**1. Write an application to perform basic arithmetic operations like add, subtract, multiply & divide. You need to define a functional interface first.**

**Description:-**

Define Functional Interface and write a program to perform arithmetic operations like add, subtract, multiply and divide using functional interface.

**Examples:**

**Input:-**13       5

**Output:-**

18.0      //Addition of 13 and 5

                                       8.0        //Subtraction of 13 and 5

                                       65.0      // Multiplication of 13 and 5

                                       2.6        //Division of 13 and 5

**FunctionalInterface:-**

A functional interface is an interface that contains only one abstract method. They can have only one functionality to exhibit. From Java 8 onwards, lambda expressions can be used to represent the instance of a functional interface. A functional interface can have any number of default methods. Runnable, ActionListener, Comparable are some of the examples of functional interfaces.

**Specifications:**

public class Assignment4Q1 {  
    public double addition(int num1,int num2){}  
    public double subtraction(int num1,int num2){}  
    public double multiplication(int num1,int num2){}  
    public double division(int num1,int num2){}  
    public static void main(String[] args) {}  
}

**Code:**

**import** java.util.Scanner;

**public** **class** Assignment6Q1 {

**interface** addition{

**double** add(**int** num1,**int** num2);

}

**interface** subtraction{

**double** sub(**int** num1,**int** num2);

}

**interface** multiplication{

**double** mul(**int** num1,**int** num2);

}

**interface** division{

**double** div(**int** num1,**int** num2);

}

**public** **static** addition getAddition(){

**return** (num1,num2)->{**return** num1+num2;};

}

**public** **static** subtraction getSubtraction(){

**return** (num1,num2)->{**return** num1-num2;};

}

**public** **static** multiplication getMultiplication(){

**return** (num1,num2)->{**return** num1\*num2;};

}

**public** **static** division getDivision(){

**try**{

**return** (num1,num2)->{**return** num1/num2;};

}**catch** (ArithmeticException e){

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

}

**return** **null**;

}

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

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

System.***out***.println("Enter the num1: ");

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

System.***out***.println("Enter the num2: ");

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

System.***out***.println("Addition = "+*getAddition*().add(a,b));

System.***out***.println("Subtraction = "+*getSubtraction*().sub(a,b));

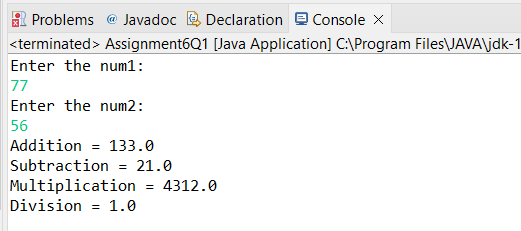
System.***out***.println("Multiplication = "+*getMultiplication*().mul(a,b));

System.***out***.println("Division = "+*getDivision*().div(a,b));

}

}

**Output:**



**Q2. Write an application using lambda expressions to print Orders having 2 criteria implemented: 1) order price more than 10000 2) order status is ACCEPTED or COMPLETED.**

**Description:**

Write a program in such a way that it has a method which returns the list of orders satisfying the 2 conditions mentioned in the question.

**Specifications:**

import java.util.ArrayList;  
  
public class Assignment4Q2 {  
  
    private int totalPrice;  
    private String status;  
  
    public static ArrayList<Assignment4Q2> listOfOrders(ArrayList<Assignment4Q2> orders) {}  
    public static void main(String[] args) {}  
}

**Code:**

**import** java.util.ArrayList;

**import** java.util.Scanner;

**public** **class** Assignment6Q2 {

**interface** listOfOrders{

**public** ArrayList<String>Result(**int** totalPrice, String status);

}

**public** **static** listOfOrders orders(){

**return** (totalPrice,status)->{

ArrayList<String>result= **new** ArrayList<>();

**if**(totalPrice>=10000){

result.add("order price more than 10000");

}

**else**{

result.add("order price less than 10000");

}

result.add(status);

**return** result;

};

}

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

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

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

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

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

String status = sc.next();

ArrayList<String>result = **new** ArrayList<>();

result = *orders*().Result(price,status);

System.***out***.println("ORDER STATUS: ");

**for** (String i: result){

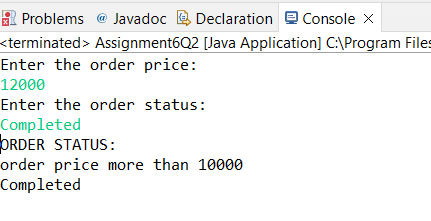
System.***out***.println(i);

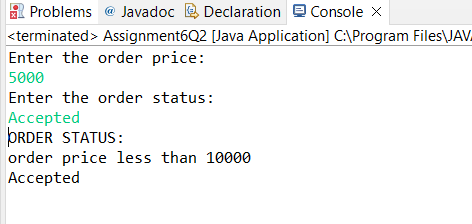
}

}

}

**OUTPUT:**

****



**Q3. Use the functional interfaces Supplier, Consumer, Predicate & Function to invoke built-in methods from Java API.**

**Description:**

Write a program using the Java API’s mentioned in the question.

