**Assignment 4**

1. Write a java program to create an user defined exception called PayOutOfBoundsException. This exception is thrown when basicpay is not in between 10000 and 30000.

import java.util.\*;

class PayOutOfBoundsException extends Exception

{

int paymny=0;

PayOutOfBoundsException(int k)

{

paymny=k;

}

public String toString()

{

return "Money Not in Range";

}

}

class UserDefineException

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

try{

int basicpay=sc.nextInt();

if(basicpay<10000|| basicpay>30000) throw new PayOutOfBoundsException(basicpay);

{

System.out.println("This is in range");

}

}

catch(PayOutOfBoundsException m)

{

System.out.println(m);

}

}

}

/\*17000

This is in range

Press any key to continue . . .

200

Money Not in Range

Press any key to continue . . .\*/

2.Write a java program to create two threads which display a message every half second.

class THread extends Thread

{

public void run()

{

try

{

int i=0;

for(i=0;i<10;i++)

{

Thread.sleep(500);

System.out.println("Thread 1");

}

}

catch (InterruptedException e)

{

}

}

}

class THread2 extends Thread

{

public void run()

{

try

{

int j;

for(j=0;j<10;j++)

{

Thread.sleep(500);

System.out.println("Thread 2");

}

}

catch (InterruptedException e)

{

}

}

}

class Program2

{

public static void main(String[]args)

{

THread t=new THread();

t.start();

THread2 t1=new THread2();

t1.start();

}

}

OUTPUT:

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Thread 1 Right

Thread 2 Stroke

Press any key to continue . . .

3. Write a java program to implement interthread communication.

class Comm

{

int amt;

synchronized void withdraw(int amt)

{

if(this.amt<amt)

{

System.out.println("Amount is not sufficent ");

try

{

wait();

}

catch(Exception e)

{

}

}

this.amt-=amt;

System.out.println("With drawn completed");

}

synchronized void deposite(int amt)

{

System.out.println("Going to deposited ");

this.amt+=amt;

System.out.println("Deposited ");

notify();

}

}

class InterCommunicationofThreads

{

public static void main(String[] args)

{

Comm b1=new Comm();

new Thread(){

public void run(){

b1.withdraw(7000);

}}.start();

new Thread()

{

public void run(){

b1.deposite(10000);

}}.start();

}

}

OUTPUT:

/\*

Amount is not sufficent

Going to deposited

Deposited

With drawn completed

Press any key to continue . . .\*/

4.Write a java program to implement Thread Synchronization.

class sync

{

synchronized void printTable(int n)

{

for(int i=1;i<=5;i++)

{

System.out.print(n\*i+" ");

}

}

}

class Mythread1 extends Thread

{

sync t;

Mythread1(sync t)

{

this.t=t;

}

public void run()

{

t.printTable(5);

}

}

class Mythread2 extends Thread

{

sync t;

Mythread2(sync t)

{

this.t =t;

}

public void run()

{

t.printTable(100);

}

}

class thread\_ex

{

public static void main(String[] args)

{

sync t=new sync();

Mythread1 t1=new Mythread1(t);

Mythread2 t2=new Mythread2(t);

t1.start();

t2.start();

}

}

/\*

5 10 15 20 25 100 200 300 400 500 Press any key to continue . . .\*/

5.Write a java program to implement Generic Class,Generic Method and Generic Constructor.

class GenericConstructor<T>

{

T percentage;

GenericConstructor(T percentage)

{

this.percentage=percentage;

}

public void display()

{

System.out.println("The percentage is "+this.percentage);

}

public <E> void find(E[] elements)

{

for(E e:elements)

System.out.println(e);

}

}

class GenConstructor

{

public static void main(String[] args)

{

GenericConstructor<Float> g1=new GenericConstructor<Float>(98.7f);

g1.display();

GenericConstructor<String> g2=new GenericConstructor<String>("PASS");

g2.display();

GenericConstructor<Integer> g3=new GenericConstructor<Integer>(90);

g3.display();

Integer a[]={1,2,3};

String b[]={"Right","Stroke","GIET"};

g1.find(a);

g1.find(b);

}

}

OUTPUT:

The percentage is 98.7

The percentage is PASS

The percentage is 90

1

2

3

Right

Stroke

GIET

Press any key to continue . . .

