**Assignment 1:**

1. **Armstrong Number:**

**import** java.util.\*;

**public** **class** Armstrong {

**public** **static** **void** main(String[] args) {

**int** a;

**int** temp,rem = 0,sum=0;

System.***out***.println("enter the number");

Scanner sc=**new** Scanner(System.***in***);

a=sc.nextInt();

temp=a;

**while**(temp!=0) {

rem=temp%10;

sum=sum+(rem\*rem\*rem);

temp=temp/10;

}

**if**(sum==a)

System.***out***.println("given number is a armstrong number");

**else**

System.***out***.println("given number is not a armstrong number");

}

}

**Output**:

enter the number

153

given number is a armstrong number

1. **Armstrong Number ranging between 100-999:**

**package armstrong;**

**public class armstrongranging {**

**public static void main(String[] args) {**

**for(int i=100;i<=999;i++)**

**{**

**int n=i;**

**int r=0,s=0;**

**while(n>0)**

**{**

**r=n%10;**

**s=s+(r\*r\*r);**

**n=n/10;**

**}**

**if(i==s)**

**{**

**System.*out*.println(i + " is a armstrong number");**

**}**

**}**

**}**

**}**

**Output:**

153 is a armstrong number

370 is a armstrong number

371 is a armstrong number

407 is a armstrong number

1. **Simple interest and compound interest:**

**import java.util .\*;**

**public class simpleandcompound interest**

**{**

**public static void main(String[] args) {**

**double p, rate, t, si, ci;**

**System.*out*.print("Enter The Amount : ");**

**Scanner read = new Scanner(System. *in*);**

**p = read.nextDouble();**

**System. *out*. print("Enter The No. of Years : ");**

**t = read.nextDouble();**

**System. *out*. print("Enter The Rate of Interest : ");**

**rate = read.nextDouble();**

**si = (p \* t \* rate)/100;**

**ci = p\*Math.*pow*(1.0+rate/100.0,t)-p;**

**System.*out*.println("Simple Interest = "+si);**

**System.*out*. println("Compound Interest = "+ci);**

**}**

**}**

**Output:**

**Enter The Amount : 2000**

**Enter The No. of Years : 2**

**Enter The Rate of Interest : 4**

**Simple Interest = 160.0**

**Compound Interest = 163.20000000000027**

**4)SupplyResults:**

**import java.util.\*;**

**public class SupplyResult {**

**@SuppressWarnings("resource")**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**int a,b,c;**

**Scanner sc=new Scanner(System.*in*);**

**System.*out*.println("enter marks of a: ");**

**a=sc.nextInt();**

**System.*out*.println("enter marks of b: ");**

**b=sc.nextInt();**

**System.*out*.println("enter marks of c: ");**

**c=sc.nextInt();**

**if(a>60 && b>60 && c>60)**

**{**

**System.*out*.println("Passed");**

**}**

**else if(a>60 && b>60 || b>60 && c>60 || a>60 && c>60)**

**{**

**System.*out*.println("Promoted");**

**}**

**else if(a>60 || b>60 || c>60 || a<60 && b<60 && c<60)**

**{**

**System.*out*.println("Failed");**

**}**

**}**

**}**

**Output:**

**enter marks of a:**

**60**

**enter marks of b:**

**77**

**enter marks of c:**

**88**

**Promoted**

**5) Income Tax:**

**Program:**

**package incometax;**

**import java.util.\*;**

**public class IncomeTax {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**double tax=0;**

**int nil=0;**

**double it=0;**

**System.*out*.println("enter the income: ");**

**Scanner sc=new Scanner(System.*in*);**

**it=sc.nextDouble();**

**if(it>=0 && it<=180000)**

**{**

**tax=nil;**

**}**

**else if(it>=181000 && it<=300000)**

**{**

**tax=it\*10/100;**

**}**

**else if(it>=300001 && it<=500000)**

**{**

**tax=it\*20/100;**

**}**

**else if(it>=500000 && it<=1000000)**

**{**

**tax=it\*30/100;**

**}**

**System.*out*.println("tax amount is " + tax);**

**}**

**}**

**Output:**

**enter the income:**

**243213**

**tax amount is 24321.3**

**6)LoginPassword:**

**import java.util.Scanner;**

**public class LoginPassword {**

**public static void main(String[] args) {**

**Scanner sc = new Scanner(System.*in*);**

**String sp = " ";**

**System.*out*.println("Enter the login name");**

**String username = sc.nextLine();**

**System.*out*.println("Enter the Password");**

**String password = sc.nextLine();**

**System.*out*.println(username + " you are Registered Successfully");**

**System.*out*.println("Enter the Username");**

**String name = sc.nextLine();**

**System.*out*.println("Enter the Password");**

**String passwordOne = sc.nextLine();**

**if (username.equals(name) && password.equals( passwordOne)) {**

**System.*out*.println("Welcome " + name + " you have Logged-in Successfully");**

**} else {**

**System.*out*.println("Username or password Mismatch");**

**}**

**int totalAttempts = 3;**

**while (totalAttempts != 0) {**

**if (name == "temp" && passwordOne == "pass") {**

**System.*out*.println("Login Correct!");**

**return;**

**} else {**

**System.*out*.println("Incorrect Login");**

**totalAttempts--;**

**System.*out*.println(totalAttempts);**

**}**

**}**

**if (totalAttempts == 0) {**

**System.*out*.println("Maximum number of attempts exceeded,,CONTACT ADMIN");**

**}**

**}**

**}**

**Output:**

**Enter the login name**

**nikitha**

**Enter the Password**

**nikki**

**nikitha you are Registered Successfully**

**Enter the Username**

**nikitha**

**Enter the Password**

**nikki**

**Welcome nikitha you have Logged-in Successfully**

**Incorrect Login**

**2**

**Incorrect Login**

**1**

**Incorrect Login**

**0**

**Maximum number of attempts exceeded,,CONTACT ADMIN**

**7) To search a number in the given array:**

**Program:**

**package array;**

**import java.util.\*;**

**public class searchnumber {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Scanner sc = new Scanner(System.*in*);**

