Pass Marks: 45
Time : 3hrs.

Ch3 Exception, stream & I/O (6)

(3.1) Handling the errors of exception, catching exception, tips on handling exception, Debugging technique

(3.2) Stream, zipfile stream, object stream

(3.3) Handling files.

Exception can be defined as the any kind of abnormal errors or behaviour seen in our program that will halt the normal flow of program and hence need to be handled properly.

While running the program, there is the possibility of getting errors, which must be trapped, provide the proper info of those errors and handled in order to run the program smoothly. This whole process is known to be exception Handling.

The mechanism for handling the exceptions is embedded in try catch and finally statement construct that have the following form:

```
try{
//statement or body
}catch(exception e)
{
  // handling exception
}finally
{ }
```

(A)

Here exception occurs during the execution of try block, can be caught and handled in the catch block. Similarly the statement inside of the finally block will be executed in any situation whether the exception is handled or not, no matters. Also the finally statement usually consists of things like closing DB connection, Resource cleanup, flushing, cleaning up temporary data etc. All the exceptions in java are derived from java.lang.Throwable class. The Throwable class is the root class of the exception and error in java.

Hierarchy of Throwable class is:

java.lang (package)

Throwable (class)

- Exception
  - (A) → IO Exception (Related with file)
  - (B) → SQL Exception (Related with DB)
  - (C) → ClassNotFoundException
  - (D) → RuntimeException
    - → ArithmeticException
    - → NullPointerException
    - → NumberFormatException
- Error
  - → VirtualMachine Error
  - → StackOverflow Error
  - → OutOfMemoryError

ster: Fall
Year : 2021
Full Marks: 100
Pass Marks: 45
Time : 3hrs.

e their answers in their own words as far

(8)

(5) IndexedOutOfBoundException
- (a) ArrayIndexedOutOfBoundException
- (b) StringIndexOutOfBoundException

Types of exception in Java
(a) Checked & unchecked exception
(b)
- User defined Exception
- Built in Exception
  - Checked
  - unchecked

Built in exception are the exception which are already defined in java libraries, which are able to define the error situation and give idea about what, how and why the error is occured. It is broadly categorised into 2 types:
(a) Checked exception
(b) Unchecked exception.

Those exception which are checked at compile time by the compiler is known to be checked exception. It is also known to be compile time exception. Compiler ensures that programmer have to handle the exception and if not handled, compilation error will occurs.

Those exception which are not checked by the compiler at runtime is known to be unchecked exception. That means if a program throws an unchecked exception and even if we don't handle it, compilation job will be done. Usually it occurs, when

user provides improper data to the program while giving i/p

User defined Exception

They can be defined as the type of exception that can be written by user after extending the exception class. We can throw own exception at particular condition.

The simple snippet of custome exception is:

```
class DemoCustomException extends Exception
{
    //overriding the methods like getMessage(),
      getString() or simply printing stackTrace.
}
```

The exception class which will be extended during userdefined exception have following 3 methods:

String toString(): ⇒ get executed when Sout(e) runs
String getMessage(); ⇒ prints the exception message
void printStackTrace(); ⇒ prints the Stack Trace.

A UNIVERSITY

mester: Fall

Year : 2021
Full Marks: 100
Pass Marks: 45
Time : 3hrs.

ve their answer ... own words as far

(19)

```
import java.lang.*;
import java.util.*;
class DemoException extends Exception
{
    @Override
    public String toString()
    {
        return "I am toString() method";
    }
    @Override
    public void getMessage()
    {
        return "I'm getMessage() method";
    }
}
public class DriverClass {

    public static void main (String[] args)
    {
        DemoException de= new();
        Scanner sc= new Scanner(System.in);
        System.out.println("enter a value : ");
        int a= sc.nextInt();

        if(a<55)
        {
            try{
                throw new DemoException();
            }catch(Exception e)
            {
                System.out.println("e.getMessage());
                System.out.println("e.getString());
```

Date
Page

e.print
System.

}

{

STrea

STrea
Input
Provi
Gener

Based
@ Byt

ByteSt
read
the c
Chara
used
It

```
e.printStackTrace();
System.out.println(e); // invokes getString() method
}
}
}
```

Stream , Zip file stream , Object stream

↳ Stream can be defined as the representation of an input source or an output destination. Java provides the IO streams to read & write data. Generally stream is categorized in

ⓐ Input stream : channel to read data from an input source

ⓑ Output stream : channel through which data and info will be written in a file.

