Assignment Questions 5

Q1.What is Exception in Java?

ans:-

1) Dictionary meaning of the exception is abnormal termination.

2) An exception is a problem occurred during execution time of the program.

3) An unwanted unexpected event that disturbs normal flow of execution called exception.

4) Exception is nothing but a object.

5) Exception is a class present in java.lang package.

6) All the exceptions are nothing but objects called classes.

7) Whenever user is entered invalid data then Exception is occur.

8) A file that needs to be opened can't found then Exception is occurred.

9) Exception is occurred when the network has disconnected at the middle of the communication.

Q2.What is Exception Handling?

ans:- It is recommended to handle the Exception the main of the Exception Handling is normal

Execution of the program or graceful termination of the program at runtime.

We are able to handle the exception in two ways.

1. By using try-catch blocks 2**. By using throws keyword**

Q3.What is the difference between Checked and Unchecked Exceptions and Error?

ans:-

checkedException:-

The Exceptions which are checked by the compiler at compilation time for the proper execution

of the program at runtime is called CheckedExceptions.

Ex:- IOException,SQLException etc…….

uncheckedException:-

The exceptions which are not checked by the compiler at compilation time is called

uncheckedException . These checking down at run time only.

Ex:- ArithmeticException,NullPointerException, etc……

Error:-

Errors are caused by lack of system resources . these are non recoverable.

Ex:- StackOverFlowError,OutOfMemoryError,AssertionError etc……

Goodpoint:-

The Exception whether it is checked or unchecked the Exceptions are occurred at runtime.

Q4.What are the difference between throw and throws in Java?

ans:-

Throw:-

1) The main purpose of the throw keyword is to creation of Exception object explicitly either for

predefined or user defined .

2) Throw keyword works like a try block. The difference is try block is automatically find the situation and creates a Exception object implicitly. Whereas throw keyword creates a Exception

object explicitly.

Throws :-

1) Throw keyword is used to create exception object explicitly. But the main purpose of the throws

keyword is bypassing the generated exception from present method to caller method.

2) Throw keyword is used in the method body. But throws keyword we have to use in the method

declaration.

3) It is possible to throws any number of exceptions at a time based on the programmer

requirement.

In the java language we are handling the Exception in two ways

1) By using try-catch blocks

2) By using throws keyword

Q5.What is multithreading in Java? mention its advantages

ans:-

1) The earlier days the computer's memory is occupied only one program after completion of one

program it is possible to execute another program is called uni programming.

2) Whenever one program execution is completed then only second program execution will be

started such type of execution is called co operative execution, this execution we are having lot

of disadvantages.

a. Most of the times memory will be wasted.

b. CPU utilization will be reduced because only program allow executing at a time.

c. The program queue is developed on the basis co operative execution

Advantages of multiprogramming:-

1. CPU utilization will be increased.

2. Execution speed will be increased and response time will be decreased.

3. CPU resources are not wasted.

Q6.Write a program to create and call a custom exception

ans:-In Java, we can create our own exceptions that are derived classes of the Exception class. Creating our own Exception is known as custom exception or

user-defined exception. Basically, Java custom exceptions are used to customize the exception according to user need.

```
class MyCustomException extends Exception
{


}

// class that uses custom exception MyCustomException
public class TestCustomException2
{
   // main method
   public static void main(String args[])
   {
      try
      {
         // throw an object of user defined exception
         throw new MyCustomException();
      }
      catch (MyCustomException ex)
      {
         System.out.println("Caught the exception");
         System.out.println(ex.getMessage());
      }

      System.out.println("rest of the code...");
```

```
    }
}
```

Q7.How can you handle exceptions in Java?

ans:-

It is recommended to handle the Exception the main of the Exception Handling is normal

Execution of the program or graceful termination of the program at runtime.

We are able to handle the exception in two ways.

1. By using try-catch blocks

2. By using throws keyword.

Exception handling by using Try –catch block:-

1) In java language we are handling the exceptions By using try and catch blocks. try block contains

risky code of the program and catch block contains handling code of the program.

2) Catch block code is a alternative code for Exceptional code. If the exception is raised the

alternative code is executed fine then rest of the code is executed normally.

Syntax:-

```
try
{
Code to run [break;]
}
Catch(ExceptionName reference_variable)
```

{

Code to run if an exception is raised

}

Q8.What is Thread in Java?

ans:-

Thread:-

1) Thread is nothing but separate path of sequential execution.