**int i,n,search,flag=0;**

**int[] a = new int[15];**

**System.*out*.println("Enter the elements") ;**

**for(i=0;i<15;i++)**

**{**

**a[i] = sc.nextInt();**

**}**

**System.*out*.println("Enter the element to be searched");**

**search = sc.nextInt();**

**/\*Perform search operation\*/**

**for(i=0;i<15;i++)**

**{**

**if(a[i]==search)**

**{**

**System.*out*.println("Element "+search+" found at "+i+" position");**

**flag=1;**

**break;**

**}**

**}**

**if(flag==0)**

**{**

**System.*out*.println("Element "+search+" not found");**

**}**

**}**

**}**

**Output:**

**Enter the elements**

**5 12 14 6 78 19 1 23 26 35 27 7 52 86 47**

**Enter the element to be searched**

**19**

**Element 19 found at 5 position**

**8) Bubble Sort**

**Program:**

**package sort;**

**public class bubblesort {**

**static void bubbleSort(int[] a) {**

**int n = a.length;**

**int temp = 0;**

**for(int i=0; i < n; i++){**

**for(int j=1; j < (n-i); j++){**

**if(a[j-1] > a[j]){**

**temp = a[j-1];**

**a[j-1] = a[j];**

**a[j] = temp;**

**}**

**}**

**}**

**}**

**public static void main(String[] args) {**

**int a[] ={5,12,14,6,78,19,1,23,26,35,37,7,52,86,47};**

**System.*out*.println("Array Before Bubble Sort");**

**for(int i=0; i < a.length; i++){**

**System.*out*.print(a[i] + " ");**

**}**

**System.*out*.println();**

***bubbleSort*(a);**

**System.*out*.println("Array After Bubble Sort");**

**for(int i=0; i < a.length; i++){**

**System.*out*.print(a[i] + " ");**

**}**

**}**

**}**

**Output:**

**Array Before Bubble Sort**

**5 12 14 6 78 19 1 23 26 35 37 7 52 86 47**

**Array After Bubble Sort**

**1 5 6 7 12 14 19 23 26 35 37 47 52 78 86**

**9) Total and Avergae:**

**Program:**

**​​package totalavg;**

**import java.util.\*;**

**public class TotalAvgMarks {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Scanner in=new Scanner(System.*in*);**

**int m1[]=new int[3];**

**int m2[]=new int[3];**

**int m3[]=new int[3];**

**int totSub[]=new int[3];**

**int totStu[]=new int[3];**

**double avgstu[]=new double[3];**

**double avgSub[]=new double[3];**

**int i=0;**

**System.*out*.println("Enter the marks of 1st student");**

**for(i=0;i<3;i++) {**

**m1[i]=in.nextInt();**

**totStu[0]+=m1[i];**

**}**

**System.*out*.println("Enter the marks of 2nd student");**

**for(i=0;i<3;i++) {**

**m2[i]=in.nextInt();**

**totStu[1]+=m2[i];**

**}**

**System.*out*.println("Enter the marks of 3rd student");**

**for(i=0;i<3;i++) {**

**m3[i]=in.nextInt();**

**totStu[2]+=m3[i];**

**}**

**for(i=0;i<3;i++) {**

**totSub[i]=m1[i]+m2[i]+m3[i];**

**}**

**for(i=0;i<3;i++) {**

**avgSub[i]=totSub[i]/3;**

**}**

**for(i=0;i<3;i++) {**

**avgstu[i]=totStu[i]/3;**

**}**

**System.*out*.println("Total and Average marks of 1st Student are "+totStu[0]+" and "+avgstu[0]);**

**System.*out*.println("Total and Average marks of 2nd Student are "+totStu[1]+" and "+avgstu[1]);**

**System.*out*.println("Total and Average marks of 3rd Student are "+totStu[2]+" and "+avgstu[2]);**

**System.*out*.println("Total and Average marks of 1st Subject are "+totSub[0]+" and "+avgSub[0]);**

**System.*out*.println("Total and Average marks of 2nd Subject are "+totSub[1]+" and "+avgSub[1]);**

**System.*out*.println("Total and Average marks of 3rd Subject are "+totSub[2]+" and "+avgSub[2]);**

**in.close();**

**}**

**}**

**Output:**

**Enter the marks of 1st student**

**80**

**75**

**45**

**Enter the marks of 2nd student**

**76**

**86**

**97**

**Enter the marks of 3rd student**

**78**

**66**

**55**

**Total and Average marks of 1st Student are 200 and 66.0**

**Total and Average marks of 2nd Student are 259 and 86.0**

**Total and Average marks of 3rd Student are 199 and 66.0**

**Total and Average marks of 1st Subject are 234 and 78.0**

**Total and Average marks of 2nd Subject are 227 and 75.0**

**Total and Average marks of 3rd Subject are 197 and 65.0**

**Assignment 2:**

**1)Singleton:**

class sample{

private static sample *obj*;

private sample(){ }

public static sample getInstance()

{

if(*obj*==null) {

*obj*=new sample();

}

return *obj*;

}

public static void getConnection()

{

System.*out*.println("you are connected");

}

}

class Singleton {

public static void main(String[] args)

{

sample obj;

obj=sample.*getInstance*();

sample.*getConnection*();

}

}

**Output:**

you are connected

**2) Total salary of employees:**

**Program:**

package totalsalary;

class employee

{

int incentive,overtime;

public int totalsal()

{

return 0;

}

}

class manager extends employee

{

public manager(int incentive) {

this.incentive=incentive;

// TODO Auto-generated constructor stub

}

public int totalsal()

{

return incentive;

}

}

class labour extends employee

{

public labour(int overtime) {

this.overtime=overtime;

// TODO Auto-generated constructor stub

}

public int totalsal()

{

return overtime;

}

}

public class methodoverriding {

public static void main(String[] args)

{

manager m= new manager(400000);

labour l= new labour(600000);

System.*out*.println("total salary of employees: " + (m.totalsal()+l.totalsal()));

}

}

**Output:**

total salary of employees: 1000000

**3)TotalAmount:**

package account;

class bank {

int amount;

bank()

{

amount=0;

}

bank(int amount)

{

this.amount=amount;

}

int getAmount()

{

return amount;

}

}

class SavingsAccount extends bank {

SavingsAccount()

{

amount=0;

}

SavingsAccount(int deposit)

{

super(deposit);

}

int getAmount()

{

return (super.getAmount());

}

}

class CurrentAccount extends bank {

CurrentAccount()

{

super();

}

CurrentAccount(int credit)

{

super(credit);

}

int getSalary()

{

return (super.getAmount());

}

}

public class BankAccount {

public static void main(String[] args)

{

SavingsAccount s= new SavingsAccount(400000);

System.*out*.println("Amount: " +s.getAmount());

CurrentAccount c= new CurrentAccount(600000);

System.*out*.println("Amount: " + c.getSalary());

int sum=s.getAmount()+c.getSalary();

System.*out*.println("Total Amount in the bank: " +sum);

}

}

**Output:**

Amount: 400000

Amount: 600000

Total Amount in the bank: 1000000

1. **i) a**bstract class student

{

public void age()

{

System.*out*.println(20);

}

}

class sree extends student{

public void age()

{

System.*out*.println(20);

}

}

public class Abstract {

public static void main(String[] args) {

// TODO Auto-generated method

sree s = new sree();

s.age();

}

}

**Output:**

**12**

**ii)**

abstract class AbstractClass{

public void myMethod(){

System.*out*.println("Hello");

}

abstract public void anotherMethod();

}

public class Demo extends AbstractClass{

public void anotherMethod() {

System.*out*.print("Abstract method");

}

public static void main(String args[])

{

//error: You can't create object of it

AbstractClass obj = new AbstractClass();

obj.anotherMethod();

}

}

**Output:**

**AbstractClass.java:15: error: AbstractClass is abstract; cannot be instantiated**

**iv)**  abstract private class student

{

public void age()

{

System.*out*.println(20);

}

}

class sree extends student{

public void age()

{

System.*out*.println(20);