Based on the data they handle, its types are:
ⓐ Byte stream ⓑ character stream.

ByteStream handles the data in byte (8 bits) i.e it can read and write data of 8 bit. InputStream & OutputStream are the classes used to read / write data. eg: Image, audio, video etc are stored, handled.

Character Stream:
↳ used to handle character data
↳ It can be used to read & write text data only.

ARA UNIVERSITY

Semester: Fall | Year: 2021 | Full Marks: 100 | Pass Marks: 45 | Time: 3hrs

ava

to give their answers in their own words as far

n indicate

File Handling

File Handling is the process of reading data from a file & writing data into a file. It involves reading & writing data from and into a file, updating the content of file with the helps of different package using java.io package

Some of the classes used for file handling is:

(a) File: To print the file and get info about file

(b) FileReader: To read the character data from file

(c) BufferedReader: To read character data into buffer

(d) FileInputStream

⇒ To open the file in i/p mode & read the content of file in byte format.

(e) FileOutputStream

↳ To open the file in o/p mode and write the content of file in byte formate

(f) PrintWriter:

QBank

22 fall 25

⇒ Impo

publ

{

QBank (13)

22fail25

=> import java.io.*;

public class TTFail1100B
{
public static void main (String [] args)
{ // writing to a file
try
{
FileOutputStream fos = new FileOutputStream
("javaExam.jsp");
String data = "Hello world";

//Array of type byte
byte[] bytes = ~~byte~~ data.getBytes();
fos.write(bytes); //writing to a file

System.out.println("Data is written to a file");

fos.close();
} catch(exception e)
{
e.printStackTrace();
}

//Reading from a file in byteformat.
try
{
FileInputStream fis = new FileInputStream
("javaExam.jsp");
int byteRead;

HARA UNIVERSITY

Semester: Fall

Year : 2021
Full Marks: 100
Pass Marks: 45
Time : 3hrs.

Java

d to give their answers in their own words as far

in indicate full marks.

ns.

(iv)

```
while((byteRead = fis.read()) != -1)
    {
      System.out.println((char)byteRead);
    }
} catch(Exception e)
    {
      e.printStackTrace();
    }
  }
}
```

}
}

File Reader and File writer

↳ FileReader and FileWriter are the classes that provides a way to read and write from/on a file:
character data

(a) FileReader: This class is used to read character data from a file. It reads the data as a character and can be used to read text files.

(b) FileWriter: This class is used to write character data to a file. It write the data as a character and used to create or append to a textfile.

Buffered Reader & Buffered Writer

↳ BufferedReader and BufferedWriter are classes that provide buffering for character input and o/p operations, which can significantly improve i/o performance.

efficien

So the core c
BufferedRe
(a) FileReader
from & to
used of
underlyi
initial

21fall39

↳ class custo
{

(a) BufferedReader: This class wraps up the character based input stream like FileReader and provides a buffer for reading characters from the underlying stream more efficiently.

(b) BufferedWriter: This class wraps up the character based o/p stream like FileWriter and provides a buffere for writing character from the underlying stream more efficiently.

Sr the core difference in FileReader, FileWriter vs BufferedReader, BufferedWriter is:

(a) FileReader, FileWriter reads and write character data from & to a file directly without the helps of any kind of buffer whereas its counterpart provider a underlying buffer system that wraps up the initial class for faster I/O operation.

21fall39: creating own exception class.

```
class customException extends Exception
{
    public void getMessage()
    {
        System.out.println("Provided amount is lesser
        than demanded amount");
    }

    public void toString()
    {
        System.out.println("Amount not sufficient");
```

}

}

(10)

```
public class driverclass
{
    public static void main (String[] args)
    {
        Scanner sc= new Scanner(System.in);
        System.out.print ("enter the amount : ");
        int amount = sc.nextInt();
        if (amount < 150)
        {
            try{
                throw new customException();
            }catch (Exception e)
            {
                e.printStackTrace();
            }
        }else
        {
            System.out.println("Your purchase bill is
                ready sir, Happy purchase");
        }
    }
}
```

19.Sept.24

Q. What is Exception? How exception handling is different from error handling explain.

Ans. Exception can be defined as the abnormal disruption that comes in the program that is capable of halting the program and hence need to be handled properly with care.

