Name: Nutan Patil
Rollno - MC-I - 89

Q1 Discuss Object life time & garbage collection

The life time of the object in Java is described in some of the under mentioned points.

1 Creation - objects are created using new
1 Keyword
2 Initialization - object field are set to default values. eg. 0 for numeric types 'null' for reference.
3 Reference - objects are accessed & used through reference. References can be stored in Variables, passed as method parameter.
4 Scope & visibility - The objects are accessed & are determined by where the reference is declared. Objects can exist within a method as instance variables or static variables
5 Life time - Objects remain in memory as long as there is at least one reference pointing them.

Garbage collection
i Garbage collection is responsible for removing memory occupied by objects with no longer reference.
ii When there are no more reference to an object it becomes a candidate for garbage collection

iii) The JVM's garbage collection identifies are reclaims memory occupied by unreferenced objects.

iv) we use two methods in garbage collection

- finalize()
- gc()

```
public class TestGarbage {
  public void finalize () {
  System.out.println (" object is garbage collected");
  }
  public static void main (String args []) {
    TestGarbage tgb = new TestGarbage ();
      tgb = null;
    System.gc();
  }
}
```

output → object is garbage collected.

Q.2 Write a program to override abstract class method.

```
abstract class Math
{
  public void abstract sum();
}
class Addition extends Math
{
  public void sum()
  {
    System.out.println (5+5);
  }
}
class substraction extends Math
{
  public void sum()
  {
```

```
        System.out.println (10-5);
    }
}

public class main
{
public static void main (String args[]);
{
Addition ad = new Addition ();
subtraction sb = new subtraction ();
  ad.sum ();
  sb.sum();
}
}
```

output :- 10
5

Q3 Explain import and static import with suitable examples.

i) In Java the 'import' statement is used to bring in classes, interfaces or used to bring in classes, interfaces or packages into the current source file.
ii) This helps in using types from other packages without having to use their fully qualified names. There are
iii) two types of import as follows:-

a) Regular import:-

Syntax 'import package-name. class-name;

for eg:- import java.util. Arraylist;

The above example allows us to use 'Arraylist' without specifying its fully qualified name in the code.

b) static import :-

Syntax:- import static package-name. class-name;

e.g:- import static java.lang.math.*;

Static import are specifically for static members of a class.

import :-

```
import java.util.scanner.*;
public class main {
public static void main (String args[]) {
 Scanner sc = new scanner ();
System.out.println (" enter a number :");
  int m = sc.nextInt();
System.out.println (" You entered : " + m);
}
}
```

output :- enter a number : 10
You entered : 10

Static import :-

```
import static java.lang.math.*;
public class main {
public static void main ( String args[]) {
  double a = sqrt(25);
  double b = pow (5, 3);
System.out.println (" square root of 25 : " + a);
system.out.println (" 5 to the power of 3 : " + b);
}
}
```

output :- Square root of 25 : 5
5 to power of 3 : 125

Q4 Difference between String builder and String Buffer.

String Builder	String Buffer
i) String Builder was introduced in Java 1.5	String Buffer was introduced in java 10
ii) multiple thread can call the	Multiple thread cannot call the

methods of string builder simultaneously.	methods simultaneously.
iii) String Builder is not thread safe class	String Buffer is called a thread safe class.
iv) It is faster, therefore more efficient	It is slower and less efficient.
v) example :- StringBuilder a= new StringBuilder (str);	example :- StringBuffer b= new StringBuffer (str);

Q.5 Write a program to demonstrate the inter thread communication.

```
package java;
import java.util.*;
Class customer
{

 int amount = 10000;

Synchronized void withdraw (int amount)
{

System.out.println (" going to withdraw");
if (this.amount < amount)
{
System.out.println (" less balance... waiting...");
try {
    wait ();
  }
Catch (exception e)
  {
  }
```

```
this.amount = amount;
System.out.println("withdraw completed");
}
Synchronized void deposit(int amount)
{
System.out.println("going to deposit");
    this.amount += amount;
System.out.println("deposit complete");
  notify();
 }
}
class Test
{
public static void main (String args[])
{
final customer c= new customer();
 new Thread()
 {
 public void run() {
   c.withdraw (15000);
 }
}.start();
new Thread()
{
 public void run()
 {
   c.deposit (10000);
 }
 }.start();
}
}
```

output:- going to withdraw, less balance; waiting for deposit
going to deposit, deposit completed
withdraw completed.