}

}

public class Abstract {

public static void main(String[] args) {

// TODO Auto-generated method

sree s = new sree();

s.age();

}

}

**Output:**

**Exception in thread "main" java.lang.Error: Unresolved compilation problems:**

**Illegal modifier for the class student; only public, abstract & final are permitted**

**Implicit super constructor student() is not visible for default constructor. Must define an explicit constructor**

**at sree.<init>(Abstract.java:1)**

**at Abstract.main(Abstract.java:18)**

**v) abstract final class student**

**{**

**public void age()**

**{**

**System.*out*.println(20);**

**}**

**}**

**class sree extends student{**

**public void age()**

**{**

**System.*out*.println(20);**

**}**

**}**

**public class Abstract {**

**public static void main(String[] args) {**

**// TODO Auto-generated method**

**sree s = new sree();**

**s.age();**

**}**

**}**

**Output:**

**Exception in thread "main" java.lang.Error: Unresolved compilation problems:**

**The class student can be either abstract or final, not both**

**The type sree cannot subclass the final class student**

**at sree.<init>(Abstract.java:1)**

**at Abstract.main(Abstract.java:18)**

**vi) abstract class MyClass {**

**public void display() {**

**System.*out*.println("this is abstract method");**

**}**

**}**

**public class Abstract {**

**public static void main(String[] args) {**

**// TODO Auto-generated method**

**new Abstract().display();**

**}**

**private void display() {**

**// TODO Auto-generated method stub**

**System.*out*.println("this is abstract method");**

**}**

**}**

**Output:**

**this is abstract method**

1. **shapes:**

**Program:**

**package shapes;**

**abstract class shape**

**{**

**public abstract void draw();**

**}**

**class line extends shape**

**{**

**@Override**

**public void draw() {**

**System.*out*.println("this is line");**

**// TODO Auto-generated method stub**

**}**

**}**

**class rectangle extends shape**

**{**

**@Override**

**public void draw() {**

**// TODO Auto-generated method stub**

**System.*out*.println("this is rectangle");**

**}**

**}**

**class cube extends shape**

**{**

**@Override**

**public void draw() {**

**// TODO Auto-generated method stub**

**System.*out*.println("this is cube");**

**}**

**}**

**public class Shapes {**

**public static void main(String[] args)**

**{**

**line l=new line();**

**l.draw();**

**rectangle r=new rectangle();**

**r.draw();**

**cube c=new cube();**

**c.draw();**

**}**

**}**

**Output:**

**this is line**

**this is rectangle**

**this is cube**

**6)**

**Program:**

**abstract class Persistence**

**{**

**abstract void persist();**

**}**

**class FilePersistence extends Persistence**

**{**

**public void persist() {**

**System.*out*.println("Data is being saved in File");**

**}**

**}**

**class DatabasePersistence extends Persistence**

**{**

**public void persist() {**

**System.*out*.println("Data is being saved in Database");**

**}**

**}**

**public class main {**

**public static void client(Persistence p) {**

**p.persist();**

**}**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Persistence p1=new FilePersistence();**

**Persistence p2=new DatabasePersistence();**

***client*(p1);**

***client*(p2);**

**}**

**}**

**Output:**

**Data is being saved in File**

**Data is being saved in Database**

**7)**

**Program:**

**public abstract class DessertItem {**

**public abstract int getCost();**

**}**

**public class Candy extends DessertItem {**

**public int getCost() {**

**// TODO Auto-generated method stub**

**return 2;**

**}**

**}**

**public class Cookie extends DessertItem{**

**public int getCost() {**

**// TODO Auto-generated method stub**

**return 1;**

**}**

**}**

**public class Icecream extends DessertItem{**

**@Override**

**public int getCost() {**

**// TODO Auto-generated method stub**

**return 25;**

**}**

**}**

**import java.util.\*;**

**public class MainClass {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**Scanner in=new Scanner(System.*in*);**

**DessertItem candy=new Candy();**

**DessertItem cookie=new Cookie();**

**DessertItem icecream=new Icecream();**

**List<Integer> candies=new ArrayList<Integer>();**

**List<Integer> cookies=new ArrayList<Integer>();**

**List<Integer> icecreams=new ArrayList<Integer>();**

**int n,i,size;**

**while(true) {**

**System.*out*.println("Type o for Owner, c for customer and e for exit");**

**String type=in.next();**

**if(type.equals("o")) {**

**System.*out*.println("Enter the no.of candies to add into storage");**

**n=in.nextInt();**

**for(i=0;i<n;i++) {**

**candies.add(1);**

**}**

**System.*out*.println("Enter the no.of cookies to add into storage");**

**n=in.nextInt();**

**for(i=0;i<n;i++) {**

**cookies.add(1);**

**}**

**System.*out*.println("Enter the no.of icecreams to add into storage");**

**n=in.nextInt();**

**for(i=0;i<n;i++) {**

**icecreams.add(1);**

**}**

**}**

**else if(type.equals("c")) {**

**System.*out*.println("Place the order:");**

**System.*out*.println("Enter the no.of candies");**

**n=in.nextInt();**

**size=candies.size()-1;**

**System.*out*.println(n+" "+size);**

**for(i=0;i<n;i++) {**

**candies.remove(size-i);**

**}**

**if(n>0)**

**System.*out*.println("Cost for candies is "+(n\*candy.getCost())+" dollars");**

**System.*out*.println("Enter the no.of cookies");**

**n=in.nextInt();**

**size=cookies.size()-1;**

**for(i=0;i<n;i++) {**

**cookies.remove(size-i);**

**}**

**if(n>0)**

**System.*out*.println("Cost for cookies is "+(n\*cookie.getCost())+" euros");**

**System.*out*.println("Enter the no.of icecreams");**

**n=in.nextInt();**

**size=icecreams.size()-1;**

**for(i=0;i<n;i++) {**

**icecreams.remove(size-i);**

**}**

**if(n>0)**

**System.*out*.println("Cost for icecreams is "+(n\*icecream.getCost())+" rupees");**

**}**

**else {**

**System.*out*.println("Exited");**

**break;**

**}**

**}**

**in.close();**

**}**

**}**

**Output:**

**Type o for Owner, c for customer and e for exit**

**o**

**Enter the no.of candies to add into storage**

**7**

**Enter the no.of cookies to add into storage**

**6**

**Enter the no.of icecreams to add into storage**

**7**

**Type o for Owner, c for customer and e for exit**

**o**

**Enter the no.of candies to add into storage**

**3**

**Enter the no.of cookies to add into storage**

**2**

**Enter the no.of icecreams to add into storage**

**8**

**Type o for Owner, c for customer and e for exit**

**c**

**Place the order:**

**Enter the no.of candies**

**2**

**2 9**

**Cost for candies is 4 dollars**

**Enter the no.of cookies**

**4**

**Cost for cookies is 4 euros**

**Enter the no.of icecreams**

**2**

**Cost for icecreams is 50 rupees**

**Type o for Owner, c for customer and e for exit**

**4**

**Exited**

**Assignment 3:**

**String class:**

**Program:**

**public class string {**

**public static void main(String[] args) {**

**// TODO Auto-generated method stub**

**String str="Hello World";**

**String s1="Hello";**

**String s2=" How are you?";**

**int len=str.length();**

**System.