When we run the program, there is possible chances that some issue may come in program ~~[illegible]~~, which must be trapped, provide the proper information about those issue and need to be resolved, by the proper use of differe classes that java provides. This whole process is known to be exception handling.
Exception allows developers to handle the abnormal condition in a well structured manner ~~maintaining~~ preventing the program to crash.

Exception handling is way more different from error handling. They are the branching from Throwable class. In the exception hierarchy there are multiple exceptions like:- IOException, SQLException, NoClassFoundException etc whearas Errors are like: VirtualMemoryError, StackOverflowError etc.

The exception handling is the mechanism of handling the exception with the use of try catch and finally construct. It helps us to think about the potential bugs or issue that can come in our program and handle them properly. The main workflow is any kind of snippet or code where exceptions may come

is wrapped up by try block and if any exception occurs it will be caught by catch block and handles the exception. Also finally construct is used to execute the code which must be execute any how. Usually, DB connection closing, closing resources, cleaning and releasing is done here. (19)

```
try
{
    int result = 10/0;
    Sout(result);
} catch (ArithmeticException e)
{
    Sout(e);
}finally
{
    Sout("Here necessary operation is done");
}
```

On the other hand error handling refers to dealing with several issues that are typically beyond the control of program itself. It is often related to JVM or environment of system. It involves error like: StackOverflow error, VirtualMemoryError etc. Unlike exception errors are not expected to caught and handled by program itself. It is the issue which are not recoverable. So instead of attempt to handle those errors with in a program, it's wise decision to terminate a program

Date
Page

Exception are the fault that can occurs in program and hence need to be handle properly for smooth running of program known to be exception i.e. while where Error are the issue that comes mostly in physical case which is impossible to handle in program itself.

Short note

18 sp 7b C++ vs Java

⇒ C++ is a multiparadigm programming language that supports procedural oriented, object oriented and generic programming whereas Java is ~~pure~~ object oriented programming language with a strong support of class and object, interfaces.

⇒ C++ supports manual memory management through pointers and new/delete operators. It also supports automatic memory management through using destructors and constructors. Whereas Java supports automatic memory management with the help of automatic garbage collectors.

⇒ C++ code is compiled into machine code making it platform dependent. It needs to be recompiled for every individual platform whereas Java code is compiled to bytecode and taken by JVM, that makes it platform independent. The bytecode generated in windows machine can be ran in mac os with out recompilation.

⇒ C++ uses try, catch, throw and exception keyword for exception handling. But Java use try, catch, finally, throw keyword for exception handling.

⇒ C++ is mostly used for low level system programming, in performance oriented apps where though java being fast, its mostly used for enterprise app dev.

(18 fall 7th) History of Java

Java is a platform independent, highly robust, secure programming language developed by Sun microsystem in 1995. It is object oriented programming language, providing strong support for object, class and interface.
James Gosling is known to be father of Java, which was initially known to be 'Oak'. Later on oracle bought the Sun microsystem that make s oracle as the present owner of Java. It supports WORA that stands for write once run anywhere that means java code is compiled to byte code that can run on any machine either in windows, mac or linux etc without recompiling until the o.s have JDK. JVM is the one that makes Java, a platform independent and secure programming language, having feature like automatic garbage collection. Java is evolution continued through various versions with java SE, EE and ME which stands for Java SE Edition, Java Enterprise Edition & Java micro Edition. Now a day in every six month Java is releasing updates.

Java project came into mind of Gosling and his team when they face problem with embeded system SW used for consumed based electronic appliances. They wann get rid of writing individual SW for each device. They were aiming to create a prog lang and platform that could run

smart appliances which can communicate with each other irrespective of the coe they were running. This project is known to be green project which later on named as 'Oak' based on oak tree outside of James's office. Later on due to naming restriction, it converted to Java.