2) The independent execution technical name is called thread.

3) Whenever different parts of the program executed simultaneously that each and

every part is called thread.

4) The thread is light weight process because whenever we are creating thread it is not

occupying the separate memory it uses the same memory. Whenever the memory is

shared means it is not consuming more memory.

5) Executing more than one thread a time is called multithreading

Single threaded model:-

class Test

{ begins

public static void main(String[] args)

```
{
System.out.println("Hello World!");
System.out.println("hi rattaiah"); body
System.out.println("hello durgasoft");
}
} end
```

In the above program only one thread is available is called main thread to know the name of

the thread we have to execute the fallowing code.

Q9. What are the two ways of implementing thread in Java?

ans:-

There are two different ways to create a thread isavilable

1) Create class that extending standered java.lang.Thread Class

2) Create class that Implementing java.lang.Runnable interface

First approach to create thread extending Thread class:-

Step 1:-

Creates a class that is extend by Thread classes and override the run() method

```
class MyThread extends Thread
{
public void run()
{
System.out.println("business logic of the thread");
```

```
System.out.println("body of the thread");
}
};
```

Step 2:-

Create a Thread object

```
MyThread t=new MyThread();
```

Step 3:-

Starts the execution of a thread.

```
t.start();
```

In this approach take one user defined class class that is extending Thread class .

Ex:-

```
class MyThread extends Thread
{
public void run()
{
System.out.println("Rattaiah from durgasoft");
System.out.println("body of the thread");
}
};
class ThreadDemo
{
public static void main(String[] args)
{
```

```
MyThread t=new MyThread();
t.start();
}
}
```

Note :-

1) Whenever we are calling t.start() method the JVM search for the start() in the MyThread class

but the start() method is not present in the MyThread class so JVM goes to parent class called

Thread class and search for the start() method.

2) In the Thread class start() method is available hence JVM is executing start() method.

3) Whenever the thread class start() that start() is responsible person to call run() method.

4) Finally the run() automatically executed whenever we are calling start() method.

5) Whenever we are giving a chance to the Thread class start() method then only a new thread will

be created.

Second approach to create thread implementing Runnable interface:-

Step 1:-

Creates a class that implements Runnable interface.

```
class MyClass extends Runnable
{
public void run()
```

```
{
System.out.println("Rattaiah from durgasoft");
System.out.println("body of the thread");
}
};
```

Step 2:-

Creating a object.

```
MyClass obj=new MyClass();
```

Step 3:-

Creates a Thread class object.

```
Thread t=new Thread(obj);
```

Step 4:-

Starts the execution of a thread.

```
t.start()
```

implementing Runnable interface

```
class MyThread implements Runnable
{
public void run()
{
System.out.println("Rattaiah from durgasoft");
System.out.println("body of the thread");
}
```

```
}
class ThreadDemo
{
public static void main(String[] args)
{
MyClasss obj=new MyClass();
Thread t=new Thread(obj);
t.start();
}
}
```

Step 1:-

the Class MyClass implements the Runnable interface and overriding run() method and contains

the logic associates with the body of the thread.

Step 2:-

Creates the object of implementation class this is not like a first mechanism.

Step 3 :-

Creates a generic thread object then pass the MyClass reference variable as a parameter to that

object.

Step 4:-

As a result of third step 3 a thread object is created in order to execute this thread method we

need to class start() method. Then new thread is executed.

We are having two approaches:-

First approach:-

1) By extending the thread class, the derived class itself is a thread object and it gains full

control over the thread life cycle.

2) Another important point is that when extending the Thread class, the sub class cannot extend

any other base classes because Java allows only single inheritance.

if the program needs a full control over the thread life cycle, then extending the Thread class

is a good choice.

Second approach:-

1) Implementing the Runnable interface does not give developers any control over the thread

itself, as it simply defines the unit of work that will be executed in a thread.

2) By implementing the Runnable interface, the class can still extend other base classes if

necessary.

if the program needs more flexibility of extending other base classes, implementing the

Runnable interface would be preferable.

We are having two approaches two create a thread use any approach based on application requirement.

Q10.What do you mean by garbage collection?

ans:-Garbage collection (GC) is a memory recovery feature built into programming languages such as Java. A GC-enabled programming language includes one or more garbage collectors (GC engines) that automatically free up memory space that has been allocated to objects no longer needed by the program. The reclaimed memory space can then be used for future object allocations within that program