NAME:- SUMAN MISHRA
BIM (3rd SEM) Section 'A'
Java- programming - I

TU - 2017

Brief Answer question.

1. Define recursion.

⇒ Recursion in java is a process in which a Method calls itself continuosly. A Method of java that calles itself continuosly is called recursion Method.

2. why java is a strongly typed programming languages?

⇒ Java is a strongly typed programming languages because every variable Must be decleared with a data type.

3. differentiate between private and public constructor.

⇒ public constructors are used to instantiate and initialize the field during object instantiation.

private constructor are used to restrict the ~~initi~~ instantiation of object using 'new' operator.

4. Why identifiers can't be used as keywords?
⇒ we cannot used ~~as~~ keywords as variable name because keywords have predefined meaning.
for ex:- int float.
The above ~~example~~ code is wrong. It's because float is a keyword and cannot be used as a variable name.

5. what is the role of JVM?
⇒ It interprets the compiled java code known as the byte code and helps in program execution depending upon the specific platform.

6. List out 3 oops principle.
⇒
(i) encapsulation
(ii) inheritance
(iii) polymorphism.

7. what is the use of finalize() method?
⇒ finalize() method is used to perform some final operations or clean up operations on an object before it is removed from the memory.

8. what are different thread states?
⇒ the different thread states are:-
new, runnable, time wating, blocked, terminated.

9. How can variable length arguments be helpful?
⇒ A variable length argument are most useful when

the number of argument to be passed to the method is not known beforehand. They also reduced code as overloaded method are not required.

10. Define Auto-boxing.

⇒ Auto-boxing is the automatic conversion that the java compile make between the primitive types and their corresponding object wrapper classes. For example:- Converting an int to an integer, etc.

Exercise problem:-

11. Write a java program that asks the user to enter number in an array of size 'n'. Then display only the number that are divisible by 2 and 3.

```
Import java.util.scanner;
public class Array
{
    public static void main(String[] args)
    {
        int n, i;
        System.out.println("Enter the number");
        Scanner sn = new Scanner(System.in);
        n = sn.nextInt();
        System.out.println("Enter the numbers");
        int num[];
        for(i=0; i<=n; i++)
```

```
num[i] = sn.nextInt();
if(num[i]%2==0
        System.out.println(num[i]);
    }
}
}
```

12. create a class student with instance variables, roll_no and two Methods to read and display the roll no. Then, create another class Test that inherits class student, consisting of its own instance variables to hold the marks of two subjects and also Methods to read and display the marks. finally, create another class Result the student. similarly, it has Methods to ~~creo~~ calculate and display the total. create some instances of above classes and demonstrate inheritance.

```
class student
{
    int roll_no;
    public void readRoll(int r)
    {
        roll_no = r;
    }
    public void displayRoll()
    {
        System.out.println(roll_no);
    }
}
```

~~num[i] = sn.nextInt();~~
~~if (num[i]%2==~~

```
class Test extends student
{
    int Markjava, Markweb;
    public void readMark(int Mj, int Mw)
    {
        Markjava = Mj;
        Markweb = Mw;
    }
    public void displayMarks()
    {
        System.out.println("java: " + Markjava + "." + web);
        System.out.println( + Markweb);
    }
}
Class Result extends ~~Student~~ Test
{
    int total;
    public void calculate()
    {
        total = Markjava + Markweb;
    }
    public void display()
    {
        System.out.println("The total Mark is: " + total);
    }
    public static void Main(string[]args)
    {
        Result obj = new Result();
```

```
        obj.readRoll(1);
        obj.readMarks(49,40);
        obj.calculate();
        obj.displayRoll();
        obj.displayMarks();
        obj.display();
    }
}
```