18 feb 25 Program to handle Arithmetic Exception.

```
import java.lang.*;

public class DriverClass
{
    public static void main(String[] args)
    {
        int dividend = 10;
        int divisor = 0;
        try
        {
            int result = dividend / divisor;
            System.out.println("result : " + result);
        } catch (ArithmeticException e)
        {
            System.out.println("Error: Division by Zero");
        }
        System.out.println("The demo of Arithmetic
            exception handling is shown");
    }
}
```

Q 17fall39 wap to store object of class Student into file "student.dat". also read the object from same file and display the state of obj on the screen. Handle possible exceptions.

```
import java.io.*;

class Student implement Serializable
{
  String name;
  ~~int rollnumber~~
  int age;
  public Student(String name, int age)
  {
    this.name = name;
    this.age = age;
  }
  public String toString()
  {
    return "Name: " + Name + ", ~~roll number:~~ age
        + age";
  }

} //Driver Code
public DriverClass {
  public static void main()
  {
    // creating student class object
    Student s1 = new Student("John", 22);
    Student s2 = new Student("Steven", 32);

    //saving the object of Student class into a file
```

```
try
{
 ObjectOutputStream oos= new ObjectOutputStream(new
                         FileOutputStream("Student.dat"));
 oos.writeObject(s1);
 oos.writeObject(s2);

 System.out.println("Student object is stored in student.dat
                     file");
}catch(Exception e)
 {
  System.out.println("An error is stored while saving."
                     + e.getMessage());
 }

// reading the object from file

try
{
 ObjectInputStream ois= new ObjectInputStream(new
      FileInputStream("student.dat"));

 System.out.println("Loaded objects");

 System.out.println("ois.readObject());
  System.out.println(ois.readObject());
} catch (IOException | ClassNotFoundException e)
 { System.out.println("An error occured while reading
      object : " + e.getMessage());
}
```

} //main method
} //Driver class

16 fallth

checked exception	unchecked exception
↳ These kind of exception that lies under built in exception, and is checked by the compiler during compilation is known to be checked exception.	↳ Those kind of exception that lies under built in exception but isn't checked at compilation time by compiler is known to be unchecked exception.
↳ At check exception, if any exception occurs, until and unless it is handled it will move ahead.	↳ At unchecked exception, even if the exception is thrown and if we don't handle it, compilation job will be done.
Eg:- IOException, SQLException etc.	Eg:- ArrayIndexOutOfBound exception, NullPointer Exception, ArithmeticException, or any kind of custom exception.
checked exception occurs at compile time	Unchecked exception occurs in runtime.
These exception are the subclass of Exception class.	They are runtime so not part of Exception class.
JVM needs to catch the exception & handle.	Jvm doesn't catch the exception instead program itself find and handle it.

Date
Page

JIT compile

↳ JIT which stands for just-in-time compilation is technique used in computer programming and run time environment to improve the performance of program. Here in this method, instead of translating whole programs source code into machine code during compile time run time (execution), JIT compiles block of code to machine code at runtime befor it execution. This method combine advantage of interpretation + ahead-of-time compilation.

Some of the popular programming language that use JIT compilation method is:

(a) Dart: Features of flutter, dart programming like: hot reload, hot restart use the advantage of JIT compilation, to see any kind of changes done while developing apps ~~while~~ without starting the entire apps.

(b) Python: Though python being an interpreted language, it uses JIT compilation implementations like PyPy which use JIT compiller to improve execution speed.

(c) Java: Java runs on JVM, that uses JIT compilation extensively to compile most frequently executed portion of bytecode.

(d) Javascript: Modern JS Engines like: Google's V8 & Mozilla's SpiderMonkey uses JIT compilation to translate

JS code to machine code for faster execution.

(78)

14 S 25 Why do we need to serialize an object?
WAP to read a number from the console using InputStream. Also apply the method of object serialization in your program.

Soln

Serialization in java refers to the process of converting an object state into a format that can be easily stored, transmitted or reconstructed later. This format can be sequence of bytes, files etc. The main purpose of serialization is to make the object in that format such that it can be transported or stored easily

14 fall 25

```
import java.util.*;
public userInput
{
    public static void main(String[] args)
    {
        try
        { Sout("enter an integer: ");
          Scanner sc = new Scanner(System.in);
          int userdata = sc.nextInt();
          System.out.println("provided integer: "+
                  userdata);
        } catch(java.util.InputMismatchException e)
        {
          Sout("error: input is not an integer");
        }
```

Date
Page

?

?