*out*.println("strings joined are:" +(s1+s2));**

**System.*out*.println("the length of the string is: " +len);**

**}**

**}**

**Output:**

**strings joined are:Hello How are you?**

**the length of the string is: 11**

**Program:**

**import java.util.\*;**

**public class stringmethods {**

**public static void main(String[] args) {**

**String str;**

**Scanner sc=new Scanner(System.*in*);**

**System.*out*.println("enter the sentence: ");**

**str=sc.nextLine();**

**String s1=str.toLowerCase();**

**String s2=str.toUpperCase();**

**String s3=str.replace('a', '$');**

**System.*out*.println("lower case: " +s1);**

**System.*out*.println("Upper case: " +s2);**

**System.*out*.println("after replace " +s3);**

**if(str.contains("collection"))**

**{**

**System.*out*.println("the string contains the word collection");**

**}**

**else**

**{**

**System.*out*.println("the string does not contains the word collection");**

**}**

**String s4;**

**System.*out*.println("enter the sentence: ");**

**s4=sc.nextLine();**

**if(str.equals(s4))**

**{**

**System.*out*.println("str equals s4");**

**}**

**else**

**{**

**System.*out*.println("str not equals s4");**

**}**

**if(str.equalsIgnoreCase(s4))**

**{**

**System.*out*.println("str equals s4");**

**}**

**else**

**{**

**System.*out*.println("str not equals s4");**

**}**

**}**

**}**

**Output:**

**enter the sentence:**

**Java String pool refers to collection of Strings which are stored in heap memory**

**lower case: java string pool refers to collection of strings which are stored in heap memory**

**Upper case: JAVA STRING POOL REFERS TO COLLECTION OF STRINGS WHICH ARE STORED IN HEAP MEMORY**

**after replace J$v$ String pool refers to collection of Strings which $re stored in he$p memory**

**the string contains the word collection**

**enter the sentence:**

**java string pool refers to collection of strings which are stored in heap memory**

**str not equals s4**

**str equals s4**

**String Buffer:**

**Program:**

**public class stringbuffer {**

**public static void main(String[] args) {**

**StringBuffer str= new StringBuffer("StringBuffer is a peer class of string that provides much of");**

**str.append(" the functionality of strings");**

**System.*out*.println("after append: " +str);**

**StringBuffer s1=new StringBuffer("it is used to at the specified index position");**

**s1.insert(13, " insert text");**

**System.*out*.println("after inserting text: " +s1);**

**StringBuffer s2=new StringBuffer("This method returns the reversed object on which it was called");**

**s2.reverse();**

**System.*out*.println(s2);**

**}**

**}**

**Output:**

**after append: StringBuffer is a peer class of string that provides much of the functionality of strings**

**after inserting text: it is used to insert text at the specified index position**

**dellac saw ti hcihw no tcejbo desrever eht snruter dohtem sihT**

**String Builder:**

**Program:**

**public class stringbuilder {**

**public static void main(String[] args) {**

**StringBuilder str= new StringBuilder("StringBuffer is a peer class of string that provides much of");**

**str.append(" the functionality of strings");**

**System.*out*.println("after append: " +str);**

**StringBuilder s1=new StringBuilder("it is used to at the specified index position");**

**s1.insert(13, " insert text");**

**System.*out*.println("after inserting text: " +s1);**

**StringBuilder s2=new StringBuilder("This method returns the reversed object on which it was called");**

**s2.reverse();**

**System.*out*.println(s2);**

**}**

**}**

**Ouput:**

**after append: StringBuffer is a peer class of string that provides much of the functionality of strings**

**after inserting text: it is used to insert text at the specified index position**

**dellac saw ti hcihw no tcejbo desrever eht snruter dohtem sihT**

**Assignment 4:**

**1)**

**import java.util.\*;**

**public class test {**

**private static final Scanner *sc* = null;**

**public static void main(String[] args)**

**{**

**try {**

**Scanner sc=null;**

**sc=new Scanner(System.*in*);**

**System.*out*.println("enter the dividend number: ");**

**int dividend=sc.nextInt();**

**System.*out*.println("enter the divisor number: ");**

**int divisor=sc.nextInt();**

**int result=dividend/divisor;**

**System.*out*.println("the result is: " +result);**

**sc.close();**

**} catch (Exception e) {**

**// TODO Auto-generated catch block**

**e.printStackTrace();**

**}**

**finally**

**{**

**if(*sc*!=null)**

***sc*.close();**

**}**

**System.*out*.println("main method ends...");**

**}**

**}**

**Output:**

**enter the dividend number:**

**22**

**enter the divisor number:**

**0**

**java.lang.ArithmeticException: / by zero**

**main method ends...**

**at test.main(test.java:14)**

**2)handle arithmetic exception by raising unsupportedoperatonexception in java:**

**import java.util.Scanner;**

**public class test1 {**

**private static final Scanner *sc* = null;**

**public static void main(String[] args)**

**{**

**try {**

**Scanner sc=null;**

**sc=new Scanner(System.*in*);**

**System.*out*.println("enter the dividend number: ");**

**int dividend=sc.nextInt();**

**System.*out*.println("enter the divisor number: ");**

**int divisor=sc.nextInt();**

**int result=dividend/divisor;**

**System.*out*.println("the result is: " +result);**

**sc.close();**

**}**

**catch (Exception e) {**

**// TODO Auto-generated catch block**

**throw new UnsupportedOperationException("division by zero is not possible");**

**}**

**finally**

**{**

**if(*sc*!=null)**

***sc*.close();**

**}**

**System.*out*.println("main method ends...");**

**}**

**}**

**Output:**

**enter the dividend number:**

**33**

**enter the divisor number:**

**0**

**Exception in thread "main" java.lang.UnsupportedOperationException: division by zero is not possible**

**at test1.main(test1.java:20)**

**3)a) ​​InsufficientBalanceException.java**

public class InsufficientBalanceException extends RuntimeException {

//private String message;

public InsufficientBalanceException(String message) {

super(message);

}

}

**Exception.java**

import java.util.\*;

public class exception {

double withdraw,deposit;

double balance=0;

long id;

/\*public void bankaccount(long id,double balance)

{

this.id=id;

this.balance=balance;

}\*/

public void deposit() {

double amount;

Scanner sc=new Scanner(System.*in*);

System.*out*.println("deposit the amount");

amount=sc.nextInt();

if (amount <= 0) {

throw new InsufficientBalanceException("the amount is deposited");

}

}

public void withdraw() throws InsufficientBalanceException {

double amount;

Scanner sc=new Scanner(System.*in*);

System.*out*.println("withdraw the amount");

amount=sc.nextInt();

if (amount > balance) {

throw new InsufficientBalanceException("insufficient balance");

}

balance = balance - amount;

}

public static void main(String[] args) {

exception e=new exception();

e.deposit();

try {

e.withdraw();

}

catch(InsufficientBalanceException ex)