13.

```
interface shapes
    public void get_data(int l, int b);
    public void display_area();
}
class Rectangle implements shape
{
    int length, breadth, area;
    public void get_data(int l, int b)
    {
        length = l;
        breadth = b;
    }
    public void display_area()
    {
        area = length * breadth;
        system.out.println("The area of rectangle is: "+area);
    }
}
class square implements shape
{
```

```
int length, breadth, area;
public void get_data(int l, int b)
{
    length = l;
}
public void display_area()
{
    area = length * ~~square~~ length;
    system.out.println("The area of square is:" +area);
}
}
class demonstrate
{
public static void main(string[]args)
Rectangle obj = new Rectangle();
obj.get_data(10,20)
obj.display_area();
square obj1 = new square();
obj1.get_data(10,0);
obj1.display_area();
}
}
```

14.

```
class Array
{
    public static void main(string[]arg)
    {
        string[] contry = {"Nepal","use","Austria","Brazil",
                           "paragua","canada"};
```

```
int i;
for(i=0; i<=6; i++)
{
    if(country[i].endsWith("a")
    {
        System.out.println(country[i]);
    }
  }
 }
}
```

7.

=> In java we can create our own exception class and throw that exception using throw ~~ex~~ keyword. These exceptions are known as user-defined or custom exceptions.

```
class javaException
{
  public static void main(String args[])
  {
    try{
        throw new MyException.
      } catch (MyException e)
        {
          System.out.println(e);
        }
    }
}
class MyException extends Exception
{
```

```
int a ;
MyException (int b)
{
    a = b;
}
public string to string ()
{
    return ("Exception NUM = " + a );
}
}
```

"Group-c"

Comprehensive Answer question.

15.

```
import java.io.* ;
import.java.util.scanner;
class fil
{
    public static void Main (string[]args)
    throw IOException
    {
        try
        {
            filewriter fw = new filewriter ("Emp.txt");
            Buffered writter bw = new Buffered writer(fw);
            String name, id, DOB ;
            Scanner scan = new scanner (System.in);
            System.out.println ("Enter name, id, & DOb: ");
            name = scan.nextline();
            id = scan.nextline();
```

```
DOB = scan.nextline();
bw.write(id);
bw.write(DOB)
bw.write(name);
bw.close();
}catch(Exception e)
{
    System.out.println(e);
}
try
{
  FileReader fr = new FileReader("EMp.txt");
  BufferedReader br= new BufferedReader(fr);
  String MSG = null;
  while((MSG = br.readline())!= null)
  {
      system.out.print(MSG);
  }
  br.close();
}catch(Exception e)
{
    System.out.println(e);
}
}
}
```

16.

```
Class new thread implement Runnable
{
   string name; Thread t;
```

```
new thread (String tname)
{
    name = tname;
    t = new Thread (this, name);
    System.out.println ("New Thread: "+t);
    t.start();
}
public void run()
{
  try{
     for (int i=3; i>0; i--)
     {
        System.out.println (name+" : "+i);
        Thread sleep (1000);
     }
  } catch (Interrupted exception e)
  {
     System. out.println ("interupted");
  }
    System.out.println(name+" existing");
  }
}
class Multiple_thread
{
   new newthread ("KCHIT");
   new newthread ("NCC");
   new newthread "DAV");
try{
    Thread.sleep(5000);
} catch (Intmupted exception e)
```

```
}
    System.out.println("Interrupted");
}
System.out.println("Main thread exiting");
}
}
```

TU - 2018

1. What is the significance of package?

⇒ The significance of package are:-

(i) package is used to categorize the class and interface so that can be easily maintained.

(ii) package provide access protection.

2. List out any 4 feature of java programming languages?

⇒ Any 4 feature of java programming language are:-

(i) object-oriented (iii) Dynamic

(ii) simple (iv) Multithreaded.

3. Why java is called architecture neutral language?

⇒ Java is called architecture neutral language because, java allows its application to compile on one hardware architecture and to execute on another hardware architecture.

4. How has the concept of exception handling helped java to be robust?

⇒ The main idea behind exception handling is to make

programs more reliable and robust. Many programming languages provides exception handling mechanism to allow software developer to define exception condition. This concept of exception handling helped java to be robost.

5. Difference between constructor and method?

=>

(i) constructor are the ~~sty~~ special types of method used to initialize object of its class whereas Method are the set of instruction that involved of any point in a program to return a result.

(ii) The purpose of constructor is to create an instance of a class whereas. The purpose of Method is to execute java class.