6.Write a java program to count no of vowels in a given file.

import java.io.\*;

import java.util.\*;

class CountVwInFile

{

static int find(char s[])

{

int vowels = 0;

for(int i = 0; i < s.length; ++i)

{

char ch = s[i];

if(ch == 'a' || ch == 'e' || ch == 'i'

|| ch == 'o' || ch == 'u' || ch == 'A' || ch == 'E' || ch == 'I'

|| ch == 'O' || ch == 'U') {

++vowels;

}

}

return vowels;

}

public static void main(String[] args) throws IOException

{

int c=0;

FileReader fr=new FileReader("D:\\java\\Filewriter.txt");

ArrayList<Character> a=new ArrayList<>();

int k=0;

while((k=fr.read())!=-1)

{

a.add((char)k);

}

char c1[]=new char[a.size()];

for(int i=0;i<c1.length;i++)

c1[i]=a.get(i);

System.out.println("The count of vowels is "+find(c1));

}

}

/\*

The count of vowels is 14

Press any key to continue . . .

Filewriter.txt

This is nagendra and character oriented and\*/

7. Write a java program to implement autoboxing and unboxing.

import java.util.\*;

class auto\_un

{

public static void main(String[] args)

{

int a=50;

Integer a2=new Integer(a);// AutoBoxing

System.out.println("Autoboxing "+a2);

Integer i=new Integer(200);

int b=i; //Unboxing

System.out.println("Unboxing "+b);

}

}

OUTPUT:

Autoboxing 50

Unboxing 200

Press any key to continue . . .

8.Write a java program to copy a file.

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class Copyfile

{

public static void main(String[] args) throws Exception

{

FileInputStream instream = null;

FileOutputStream outstream = null;

try{

File infile =new File("D:\\Java\\Filewriter.txt");

File outfile =new File("D:\\Java\\CopyFile.txt");

instream = new FileInputStream(infile);

outstream = new FileOutputStream(outfile);

byte[] buffer = new byte[1024];

int length;

while ((length = instream.read(buffer)) > 0){

outstream.write(buffer, 0, length);

}

instream.close();

outstream.close();

System.out.println("File copied successfully!!");

}catch(IOException ioe){

ioe.printStackTrace();

}

}

}

OUTPUT:

/\*

File copied successfully!!

Press any key to continue . . .

CopyFile.txt

This is nagendra and character oriented and\*/

9.Write a java program to implement Stack using Generic class.

import java.util.\*;

public class GenericStack <T> {

private ArrayList<T> stack = new ArrayList<T> ();

private int top = 0;

public int size () { return top; }

public void push (T item) {

stack.add (top++, item);

}

public T pop () {

return stack.remove (--top);

}

public static void main (String[] args) {

GenericStack<Integer> s = new GenericStack<Integer> ();

s.push (5);

s.push(100);

int s1=s.size();

System.out.println("size of stack: "+s1);

int i = s.pop ();

System.out.println("top element: "+i);

}

}

OUTPUT:

/\*

size of stack: 2

top element: 100

Press any key to continue . . .\*/

10.Write java program to swap two values using generic method.

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class GenericSwap {

public static final <T> void swap (T[] a, int i, int j) {

T t = a[i];

a[i] = a[j];

a[j] = t;

}

public static final <T> void swap (List<T> l, int i, int j) {

Collections.<T>swap(l, i, j);

}

private static void test() {

String [] a = {"Right", "Stroke"};

System.out.println("original array : "+Arrays.toString(a));

swap(a, 0, 1);

System.out.println("Swap using array : "+Arrays.toString(a));

List<String> l = new ArrayList<String>(Arrays.asList(a));

swap(l, 0, 1);

System.out.println("Swap again Using list : "+l);

}

public static void main(String args[])

{

test();

}

}

OUTPUT:

/\*

original array : [Right, Stroke]

Swap using array : [Stroke, Right]

Swap again Using list : [Right, Stroke]

Press any key to continue . . .\*/

Theory

1. What is thread?

Answer:

A multithreaded program contains two or more parts that can run concurrently.Each part of such a program is called thread,and each thread defines a separate path of execution.