{

ex.printStackTrace();

}

}

}

Output:

deposit the amount

1000

withdraw the amount

1100

InsufficientBalanceException: insufficient balance

at exception.withdraw(exception.java:26)

at exception.main(exception.java:35)

**b) IllegalBankTransactionException.java**

public class IllegalBankTransactionException extends RuntimeException {

public IllegalBankTransactionException() {

super();

}

}

**Exception.java**

import java.util.\*;

public class exception {

double withdraw,deposit;

double balance=0;

long id;

/\*public void bankaccount(long id,double balance)

{

this.id=id;

this.balance=balance;

}\*/

public void deposit() {

double amount;

@SuppressWarnings("resource")

Scanner sc=new Scanner(System.*in*);

System.*out*.println("deposit the amount");

amount=sc.nextInt();

balance+=amount;

}

public void withdraw() throws IllegalBankTransactionException {

double amount;

@SuppressWarnings("resource")

Scanner sc=new Scanner(System.*in*);

System.*out*.println("withdraw the amount");

amount=sc.nextInt();

if ( balance<balance-amount) {

throw new IllegalBankTransactionException();

}

}

public static void main(String[] args) {

exception e=new exception();

e.deposit();

try

{

e.withdraw();

}

catch(IllegalBankTransactionException ex)

{

ex.printStackTrace();

}

}

}

Output:

deposit the amount

1000

withdraw the amount

-1000

IllegalBankTransactionException

at exception.withdraw(exception.java:26)

at exception.main(exception.java:36)

**GENERICS**

**Assignment 5**

1. **Use a HashSet to hold Employee objects:**

import java.util.HashSet;

import java.util.Set;

public class Employee {

private int id;

private String name;

private double salary;

private String department;

public Employee(int id, String name, double salary, String department) {

super();

this.id = id;

this.name = name;

this.salary=salary;

this.department=department;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

@Override

public String toString() {

return "id=" + id + ", name=" + name + ", salary=" + salary + ", department=" + department + "";

}

public static void main(String[] args) {

Set<Employee> employees = new HashSet<>();

employees.add(new Employee(1 , "Rishi" , 300000, "HR"));

//System.out.println("\n");

employees.add(new Employee(2, "Priya", 400000, "A1"));

System.*out*.println(employees);

}

}

**Output:**

[id=1, name=Rishi, salary=300000.0, department=HR, id=2, name=Priya, salary=400000.0, department=A1]

**2)Randomkeyandvalue:**

import java.util.\*;

import java.util.HashMap;

public class RandomKeysandValues {

public static void main(String[] args) {

// TODO Auto-generated method stub

HashMap<Integer, Double> h=new HashMap<Integer, Double>();

h.put(1,1.1);

h.put(2,2.1);

h.put(3,3.1);

h.put(4,4.1);

h.put(5,5.1);

h.put(6,6.1);

h.put(7,7.1);

h.put(8,8.1);

h.put(9,9.1);

h.put(10,10.1);

System.*out*.println("random values: " +h);

}

}

**Output:**

random values: {1=1.1, 2=2.1, 3=3.1, 4=4.1, 5=5.1, 6=6.1, 7=7.1, 8=8.1, 9=9.1, 10=10.1}

**3) Swapping the position the two elements in array:**

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class Swap {

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 = {"Hello", "Hi",};

*swap*(a, 0, 1);

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

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

*swap*(l, 0, 1);

System.*out*.println("l:"+l);

}

public static void main(String[] args)

{

*test*();

}

}

Output:

a:[Hi, Hello]

l:[Hello, Hi]

**4)**

public class Pair<K,V> {

private K key;

private V value;

public K getKey() {

return key;

}

public void setKey(K key) {

this.key = key;

}

public V getValue() {

return value;

}

public void setValue(V value) {

this.value = value;

}

public static void main(String[] args)

{

Pair<String,String> p=new Pair<String, String>();

p.setKey("1");

p.setValue("Hello");

System.*out*.println("key is: "+p.getKey() + " Value is: " +p.getValue());

Pair<String,java.util.Date> p1=new Pair<String, java.util.Date>();

p1.setKey("Today is");

p1.setValue(new java.util.Date());

System.*out*.println("key is: "+p1.getKey()+ " Value is: " +p1.getValue());

}

}

**Output:**

key is: 1 Value is: Hello

key is: Today is Value is: Mon Aug 09 13:54:19 IST 2021

**COLLECTIONS**

**ASSIGNMENT 6**

1. **Contact.java**

**public** **class** Contact {

**private** **long** phonenumber;

**private** String name;

**private** String email;

**private** String gender;

**public** Contact(**long** phonenumber,String email,String name, String gender)

{

**this**.phonenumber=phonenumber;

**this**.name=name;

**this**.email=email;

**this**.gender=gender;

}

**public** **long** getPhonenumber() {

**return** phonenumber;

}

**public** **void** setPhonenumber(**long** phonenumber) {

**this**.phonenumber = phonenumber;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** String getGender() {

**return** gender;

}

**public** **void** setGender(String gender) {

**this**.gender = gender;

}

@Override

**public** String toString() {

**return** "Contact [phonenumber=" + phonenumber + ", name=" + name + ", email=" + email + ", gender=" + gender

+ "]";

}

}

**PhoneNumbers.java:**

**import** java.util.Set;

**import** java.util.TreeMap;

**import** java.util.Map.Entry;

**import** java.util.Map;

**import** java.util.Collections;

**import** java.util.Collections;

**import** java.util.Iterator;

**public** **class** PhoneNumbers {

**public** **static** **void** main(String[] args) {

Contact c1= **new** Contact(827163829, "Rishi", "rishi@gmail.com","Male");

Contact c2= **new** Contact(863725641, "Priya", "priya@gmail.com","Female");

Contact c3= **new** Contact(985623619,"Raghu","raghu@gmail.com","Male");

TreeMap<Long,Contact> tm=**new** TreeMap<Long,Contact>(Collections.*reverseOrder*());

tm.put((**long**) 843672819,c1);

tm.put((**long**) 823628919,c2);

tm.put((**long**) 271298012,c3);

Set<Entry<Long, Contact>> entrySet = tm.entrySet();

Iterator<Entry<Long, Contact>> iterator = entrySet.iterator();

**while** (iterator.hasNext()) {

Map.Entry<Long, Contact> entry = (Map.Entry<Long, Contact>) iterator.next();

Contact contact = entry.getValue();

//System.out.println(entry.getKey());

System.***out***.println(contact);

}

}

}

**Output**:

Contact [phonenumber=827163829, name=rishi@gmail.com, email=Rishi, gender=Male]

Contact [phonenumber=863725641, name=priya@gmail.com, email=Priya, gender=Female]

Contact [phonenumber=985623619, name=raghu@gmail.com, email=Raghu, gender=Male]

**2)**import java.util.HashSet;

public class UniqueObjects {

public static void main(String[] args) {

HashSet<String> set = new HashSet<>();

set.add("one");

set.add("two");

set.add("three");

set.add("four");

set.add("five");

set.add("six");

set.add("seven");

set.add("eight");

set.add("nine");

set.add("ten");

set.add("one");