6. What is the advantages of variable length argument?

=> The advantages of variable length argument as follow:-

(i) variable becomes an array of given types.

(ii) only legal for final argument of Methods.

7. What is the use of serialization ~~pro~~ process?

=> The use of serialization process are:-

- to save object to same permanent storages.
- to pass on the another object via the output stream class.

8. what are the use of "extends" and "implement" keyword?
=> The extends keyword is used to indicate that a new class is derived from the base class using inheritance.

The implement keyword is used to make a class address to contract defined by an interface.

9. How does interface differ from abstract class?
=> (i) Interface can have only abstract method but Abstract class can have both abtract and non-abract method.

(ii) Interface support multiple inheritance whereas Abstract class ~~ab~~ doesnot support multiple inheritance.

10. Define Auto-boxing.
=> Autoboxing is the automatic conversion that the java compile make between the primitive types and their corresponding object wrapper classes. For example:- converting an int to an integer etc.

"Group-B"

Exercise problem.

11.

~~import Ja~~
~~import Java~~

Q.M-11

```
import java.util.Scanner;
class Add
{
    public static void Main(String args[])
    {
        int M, n; int SUM = 0;
        Scanner arr = new Scanner(System.in)
        System.out.println("Enter row & colum of Matrix");
        M = arr.nextInt();
        n = arr.nextInt();
         int Ma[][] = new int[M][n];
        System.out.println("Enter element of Matrix:");
        for(int i = 0; i < M; i++)
        for(int j = 0; arr.nextInt();
        for int Ma[i][j] = arr.nextInt();

        for(int i = 0; i < M; i++)
        {
            for(int j = 0; i < n; j++)
            {
                SUM = SUM + Ma[i][j];
            }
        }
        System.out.println("The SUM is : " + SUM);
    }
}
```

Q. N-12

-> The two ways to create thread are:-

i) Subclassing the Thread class and instantiating a new object of that class.

ii) Implementing the Runnable interface.

The life-cycles of thread are:-

1) New
2) Runnable
3) Running
4) Non-Runnable (Blocked)
5) Terminated.

1) The thread is in new stage if you create an instance of thread class but before the invocation of start() Methods.

2) The thread is in runnable stage after invocation start() Method, but the thread scheduler has not selected it to be the running thread.

3) The thread is in running stage if the thread scheduler has selected it.

4) The is the stage when the thread is still alive, but is currently not eligible to run.

5) A thread is in terminated or dead stage when its run() Method exits.

Q.N-13

```
Abstract class fMachine
{
    abstract void getdata();
    void putdata()
    {
        System.out.println("code is: Af403");
        System.out.println("Name: Nepal Airlines");
        System.out.println("capacity: 4030");
    }
}
class Airplane extends fMachine
{
    string code, name, capacity;
    void getdata()
    {
        super.putdata();
        System.out.println("overridden in abstract class");
    }
}
class Main
{
    public static void main(string args[])
    {
        Airplane object = new Airplane();
        object.getdata();
    }
}
```

Q.N-14

```
import java.util.scanner;
class check
{
    public static void main(string args[])
    {
        int i; n;
        scanner input = new scanner (system.in)
        system.out.println ("enter size of array");
        n = input.nextInt();
        string arr[] = new string[n];
        system.out.println ("enter 10 name");
        for(i=0; i<10; i++)
        {
            arr[i] = input.nextline();
            system.out.println(arr[i].replace(i,'1'));
        }
    }
}
```

Q.N-15

```
// program to create file
import java.io.filewriter;
import java.io.fileReader;
import java.util.scanner;
class Main
{
    void show()
    {
        string str;
```

```
Scanner info = new Scanner (System.in)
System.out.println("Enter your name, address & college")
filewriter.(student.txt) = info.nextInt();
if((pupil.txt).exists())
{
  st2 = (pupil.txt).write(student.txt);
}
  else
  System.out.println("file does not exist");
  System.out.println(str);
  }
}
class info_std
{
    public static void main(string args[])
    {
      Main ob = new Main();
      ob.show();
    }
}
}
```

8/N-16 "Group-C"

→ Exception are events that occur during the execution of programs that disrupt the normal flow of instruction.