Scanned with OKEN Scanner

Q.6 Explain Sequence, map with suitable example.

i) A sequence typically refers to an ordered collection of elements, where each element can be identified by an index or a key. Sequence can be linear (like arrays and list) or non-linear (like tree and graphs)

```
package java;
import java.util.*;
public class sequence
{
public static void main (string args [])
{
list < string > sequencelist = new Arraylist <> ();
    sequencelist.add ("Hello");
    sequencelist.add ("Java");
    System.out.println ("first element:" + sequencelist.get(0))
System.out.println("second element:" + sequencelist.get(1));
}
}
```

output :-
first element = Hello
second element = Java.

Arraylist is used to create a linear sequence o string.

ii) Map is collection of key values pairs where each key is associated with exactly one value. Maps are also known as dictionaries or associative arrays in other programming language. In Java the 'map' interface is implemented by classes such as 'Hashmap' or 'Tree map'.

```
package java;
import ;ava.util.*;

public class mapexample {
 public static void main (String args[])
 {
  map <String, Integer> student map = new Hashmap ();
    studentmap.put ("Jhon", 23);
    studentMap.put (" Tom", 25);
    Set set = studentMap.entryset ();
    Iterator itr = set.iterator();
    while (itr.hasNext ())
    {
    Map.entry entry = (map.entry) itr.next ();
    System.out.println (entry.get key () + " " + entry.get
                                                  value());
    }
  }
}
```

output :- John 23
Tom 25

Q7 explain adapter class with example

Java adapter classes provide the default implementation of listener Interface. If you inherit the adapter class, you will not be forced to provide the implementation of all the methods of listener interface so it saves code.

```
import java.awt.*;
import java.awt.event.*;
public class Adapterexample
{
  Frame f;
  Adapterexample ()
```

```
{
 f= new frame ("window Adapter");
 f.addwindowlistener (new windowAdapter ()
 {
 public void windowclosing (windowEvent we)
  f.dispose ();
 }
 });
 f.setsize (400, 400);
  f.setvisible (true);
  f.setlayout (null);
 }
 public static void main (String args[])
 {
  new AdapterExample ();
 }
}
```

Q.8 Write a servlet program to find GCD of two given values (use web.xml, index.html and servlet java)

for this program we need to have three main files

- index.html (for user Input)
- web.xml (for servlet configuration)
- servlet file

i) index.html

```
<! DOCTYPE html>
<html>
<head>
<title> GCD calculator </title>
</head>
<body>
<h2> GCD Calculator </h2>
<form action= " GCDservlet" method: "post">
<br>
```

```
<p> Enter first number </p>
<input type="text" name="num1"><br>
<p> Entere second number </p>
<input type="text" name="num2"><br>
<input type="submit" value="calculate">
</form>
</body>
</html>
```

ii) web.xml

```
<? xml version="1.0" encoding="UTF-8" ?>
<web-app xmlns="https://xmlns.jcp.org/xml/ns
                                  /javaee"
    xmlns:xsi="https://www.w3.org/2001/
                                  xmlschema-instance"
xsi:schemalocation="https://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/
    web-app-4-0.xsd"
    version="4.0">
<servlet>
<servlet-name> GCDservlet </servlet-name>
<servlet-class> GCD servlet </servlet-class>
</servlet>
<servlet-mapping>
<servlet-name> GCD servlet </servlet-name>
<url-pattern> /gcd servlet/ </url-pattern>
</servlet-mapping>

<welcome-file-list>
<welcome-file> index.html </welcome-file>
</web-app>
```

iii) GCD servlet.java

```
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

public class GCDServlet extends HttpServlet
{
protected void doPost (HttpServletRequest request,
HttpServletResponse response) throws ServletException (IOException)
{
String strNum1 = request.getParameter ("N1");
String strNum2 = request.getParameter ("N2");
int n1 = Integer.parseInt (strNum1);
int n2 = Integer.parseInt (strNum2);
int gcdResult = findGCD (n1, n2);

response.setContentType ("text/html");
PrintWriter out = response.getWriter();
out.println ("<htm><body>");
out.println ("<h2> GCD Result : </h2>");
out.println ("<p> GCD of " + n1 + " and " + n2 + " is "
+ gcdResult + " </p>");
out.println ("</body></html>");
}
private int findGCD (int a, int b)
{
while (b != 0)
{
int temp = b;
b = a % b
a = temp;
```

```
}
return math.abs (a);
}
}
```

