22/10/2020

OBJECT ORIENTED PROGRAMMING

M. Yuva Raja
19CSE063
913119104124
2nd yr/CSE/A

PART-A

(1)

Ans

- Object is an instant of a class. The objects represent real world entity The objects are used to provide a practical basis for the real world. Object can be declared by specifying the name of the class.
- Class is a collection of data and the function that manipulate the data. The data components of the class are called data fields and the function components are called member function. The class that contains main function is called main class.

(2)

Ans Access Specifiers (Visibility Specifiers) regulate access to classes, fields and methods in JAVA. It determines whether a field or method in class can be used

Scanned by CamScanner

or invoked by another method of subclass. It can be used to restrict access.

(3) It is the process of duplicating an object so that two identical objects will exist in the memory at the same time.

(4) Multithreading is a concept of programming concept where a program is divided into two or more subprograms, which can be implemented at the same time in parallel. A multithreaded program contains two or more parts that can run concurrently. Each part of such a program is called a thread. and each thread defined as a seperate path of execution.

(8) Code that uses generics has many benefits

Stronger type checks at compile time.

A Java compiler applies strong type checking to generic code and issues errors if the code violates type safety. Fixing compile-time errors is easier than fixing runtime errors,

Scanned by CamScanner

(9)

Ans The class WindowAdapter is an abstract (adapter) class for receiving window events. All methods of this class are empty. This class is convenience class for creating listener objects.

(10)

Ans Features of AWT in Java

- It is a set of native user interface components
- It is based upon a robust event handling model
- It provides graphics and imaging tools, such as shape, color and font classes
- Data transfer classes are also a part of AWT that helps in cut and paste through the native platform clipboard.

(4)

Ans Syntax:

import packagename;

eg: import java.util.*;

Scanned by CamScanner

(5)

Ans It is the superclass of those exception that can be thrown during the normal operation of the Java Virtual Machine.

RuntimeException and its subclasses are unchecked exceptions.

(6)

Ans

Input Stream	Reader
It is byte based	Reader is character based, it can be used to read or write characters
It can be used to read bytes or write bytes	
It is used to binary I/P and O/P	It is used to character I/P and O/P
Used to read binary files	Used for reading text files in platform default encoding
read() reads 1 byte (8 bit) at a time	read() reads 2 bytes (16-bit) at a time.

Scanned by CamScanner

PART-B

(11) (a)

i) Characteristics of JAVA

- **Simple:**

  It is easy to write and more reliable

- **Secure:**

  It cannot harm other System thus making it secure.

  Java provides secure way to access web-application.

- **Portable:**

  It can be run on any platform (Linux, Windo

- **Object-oriented:**

  It is object oriented language.

- **Robust:**

  Java encourages error-free programming by being strictly typed and performing runtime checks.

- **Dynamic:**

  Java programs carry with them substantial amounts of run-time type information that is used to verify and

Scanned by CamScanner

resolve accesses to object at run time.

- Distributed:

  Java can be transmit, run over internet.

- High performance:

  Bytecode are highly optimized.

- Multithreaded:

  It provides integrated support for multithreaded programming

- Interepted:

  Bytecode can be interepreted on any platform by JVM.

ii)

Packages:

Packages are java's way of grouping a variety of classes and interfaces together.

Packages are containers for the classes.

It is the header file in C++.

It is stored in hierarchical manner

Scanned by CamScanner

If we want to use the packages in a class, we want to impore it.

The two types of packages are

- System package
- User defined package

Uses of packages:

a) It reduce the complexity of the software
b) We can create classes with same name in different packages. Using packages we can hide classes.

Creating the package

To create our own packages

```
package first package; // package declaration
public class first class // class defination
{...
 (body of class)
 ...}
```

the file is saved as first class.java, located at first package directory. when it is compiled. class file will be created in same directory.

Scanned by CamScanner

Import package1[.package2][.package3].classname;
Using the package.

```
package package1;
public class classA
{
 public Void classA
 {
  public void displayA()
  {
 System.out.println("classA");
 }
 }
```

Adding a class to package.

```
Paeage P1;
Public class B
 Package name
  {
   //body of B
  }
Package college
 class student
 {
   int regno;
   String name;
  Student (int, String na);
    {
     Regno=r
      Name=na
    }
```

Scanned by CamScanner

```
Public void print()
  {
System.out.println("Regno" +regno);
System.out.println("Name" +name);
  }
  }
```

(12) (d)

Inner class:-

A non static class that is created inside a class but outside a method is called member inner class.

Syntax

```
class Outer{
 //code
  class Inner{
  //code
  }
  }
```

Java member inner class example.

In this eg: we are creating msg() method in member inner class that is accessing the private data of member of outer class

Scanned by CamScanner

```
class Test Member Outer 1 {
private int data = 30;
Class Inner {
Void msg() { System.out.println ("data is"
    + data); }
}
public static void main (String args []) {
Test Member Outer 1 obj = new Test Member
                          Outer 1 ();
Test Member Outer 1. Inner in = obj. new
                                Inner();
in. msg()
}
}
```

There are four types of inner classes:

1. member class.
2. Static member
3. local
4. anonymous.

Scanned by CamScanner

(*) member class - It is defined at the top level of the class. It may have the same access modifiers as variables (public, protected, final) and is accessed in much the same way as variables of the class

(*) Static member class - It is defined like a member class, but with the key word. Despite its position inside another class a static member class is actually an "outer" class it has no special access to names in its containing class.

