 departmentID =jTextField5.getText(); //  input department ID
 int empID=Integer.parseInt(employeeID); // get Employee id as a integer value
int depID=Integer.parseInt(departmentID); // get department id as a integer value
 try // Starting exception handling part
 {
 Class.forName("com.mysql.jdbc.Driver"); // Create mysql JDBC driver class
//connection with JDBC
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sample","root","");
 Statement st=con.createStatement(); // Create st object from connection
//get input values using sql queries
String sql="INSERT INTO employee
 VALUES("+empID+",'"+FirstName+"','"+LastName+"','"+basic_salary+depID+"')";
 st.executeLargeUpdate(sql); // input SQL Queries to st object
// Show messagewhen record inserted
 JOptionPane.showMessageDialog(this, " record is inserted ");
 con.close(); // Close the con
 }
```

```
catch(Exception e) // If an exception occurred
 {
 System.out.println("Error."); // Show the error message
 }
 }
```

Include the display records source code. **(4 marks)**

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)

{

 Try //create exception handling

 {

 Class.forName("com.mysql.jdbc.Driver");

 Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sample","root","");


 Statement st=con.createStatement();

//get inputs from sql queries

String sql="SELECT * FROM student";

 ResultSet rs=st.executeQuery(sql); // Set execution to SQL queries

 while(rs.next()) // While rs object has lines

 {

 String employeeID=String.valueOf(rs.getInt(1)); // input User ID to table column as 1st column

 String First_Name=rs.getString(2); // input First Name to table column as 2nd column

 String Last_Name=rs.getString(3); // input Last Name to table column as 3rd column

 String  basic_salary=rs.getString(4); // input Batch ID to table as 4th column

String department_ID= rs.getString(5); // input department ID to table as 5th column

 String TableData[]={empID,first_name,last_name,basic_salary,depID}; // keep them in a array called
TableData

//get employee as a the default table

DefaultTableModel tblModel=(DefaultTableModel)jTable1.getModel();

//display employee detail as rows

tblModel.addRow(tbData)

 }

 con.close(); // Close the con

}
```