Example of try-catch.

```
class try-catch-demo
{
    public static void main(String args[])
    {
        int a=100, b=25, c=25;
        int r;
        try{
          System.out.println("Before exception");
          r= a/(b-c)
          System.out.println("After exception");
          }
        catch(ArithMetic exception e)
        {
            System.out.println("exception handled");
            System.out.println(e);
        }
    }
}
```

Example of try-finally

```
class finally-demo.
{
    public static void main (String args[])
    {
      try{
        System.out.println ("Hello");
        int a=5/0;
        System.out.println(a);
      }
```

```
catch(Null pointer Exception e)
{
    System.out.println("Exception Occurs");
}
finally
{
    System.out.println("finally block");
}
system.out.println("Thank you");
}
}
```

Q. N-IT

⇒ Multiple inheritance is the process of deriving a new class from more than one super classes.

Java doesn't support Multiple inheritance with classes, but java supports Multiple inheritance with interfaces.

Example (format)

```
Class A
{
    statement___.
}
Class B extends A
{
    statements ___.
}
Class C extends B{
    statement.___
}
```

```
class college
{
    college()
    {
        System.out.println("KCMIT college");
    }
}
class faculty extends college
{
    faculty()
    {
        System.out.println("BIH");
    }
}
class semester extends college
{
    semester()
    {
        System.out.println("Third semester");
    }
}
class Main
{
    public static void main(string args[]){
        college clg = new college()
        clg.college();
        faculty fclt = new faculty();
        fclt.faculty();
        semester sem = new semester();
        sem.semester();
        System.out.println("Multiple inheritance achieved");
        System.out.println("program termination ---");
    }
}
```

TU - MARKUP - 2019

"Group-A"

1. What is a bytecode?

=> Java Byte Code is the language to which java source is compiled and the java virtual Machine understands.

2. List out the principle of object oriented programming language.

=>

(i) Encapsulation
(ii) polymorphism
(iii) Inheritance.

3. Why can't keywords be used as identifiers?

=> 'keyword' only consists of letters whereas identifier consists of letters and digits. So that keywords can't be used as identifiers. Also keywords are built-in words but identifiers may be user-defined too.

4. write uses of '+' operator.

=> In string handling '+' operator acts as concatination of two or more string and character but in arithmetic operation it adds two or more number.

5. How does a default constructor differ from parameterized constructor?

=>

Default construction
- Has no parameter.
- No need to pass any argument.
- used to initialize object with same data.

parameter constructor
- Has parameter.
- needs to pass defined argument in parameters.
- used to create distinct object with different data.

Q.N-6 How does final differ from finally?

=>

final :-
-> It is keyword
-> It is used to restriction on class, method or variable.
-> After asigned 'final' keyword, we can't change the value of variable.

finally :-
-> It is a block
-> It is used in exception handling along with try-catch block.
-> It will execute if there is even no exception.

7. What is the importance of synchronization in multi-threaded program?

=> Java synchronization is better option where we want to allow only one thread to access the shared resource. It allows the ~~continuous~~ continuous

functioning without corrupt data or execution flow by one thread to another.

8. Which class is the superclass of all classes?

⇒ The class being inherited from is called supper class of all classes. It is also called as parent class.

9.

⇒ filewriter.

⇒ It is used ~~just~~ to write in file.

⇒ It just writes in a file.

⇒ It allows append mode.

printwriter.

⇒ It is used in write everywhere.

⇒ It writer firstly & display in screen.

⇒ It erase the data from written area.