Q9 write a program, to, show CRUD operations using JDBC. Assume suitable structure

i) Create table structure:-

```
CREATE TABLE employees
(id INT PRIMARY KEY;
name VARCHAR (50),
designation VARCHAR (50),
Salary DOUBLE)
```

ii) CRUD operations using JDBC:-

```
import java.sql.*;
public class EmployeeCRUD
{
static final string JDBC_URL=
    "jdbc:mysql://localhost:3306/stud";
Static final string JDBC_user = "A";
Static final string JDBC_Pass = "1";
public static void main (string args[])
{
try connection c = DriverManager.getConnection
            (JDBC_URL, JDBC_user, JDBC_pass))
{
Statement s= c.create statement();

insert Employee (5, "x", "Developer", 50,000.0);
System.out.println ("Read:");
read employee (5);
```

```
updateEmployeeSalary (5, "x", 75,000.0);
System.out.println ("After update:");
readEmployee (5);

deleteEmployee (5, "x");
System.out.println ("After deletion");
readEmployee (5);
}
Catch (SQLException e)
{
e.printStackTrace ();
}
}

private static void insertEmployee (Statement s)
throws SQLException
{
String selectQuery = "select * from employee";
Resultset r = s.executeQuery (selectQuery);

while (r.next())
{
int id = r.getInt ("id");
String n = r.getString ("Name");
String designation = r.getString ("Designation");
double salary = r.getDouble ("salary");

System.out.println ("ID", +id + "Name:" +n+ "Designation"
+ designation + "salary:" + salary);
}
}

public static void updateEmployeeSalary
(Statement s, String n, double news) throws SQLexception
{
String upquery = "update employee SET salary =
```

```
+ news + " where name = " + n);
  s.execute query (upquery);
 System.out.println ("salary update to news");
}

private static void deleteEmployee (Statement s, String n)
      throws SQLException
{
 String dquery =("Delete from employees
          where name = " + n);
   s.executeUpdate (dquery);
 System.out.println ("Employee deleted !" + n);
 }
}
```

Q.10 Design a registration form for solo Dance Competition using swing and store the information into registration table

(kindley take 5 different Component)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class DemoRegform extends JFrame
{
 private JTextfield nemfield;
 private JComboBox <String> dancestyle;
 private JRadioButton, MaleRadioButton, femaleRadioButton
 private JTextfield Contact;
 private JButton registrationButton;

 public Dance Registrationform ()
 {
  setTitle ("Solo Dance Compition Registration");
```

```
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  setSize(400, 400);
  setLayout(new GridLayout(8,2);

JLable nLable = new JLable("Name");
  nameField = new JTextfield();

  JLable danceStyle = new JLable("Dance style");
String[] danceStyle = {"kathak", "Bharatnatyam",
                       "kuchipudi", "odissi", "kathkali"}
  danceStyle ComboBox = new JComboBox<>(danceStyle)

JLable genderLable = new JLable("Gender");
maleRadioButton = new JRadioButton("male");
femaleRadioButton = new JRadioButton("female");

ButtonGroup genderGroup = new ButtonGroup();
  genderGroup.add(maleRadioButton);
  genderGroup.add(femaleRadioButton);

  JLable contactLable = new JLable("contact");
Txtf. Contact = new JTextfield();

  registerButton = new JButton("Register");
  registerButton.addActionListener(new ActionListener
  {
   public void actionPerformed(ActionEvent e)
   {
     registerParticipent();
   }
  });
      add(nLable);
      add(nameField);
      add(danceStyle);
      add(danceStyle ComboBox);
```

```
add (genderlable);
add (maleRadioButton);
add (femaleRadioButton);
add (new JLable());
add (contactlable);
add (contact);
add (new JTlable());
add (registerButton);
setVisible (true);
}
private void registerParticipant()
{
System.out.println ("Name: " + nametield.getText());

System.out.println ("Style: " + danestyleComboBox.get
                                SelectedD);

System.out.println ("Gender:" + (maleRadioButton.isSelected()?
                                "Male": "Female"));

System.out.println ("Contact:" + Contact.getText());
}
public static void main (String args[])
{
SwingUtilities.invokelater (() -> new DanceRegform());
}
}
```