System.*out*.println(set);

}

}

**Output:**

[nine, six, four, one, seven, ten, two, three, five, eight]

**4) leap year or not:**

**import** java.time.LocalDate;

**import** java.util.LinkedList;

**public** **class** Date {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

LocalDate d1=LocalDate.*of*(2000,12,23);

//LocalDate d2=LocalDate.of(2001,12,23);

LinkedList<Object> ll =**new** LinkedList<>();

ll.add(d1);

//ll.add(d2);

**for**(Object i: ll)

{

**long** a,c;

**int** y=d1.getYear();

// y=d2.getYear();

**if**(y!=0)

{

a=(y%400==0)?(c=1):((y%100==0)?(c=0):((y%4==0)?(c=1):(c=0)));

**if**(a==1)

System.***out***.println("your date of birth is " +d1+ " and it is a leap year" );

**else**

System.***out***.println("your date of birth is " +d1+ " and it is not a leap year");

}

}

}

}

**Output:**

your date of birth is 2000-12-23 and it is a leap year

**3)**package assignments;

import java.util.\*;

class Employee implements Comparable<Employee>{

private int empid;

private String empname;

private String empdept;

private int empsalary;

public Employee(int empid, String empname, String empdept, int empsalary) {

super();

this.empid = empid;

this.empname = empname;

this.empdept = empdept;

this.empsalary = empsalary;

}

public int getempId() {

return empid;

}

public String getempName() {

return empname;

}

public String getempDept() {

return empdept;

}

public int getempSalary() {

return empsalary;

}

@Override

public String toString() {

return "\nEmployee Id=" + empid + ",\nEmployee Name=" + empname + ", "

+ "\nEmployee Salary=" + empdept + ","

+ "\nEmployee Department=" + empsalary + "\n ";

}

@Override

public int compareTo(Employee e) {

int ret= this.empid-e.empid;

if(ret==0) {

ret= this.empname.compareTo(e.empname);

}

if(ret==0)

ret= this.empdept.compareTo(e.empdept);

if(ret==0)

ret= this.empsalary-e.empsalary;

return ret;

}

}

class IdComparator implements Comparator<Employee>{

public int compare(Employee e1, Employee e2) {

return e1.getempId()-e2.getempId();

}

}

class NameComparator implements Comparator<Employee>{

public int compare(Employee e1, Employee e2) {

int res= e1.getempName().compareTo(e2.getempName());

if(res==0)

res= e1.getempId()-e2.getempId();

return res;

}

}

class DeptComparator implements Comparator<Employee>{

public int compare(Employee e1, Employee e2) {

int res= e1.getempDept().compareTo(e2.getempDept());

if(res==0)

res= e1.getempId()-e2.getempId();

return res;

}

}

class SalaryComparator implements Comparator<Employee>{

public int compare(Employee e1, Employee e2) {

int res= e1.getempSalary()-e2.getempSalary();

if(res==0)

res= e1.getempId()-e2.getempId();

return res;

}

}

public class collection3 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Set<Employee> employees;

Scanner sc= new Scanner(System.in);

System.out.println("a) ID\nb) Name\nc) Department\nd) Salary");

System.out.println("Enter your choice");

String c= sc.next();

if(c.compareTo("a")==0)

employees= new TreeSet<>(new IdComparator());

else if(c.compareTo("b")==0)

employees= new TreeSet<>(new NameComparator());

else if(c.compareTo("c")==0)

employees= new TreeSet<>(new DeptComparator());

else if(c.compareTo("d")==0)

employees= new TreeSet<>(new SalaryComparator());

else {

System.out.println("Invalid Option\n");

employees= new TreeSet<>();

}

employees.add(new Employee(1, "Navya","CSE",45000));

employees.add(new Employee(4, "Sai","Business",56000));

employees.add(new Employee(3, "Viswani","IT",43000));

employees.add(new Employee(2, "Divya","Business",97760));

employees.add(new Employee(4, "Ammu","IT",78000));

for(Employee e: employees)

System.out.println(e);

sc.close();

}

}

**Output:**

**ANNOTATIONS**

**1)@test**

@interface Test

{

}

class method{

@Test

void method()

{

System.*out*.println("1st method");

}

}

public class Annotation1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

method m=new method();

m.method();

}

}

Output:

1st method

**2)@info**

import java.lang.annotation.ElementType;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.annotation.Target;

@Target({ElementType.*TYPE*,ElementType.*METHOD*})

@Retention(RetentionPolicy.*RUNTIME*)

@interface Info{

int Id();

String Date();

String Time();

int Version();

}

@Info(Date = "10-08-2021", Id = 1, Time = "1:00", Version = 6)

class InfoTest{

String name;

String supervisor;

public InfoTest(String name, String supervisor) {

super();

this.name = name;

this.supervisor = supervisor;

}

}

public class Annotation1 {

public static void main(String[] args) {

InfoTest obj = new InfoTest("Rishi", "Ritu");

//obj.display();

Class c=obj.getClass();

System.*out*.println("Authors name: " +obj.name);

System.*out*.println("Developers Supervisor: " +obj.supervisor);

Annotation an=c.getAnnotation(Info.class);

Info s = (Info)an;

System.*out*.println("Author ID: " +s.Id());

System.*out*.println("Date: " +s.Date());

System.*out*.println("Time: " +s.Time());

System.*out*.println("Version: " +s.Version());

}

}

Output:

Authors name: Rishi

Developers Supervisor: Ritu

Author ID: 1

Date: 10-08-2021

Time: 1:00

Version: 6

**LAMBDA EXPRESSIONS:**

**1)Arithmetic operators:**

**interface** myadd{

**int** add(**int** a,**int** b);

}

**interface** mysub{

**int** sub(**int** a,**int** b);

}

**interface** mymul{

**int** mul(**int** a,**int** b);

}

**interface** mydiv{

**int** div(**int** a,**int** b);

}

**public** **class** lambda1 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

myadd addfunction=(**int** a, **int** b) -> a+b;

System.***out***.println("addition of two numbers: ");

System.***out***.println(addfunction.add(10,10));

mysub subfunction=(**int** a, **int** b) -> a-b;

System.***out***.println("subtraction of two numbers: ");

System.***out***.println(subfunction.sub(50,10));

mymul mulfunction=(**int** a, **int** b) -> a\*b;

System.***out***.println("multplication of two numbers: ");

System.***out***.println(mulfunction.mul(10,10));

mydiv divfunction=(**int** a, **int** b) -> a/b;

System.***out***.println("division of two numbers: ");

System.***out***.println(divfunction.div(20,10));

}

}

**Output:**

addition of two numbers:

20

subtraction of two numbers:

40

multplication of two numbers:

100

division of two numbers:

2

**2)**

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Stream;

class order{

int id;

String status;

float price;

public order(int id, String status, float price) {

super();

this.id = id;

this.status = status;

this.price = price;

}

}

public class lambdaassigment2 {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<order> l=new ArrayList<>();

l.add(new order(1,"Accepted",10500));

l.add(new order(2,"Completed",28000));

l.add(new order(3,"Accepted",32000));