10. What is enumeration?

⇒ Enumeration is a list of named constants. 'enum' is mainly used to define our own data type (enumerated Data types)

"Group-B"

11

```
import java.util.Scanner;
class Main
{
  public static void Main (string args[])
  {
     System.out.println ("enter 10 number");
     Scanner input = new Scanner (System.in);
      int i, SUM = 0;
      for (i=0; i<10; i++)
      {
           SUM = input.nextInt();
      }
      System.out.println ("SUM is:"+SUM);
  }
}
```

12

Interface:

```
interface NUMber
{
    int square (int x);
    int cube (int x);
}
Class NUMberDEMO implement NUMber
{
    void operation (int x)
    {
        int square (int x);
```

```
    int cube(int x);
}
void display()
{
    System.out.println("square of "+x+" is: "+(x*x));
    System.out.println("cube of "+x+" is: "+(x*x*x));
}
class Main
{
    public static void Main(String arg[])
    {
        NumberDemo ob = new NumberDemo();
        ob.operation(5);
        ob.display();
    }
}
```

L3

```
// Subclass & superclass

class Book
{
    int book-id;
    int pages;
    void BookInfo(int id, int pg)
    {
        book-Id = Id;
        pages = pg;
        System.out.println(" Book Id: "+book-d", page: "+
    }
```

```
class fiction Book extends Book
{
    string name;
    void bookinfo(int id, int pg, string name)
    {
        this.name = name;
        System.out.print(Book Info());
        System.out.println(name);
    }
}
class Main
{
    public static void Main(string args[])
    {
        fiction Books ob = new fiction Book();
        fiction Book obj = new fiction Book();
        ob.bookinfo(10110, 255, "Hobbit");
        obj.bookinfo(19621, 677, "Royal");
    }
}
```

14.

```
// Input from user
import java.util.scanner;
class Input
{
    System.out.println("Enter seven countries' NO
    scanner input=new scanner(system.in);
    string arr[]=new string[7];
    for(int i=0; i<7; i++)
    arr[i]=scan.nextline();
    for(int i=0; i<7; i++)
    {
    if(arr[i].startswith("a")||arr[i].startsw
    ("e")||arr[i].startswith("i")||arr[i].start
    with("o")||arr[i].startwith("u"))
    System.out.println(a[i]);
    }
}
```

Q.N-15

```
// file Handling
import java.io.filewriter;
import java.io.fileReader;
import java.util.scanner;
class file
{
    string name;
    string address;
    int number;
    void write(string name, string add, int no.)
    {
```

```
        name = nam;
        address = add;
        number = no;
    file writer folder = new file writer();
    writefile.folder(string name, string address, int number);
    system.out.println(display.folder());
    Scanner input = new Scanner(system.in);
    for(int i=0; i<3; i++)
    {
        system.out.print("enter name, address & ");
        system.out.println("no. of student of college");
        arr[i] = scan.nextline();
    }
}
```

"Group-C"

Q.N-16

// Thread priority in java.

→ simply in multithreading environment of java, thread scheduler assigns processer to a thread based on priority of thread. whenever we create thread in java, it always has, some priority assigned to it. priority can either be given by java virtual machine (JVM) while creating the thread or it can be given by clearly.
Eg:-

```
import java.lang.*;
class ThreadDemo extends Thread
{
    public void run()
    {
        System.out.print("Thread name:");
        System.out.print("Thread.currentThread().getName());
        System.out.println("Thread priority:"+Thread.current
        System.out.println("Thread().getpriority");
    }
    public static void main(String arg[])
    {
        ThreadDemo ob1 = new ThreadDemo();
        ThreadDemo ob2 = new ThreadDemo();
        ob1.setpriority(Thread.MIN_PRIORITY);
        ob2.setpriority(Thread.MAX_PRIORITY);
        ob1.start();
        ob2.start();
    }
}
```

output:-

```
Thread name: Thread-0
Thread priority: 10
Thread name:-1
Thread priority:1
```

Q.No-17

```
// Inner class can access all property of outer class
class outer_Demo

class outer_DEMO
{
    int num;
    private class inner_DeMo
    {
        public void print()
        {
            System.out.println("This is an Inner class");
        }
    }
    void display_Inner()
    {
        Inner_DEMO = new Inner_DEMO();
        inner.print();
    }
}
public class Main
{
    public static void Main(String args[])
    {
        Outer_DEMO outer = new outer_DEMO();
        outer.display_Inner();
        // Accessing the display_Inner() Method.
    }
}
```

Output :-

This is inner class.