(*) local inner class - It is defined within a method and the usual scope rules apply to it. It is only accessible within that method. ∴ access restriction (public, protected, package) do not apply.

(*) Anonymous inner class - which is declared and used to create one object, all within a single statement.

Scanned by CamScanner

(13) (b)

1) Exception is an Error that can happen during the execution of a program and disrupts its normal flow.

Java being an object oriented programming language, whenever an error occurs while executing a statement, creates an exception object and then the normal flow of the program halts.

when the exception occurs in a method, the process of creating the exception object and handling it over to runtime environment is called "throwing the exception".

Once runtime receives the exception object it tries to find the handler for the exception. Exception Handler is the block of code that can process the exception object.

The handler is said to be "catching the exception". If there are no appropriate exception handler found then program

Scanned by CamScanner

terminates printing information about the exception.

Java Exception handling is a framework that is used to handle runtime. Errors only. compile time Error are not handled by exception handling in Java.

ii) User defined exception in Java :-

```
class MyExceptionextends Exception {
String str1;
MyException (String str2)
 {
     str1=str2;
 }
public String to string() {
        return ("My Exception Occured:"+str1)
 }
}
class Example1{
public static void main (String args[]){
try {
     System.out.println ("Starting of try block
// I'm throwing the custom exception using
     throw
throw new MyException ("This is My Error");
```

Scanned by CamScanner

```
}
catch (MyExceptionexp){
    System.out.println("Catch Block");
    System.out.println(exp);
  }
 }
}.
```

(14) (a)

Inter thread communication in Java.

It is all about allowing synchronized threads to communicate with each other.

It is a mechanism in which a thread is paused running in its critical section and another thread is allowed to enter (or lock) in the same critical section to be executed.

It is implemented by following method of object class.

- wait()
- notify()
- notifyAll().

Scanned by CamScanner

Suspending Resuming and Stopping Threads

The following example illustrates how to wait() and notify() methods that are inherited from object can be used to control the execution of thread. The New Thread class contains a boolean instance variable named Suspend flag. which is used to control the Execution of the thread

```
Class New Thread implements Runnable
{
 String Name;
  Thread t;
  boolean Suspend flag;
  NewThread (String threadname)
  {
   name = threadname;
    t = new Thread (this, name);
    System.out.println("New Thread :" +t);
    Suspend flag = false;
    t.start();
  }
  public void run() {
   try {
```

Scanned by CamScanner

```
for(int i=15; i>0; i--)
{
System.out.println(name+":"+i);
Thread.sleep(200);
Synchronized(this).
{
while (suspend flag)
{
wait();
}
}
}
}
catch (InterruptedException e)
{
System.out.println(name+"interrupted");
}
System.out.println(name+"existing");
}
synchronized void mysuspend()
{
suspend flag = true;
}
synchronized void myresume()
{
```

Scanned by CamScanner

```
{
  Suspend flag = false;
   notify();
  }
}
public class Suspend Resume
  {
public static void main(String args[])
  {
   New Thread ob1 = new New Thread("One");
   New Thread ob2 = new New Thread("Two");

   try
    {
     Thread sleep(1000);
     ob1.my Suspend();
     System.out.println("Suspending Thread One");
     Thread.sleep(1000);
     ob1.myresume();
     System.out.println("Suspending Thred two");
     Thread.sleep(1000);
     ob2 my resume();
     System.out.println("Resume Thread two");
    }
```

Scanned by CamScanner

```
catch (InterruptedException e)
{
    System.out.println("Main thread
                        Interrupted");
    }
try
  {
    System.out.println("Waiting for
                        threads to finish.");
    ob1.t.join();
    ob2.t.join();
  }
catch (InterruptedException e)
  {
    System.out.println("Main thread
                        Interrupted");
  }
System.out.println("Main thread
                    Exiting.");
  }
}.
```

Scanned by CamScanner

(15) (a)

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
/*
* <applet code =
" Traffic lights Example" width = 1000
    height = 500>
* </applet>
**/

public class Traffic lights Example
extends Applet implements.
 ItemListener {

Check box Group grp = new
checkbox Group ();
Checkbox redlight, yellow light, green ligh
Lable msg;
public void init() {
   red light = new checkbox ("Red", grp
                          false);
   yellow light = new checkbox ("yellow",
                          grp, false);
```

Scanned by CamScanner

```
Checkbox ("Green", grp, false);
msg = new Label (" ");
redlight.addItemListener (this);
yellowlight.addItemListener (this);
greenlight.addItemListener (this);

add (redlight);
add (yellowlight);
add (greenlight);
add (msg);
msg.setFont (new Font ("Serif",
                Font.BOLD, 20));

}

Public void
  itemStateChanged (ItemEvent ie) {
      redlight.setForeground (Color.Black);
      yellowlight.setForeground (Color.Black);
      greenlight.setForeground (Color.Black);

    if (redlight.getState () == true)
    {
      redlight.setForeground (Color.RED);
        msg.setForeground (Color.RED);
        msg.setText ("STOP");
    }
```

Scanned by CamScanner

```
else if (yellowlight.getState() == true)
    {
    yellowlight.setForeground(Color.YELLOW);
    msg.setForeground(Color.YELLOW);
    msg.setText("READY");
    }
else{
greenlight.setForeground(Color.GREEN);
msg.setForeground(Color.GREEN);
    msg.setText("GO");
    }
}
}
```

Scanned by CamScanner