Stream<order> s=l.stream().filter(p->p.price>10000);

s.forEach(order->System.*out*.println(order.price+ " : " +order.status));

}

}

**Output:**

10500.0 : Accepted

28000.0 : Completed

32000.0 : Accepted

**3) person.class**

**public** **class** person {

String firstname;

String lastname;

**int** age;

**public** person(String firstname, String lastname, **int** age) {

**super**();

**this**.firstname = firstname;

**this**.lastname = lastname;

**this**.age = age;

}

**public** String getFirstname() {

**return** firstname;

}

**public** **void** setFirstname(String firstname) {

**this**.firstname = firstname;

}

**public** String getLastname() {

**return** lastname;

}

**public** **void** setLastname(String lastname) {

**this**.lastname = lastname;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

@Override

**public** String toString() {

**return** "person [firstname=" + firstname + ", lastname=" + lastname + ", age=" + age + "]";

}

}

**Supplier:**

import java.util.Arrays;

import java.util.List;

import java.util.function.Consumer;

import java.util.function.Predicate;

import java.util.function.Supplier;

public class lambda2 {

public static void main(String[] args) {

Supplier < person > supplier =() -> { return new person("Charles", "Dickens", 45);

};

person p = supplier.get();

System.*out*.println(p);

}

}

Output:

person [firstname=Charles, lastname=Dickens, age=45]

**Predicate and Consumer:**

import java.util.Arrays;

import java.util.List;

import java.util.function.Consumer;

import java.util.function.Predicate;

import java.util.function.Supplier;

public class lambda2 {

public static void main(String[] args) {

List<person> people=Arrays.*asList*(

new person("Charles", "Dickens", 45),

new person("Lewis", "Caroll", 45),

new person("Thomas", "Carlyle", 45),

new person("Charolotte", "Bronte", 45)

);

System.*out*.println("print all persons");

*Program*(people, p->true, p->System.*out*.println(p));

}

public static void Program(List<person> people, Predicate<person> predicate, Consumer<person> consumer)

{

for(person p: people)

{

//consumer.accept(p);

consumer.accept(p);

}

}

}

Output:

print all persons

person [firstname=Charles, lastname=Dickens, age=45]

person [firstname=Lewis, lastname=Caroll, age=45]

person [firstname=Thomas, lastname=Carlyle, age=45]

person [firstname=Charolotte, lastname=Bronte, age=45]

**3)Function:**

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.function.Consumer;

**import** java.util.function.Function;

**import** java.util.function.Predicate;

**import** java.util.function.Supplier;

**public** **class** lambda2 {

**public** **static** **void** main(String[] args) {

Function<person, String> funcPerToString= (person p)-> {**return** p.getFirstname();};

List<person> people=Arrays.*asList*(

**new** person("Charles", "Dickens", 45),

**new** person("Lewis", "Caroll", 45),

**new** person("Thomas", "Carlyle", 45),

**new** person("Charolotte", "Bronte", 45)

);

System.***out***.println("print all persons");

List<person> personname=*listperson*(people, Function.*identity*());

personname.forEach(System.***out***::println);

}

**public** **static** List<person> listperson(List<person> people, Function<person, person> fun)

{

List<person> name=**new** ArrayList<person>();

**for**(person p: people)

{

//consumer.accept(p);

name.add(fun.apply(p));

}

**return** name;

}

}

Output:

print all persons

person [firstname=Charles, lastname=Dickens, age=45]

person [firstname=Lewis, lastname=Caroll, age=45]

person [firstname=Thomas, lastname=Carlyle, age=45]

person [firstname=Charolotte, lastname=Bronte, age=45]

**4) Remove the words that have oddlength:**

import java.util.ArrayList;

import java.util.List;

public class lambda4 {

public static void main(String[] args) {

//int length; String str;

List<String> l = new ArrayList<>();

l.add("Rishi");

l.add("Ritu");

l.add("Raaju");

l.add("Ravi");

l.removeIf(n->(n.length()%2==1));

for(String s:l)

{

System.*out*.println(s);

}

}

}

Output:

Ritu

Ravi

**5) i**mport java.util.ArrayList;

import java.util.List;

public class lambda4 {

public static void main(String[] args) {

//int length; String str;

List<String> l = new ArrayList<>();

l.add("Rishi");

l.add("Ritu");

l.add("Raaju");

l.add("Ravi");

StringBuilder firstLetters = new StringBuilder();

l.forEach(s -> firstLetters.append(s.substring(0, 1)) );

System.*out*.println(firstLetters.toString());

}

}

Output:

RRRR

**6) i**mport java.util.ArrayList;

import java.util.List;

import java.util.function.UnaryOperator;

public class lambda4 {

public static void main(String[] args) {

//int length; String str;

List<String> l = new ArrayList<>();

l.add("Rishi");

l.add("Ritu");

l.add("Raaju");

l.add("Ravi");

//UnaryOperator firstLetters = new UnaryOperator;

l.replaceAll( s -> s.toUpperCase()) ;

System.*out*.println(l);

}

}

Output:

[RISHI, RITU, RAAJU, RAVI]

**7)**

**import** java.util.Map;

**import** java.util.TreeMap;

**public** **class** lambda4 {

**public** **static** **void** main(String[] args) {

//int length; String str;

Map<String,Integer> map = **new** TreeMap<>() ;

map.put("Rishi",1);

map.put("Ritu",2);

map.put("Raaju",3);

map.put("Ravi",4);

StringBuilder stringResult = **new** StringBuilder();

map.forEach((k, v) -> stringResult.append(k).append(v).append(" "));

System.***out***.println(stringResult.substring(0, stringResult.length()-1).toString());

}

}

**Output:**

Raaju3 Ravi4 Rishi1 Ritu2

**8)**  import java.util.ArrayList;

import java.util.List;

import java.util.function.Consumer;

public class thread {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<Integer> l=new ArrayList<>();

l.add(1);

l.add(2);

l.add(3);

l.add(4);

Object thread;

Thread t=new Thread(()-> l.forEach(i->System.out.println(i)));

t.start();

}

public void numbers(List<Integer> l, Consumer<Integer> consumer)

{

for(int i:l)

{

consumer.accept(i);

}

}

}

**Output:**

1

2

3

4

**STREAMS**

**Fruit.java:**

public class Fruit {

String name;

int calories;

int price;

String color;

public Fruit(String name, int calories, int price, String color) {

super();

this.name = name;

this.calories = calories;

this.price = price;

this.color = color;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getCalories() {

return calories;

}

public void setCalories(int calories) {

this.calories = calories;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

@Override

public String toString() {

return "Fruit [name=" + name + ", calories=" + calories + ", price=" + price + ", color=" + color + "]";

}

}

**FruitMain.java:**

**1)**

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.stream.Stream;