2. Write the difference between multithreading and multitasking

Answer:

Multithreading

1.It is a process of executing multiple threads(sub-process).

2.Thread is basically a lightweight sub-process.

3.In multithreading, processes are allocated same memory.

4.multithreading component does not involve multiprocessing

Multitasking

1.It is a process of executing many processes running simultaneously.

2.It is basically a heavy weight process.

3.In multitasking, the processes share separate memory.

.4.Multitasking component involves multiprocessing.

3. What is Enumeration?

Answer:An Enumeration is a list of named contants.In java ,an enumeration defines a class type. An enumeration can have constructors,methods,and instance variables.

4. What is autoboxing?

Answer: Autoboxing is the automatic conversion that the java compiler makes between the primitive types and their corresponding object wrapper classes.

For example,converting an int to an Integer, a double to a Double..

5. What is wrapper class?

Answer:A wrapper class is a class whose object wraps or contains a primitive datatypes.The wrapper class in java provides the mechanism to convert primitive into object and object into primitive.

6. what is transient modifier?

Answer: Transient in Java is used to indicate that a field should not be part of the serialization process.The modifier Transient can be applied to member variables of a class to turn off serialization on these member variables. Every field that is marked as transient will not be serialized.

7. What is Generic class?Write the syntax of generic class.

Answer:A generic type is a generic class or interface that is parameterized over types.We use angle brackets (<>) to specify the type parameter.

Syntax:

public class ClassName<T>{//Class Members….}

8. What is stream?

Answer: A stream is a sequence of objects that supports various methods which can be pipelined to produce the desired result. A stream is not a data structure instead it takes input from the Collections, Arrays or I/O channels.

9.What is predefined stream?

Answer: java provides three predefined stream objects: in, out, and err, defined in the System class of the java.lang package. The out object refers to the standard outpu stream or console. The in object refers to standard input, which is the keyboard. And,

the err object refers to a standard error, which again is nothing but the console.

10.What is multithreading?

Answer: Multithreading is a process of executing multiple threads simultaneously.

Multithreading is used to obtain the multitasking. It consumes less memory and gives the fast and efficient performance.

11. What is the use of toString()?

Answer: The toString method is used to return a string representation of an object. If any object is printed, the toString() method is internally invoked by the java compiler. Else, the user implemented or overridden toString() method is called.

12.What is deadlock?

Answer: Deadlock in Java is a condition where two or more threads are blocked forever, waiting for each other. This usually happens when multiple threads need the same locks but obtain them in different orders.

13. Write inter thread communication methods.

Answer: Inter-thread communication 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 methods of Object class:

O wait()

O notify()

O notifyAll()

14. Write the difference between Checked and Unchecked exception.

Answer:

|  |  |
| --- | --- |
| Checked Exception | Unchecked Exception |
| 1.The exceptions which are checked by the compiler for smooth execution of the program at runtime are called Checked Exception | 1.The exceptions which are not checked by the compiler are called Unchecked Exceptions |
| 2.If these are not handled properly, they will give compile time error | 2.If these exceptions are not handled properly, they don’t give compile time error. But the program will be terminated prematurely at runtime |
| 3.These occur at compile time | 3.These occur at runtime |
| 4.All subclasses of Exception class except RuntimeException have Checked Exception | 4.All subclasses of RunTimeException and Error are unchecked exceptions |

15. What is thread synchronization?

Answer:When we start two or more threads within a program, there may be a situation when multiple threads try to access the same resource and finally they can produce unforeseen result due to concurrency issues.

So there is a need to synchronize the action of multiple threads and make sure that only one thread can access the resource at a given point in time. This is implemented using a concept called monitors.