**public** **class** FruitMain {

**public** **static** **void** main(String[] args) {

List<Fruit> l=**new** ArrayList<>();

l.add(**new** Fruit("Apple", 95, 30, "red"));

l.add(**new** Fruit("Mango", 201, 10, "yellow"));

l.add(**new** Fruit("kiwi", 42, 20, "green"));

l.add(**new** Fruit("PineApple", 452, 30, "brown"));

l.add(**new** Fruit("Watermelon",30, 30, "green"));

Stream<Fruit> s=l.stream().filter(c->c.calories<100);

s.forEach(Fruit->System.***out***.println(Fruit.name));

// **TODO** Auto-generated method stub

}

}

**Output**:

Apple

kiwi

Watermelon

**2)**

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Stream;

public class FruitMain {

public static void main(String[] args) {

List<Fruit> l=new ArrayList<>();

l.add(new Fruit("Apple", 95, 30, "red"));

l.add(new Fruit("Mango", 201, 10, "yellow"));

l.add(new Fruit("kiwi", 42, 20, "green"));

l.add(new Fruit("PineApple", 452, 30, "brown"));

l.add(new Fruit("Watermelon",30, 30, "green"));

l.stream()

.forEach(c->System.*out*.println(c.getColor()));

// TODO Auto-generated method stub

}

}

**Output:**

red

yellow

green

brown

green

**3)**

**import** java.util.ArrayList;

**import** java.util.Comparator;

**import** java.util.List;

**import** java.util.stream.Collectors;

**import** java.util.stream.Stream;

**public** **class** FruitMain {

**public** **static** **void** main(String[] args) {

List<Fruit> l=**new** ArrayList<>() ;

l.add(**new** Fruit("Apple", 95, 30, "red"));

l.add(**new** Fruit("strawberry",30, 10, "red"));

l.add(**new** Fruit("kiwi", 42, 20, "green"));

l.add(**new** Fruit("PineApple", 452, 30, "brown"));

l.add(**new** Fruit("Watermelon",30, 60, "red"));

l.stream()

.filter(c->c.getColor().startsWith("red"))

.sorted(Comparator.*comparingInt*(Fruit::getPrice))

.forEach(c->System.***out***.println(c.getName()));

//System.out.println(s);

// **TODO** Auto-generated method stub

}

}

**Output:**

strawberry

Apple

Watermelon

**5)** import java.util.ArrayList;

import java.util.Comparator;

import java.util.List;

public class NewsMain {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<News> l=new ArrayList<>();

l.add(new News(1001,"@rishi", "budget", "best news"));

l.add(new News(1001,"@rishi", "hi", "great"));

l.add(new News(1002,"@sita", "budget", "keepitup"));

l.add(new News(1001,"@rishi", "heyyyaa", "fantastic"));

l.add(new News(1003,"@daniel", "budget", "all the best"));

long count=l.stream()

.filter(i->i.getCommentbyuser().contains("budget"))

.count();

System.*out*.println(count);

}

}

**Output:**

3

**7)**

**import** java.util.ArrayList;

**import** java.util.Comparator;

**import** java.util.List;

**import** java.util.stream.Stream;

**public** **class** NewsMain {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

List<News> l=**new** ArrayList<>();

l.add(**new** News(1001,"@rishi", "hello", "best news"));

l.add(**new** News(1001,"@ritu", "hi", "great"));

l.add(**new** News(1002,"@sita", "wassup", "keepitup"));

l.add(**new** News(1001,"@charles", "heyyyaa", "fantastic"));

l.add(**new** News(1003,"@daniel", "bye", "all the best"));

l.forEach((News)->{

System.***out***.println("usercomments=" +News.getCommentbyuser()+ ", "+" no of comments=" +News.getComment());

});

}

}

**Output**:

usercomments=hello, no of comments=best news

usercomments=hi, no of comments=great

usercomments=wassup, no of comments=keepitup

usercomments=heyyyaa, no of comments=fantastic

usercomments=bye, no of comments=all the best

**Trader.java**

public class Trader {

String name;

String city;

public Trader(String name, String city) {

super();

this.name = name;

this.city = city;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Override

public String toString() {

return "Trader [name=" + name + ", city=" + city + "]";

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

@Override

public String toString() {

return "Transaction [trader=" + trader + ", year=" + year + ", value=" + value + "]";

}

}

**Tradermain.java**

public static void main(String[] args) {

// TODO Auto-generated method stub

List<Trader> l=new ArrayList<>();

l.add(new Trader("Ashish", "pune"));

l.add(new Trader("Edward", "delhi"));

l.add(new Trader("David", "pune"));

l.add(new Trader("Brian", "delhi"));

l.add(new Trader("Charles", "indore"));

l.stream()

.distinct()

.forEach(p->System.*out*.println(p.getCity()));

System.*out*.println("-------------------------------");

l.stream()

.filter(f->f.getCity().matches("pune"))

.forEach(c->System.*out*.println(c.getName()));

System.*out*.println("-------------------------------");

l.stream()

.sorted( Comparator.*comparing*(n->n.toString()))

.forEach(c->System.***out***.println(c.getName()));

System.*out*.println("-------------------------------");

l.stream()

.filter(f->f.getCity().matches("indore"))

.forEach(c->System.*out*.println(c.getName()));

}

}

**Output:**

pune

delhi

pune

delhi

indore

-------------------------------

Ashish

David

-------------------------------

Ashish

Brian

Charles

David

Edward

-------------------------------

Charles

**Transactionmain.java**

import java.util.ArrayList;

import java.util.Comparator;

import java.util.List;

import java.util.Optional;

public class TransactionMain {

private static Trader Trader;

public static void main(String[] args)

{

List<Trader> l=new ArrayList<>();

l.add(new Trader("Ashish", "pune"));

l.add(new Trader("Edward", "delhi"));

l.add(new Trader("David", "pune"));

l.add(new Trader("Brian", "delhi"));

l.add(new Trader("Charles", "indore"));

List<Transaction> l1=new ArrayList<>();

l1.add(new Transaction(Trader,2011,1));

l1.add(new Transaction(Trader,2012,2));

l1.add(new Transaction(Trader,2014,3));

l1.add(new Transaction(Trader,2011,4));

l1.add(new Transaction(Trader,2015,5));

l1.stream()

.filter(t->t.getYear()==2011)

.sorted(Comparator.comparing(Transaction::getValue))

.forEach(c->System.out.println(c.getValue()));

System.out.println("<--------------------------------------------->");

l1.stream()

.filter(Trader.matches("delhi"))

.forEach(c->System.out.println(c.getValue()));

System.out.println("<--------------------------------------------->");

Optional<Transaction> tran= l1.stream()

.max(Comparator.comparing(Transaction::getValue));

System.out.println(tran);

System.out.println("<--------------------------------------------->");

Optional<Transaction> trans= l1.stream()

.min(Comparator.comparing(Transaction::getValue));

System.out.println(trans);

}

}

**Output:**

1

4

<--------------------------------------------->

[Transaction [trader=Edward, city=delhi, value=2]]

[Transaction [trader=Brian, year=delhi, value=54]]

<--------------------------------------------->

Optional[Transaction [trader=null, year=2015, value=5]]

<--------------------------------------------->

Optional[Transaction [trader=null, year=2011, value=1]]