**Specifications:**

public class Assignment4Q3 {  
    static void modifyValue(){  }  
    static class Product { }  
    static void display() { }  
    public static void main(String[] args) {  
}

**Q4. Remove the words that have odd lengths from the list. HINT: Use one of the new methods from JDK 8. Use removeIf() method from Collection interface.**

**Description:-**

Write a program using java 8 features which can remove the odd length words from the list.

**Specifications:**

public class Assignment4Q4 {  
    public ArrayList<String> removeOddLength(ArrayList<String> employeeList){}  
    public static void main(String[] args) { }  
}

Code:

**import** java.util.ArrayList;

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

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

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

**interface** RemoveDup

{

**public** **int** removeDup(String str);

}

**public** **class** Assignment6Q4 {

**public** ArrayList<String> removeOddLength(ArrayList<String> employeeList){

ArrayList<String> list = **new** ArrayList<>();

Predicate<String> filterEmp = (p)-> {

**if**(Math.*floorMod*(p.length(),2)==0)

**return** **true**;

**else**

**return** **false**;

};

employeeList.stream()

.filter(filterEmp)

.forEach(p->list.add(p));

**return** list;

}

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

ArrayList<String> employeeList = **new** ArrayList<>();

employeeList.add("Premal");

employeeList.add("Mihir");

employeeList.add("Rajan");

employeeList.add("Pratik");

employeeList.add("Saurabh");

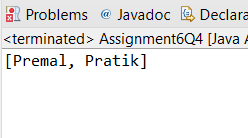
Assignment6Q4 assignment6Q4 = **new** Assignment6Q4();

System.***out***.println(assignment6Q4.removeOddLength(employeeList));

}

}

Output:



**Q5. Create a string that consists of the first letter of each word in the list of Strings provided. HINT: Use Consumer interface & a String Builder to construct the result.**

**Description:**

Write a java program using StringBuilder and Consumer interface which will return a string. The returned string should consistes of the first let of each word in the list of words.

**Specifications:**

public class Assignment4Q5 {  
    List<String> list = Arrays.*asList*("alpha", "bravo", "charlie", "delta", "echo", "foxtrot");  
  
    public static void main(String[] args) { }  
  
    public static String processWords(List<String> list) {}  
}

**Code:**

**import** java.util.Arrays;

**import** java.util.List;

**public** **class** Assignment6Q5 {

**static** List<String>*list* = Arrays.*asList*("alpha", "bravo", "charlie", "delta", "echo", "foxtrot");

**interface** ProcessWords{

**public** String processWords(List<String>list);

}

**public** **static** ProcessWords processWords(){

**return** (list)->{

String result = "";

**for** (String i: list){

result += i.charAt(0);

}

**return** result;

};

}

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

String ans;

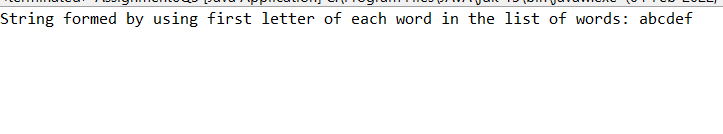
ans = *processWords*().processWords(*list*);

System.***out***.println("String formed by using first letter of each word in the list of words: "+ans);

}

}

**Output:**

****

**6. Replace every word in the list with its upper case equivalent. Use replaceAll() method & Unary Operator interface.**

Using replaceAll() method and Unary Operator interface write a java program which replaces evry word in the list with its upper case equivalent.

**Specifications:**

public class Assignment4Q6 {  
    public static void main(String[] args) {}  
    public List<String> convertToUpperCase(List<String> list) {}    
}

**Code:**

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.Locale;

**public** **class** Assignment6Q6 {

**interface** ConvertToUpperCase{

**public** List<String>convertToUpperCase(List<String>list);

}

**public** **static** ConvertToUpperCase convertToUpperCase(){

**return** (list)->{

list.replaceAll(i->String.*valueOf*(i.charAt(0)).toUpperCase()+i.substring(1));

**return** list;

};

}

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

List<String>list = Arrays.*asList*("alpha", "bravo", "charlie", "delta", "echo", "foxtrot");

System.***out***.println("List before converting first letter to uppercase: ");

**for** (String i: list){

System.***out***.println(i+" ");

}

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

list = *convertToUpperCase*().convertToUpperCase(list);

System.***out***.println("List after converting first letter to uppercase: ");

**for** (String i: list){

System.***out***.println(i+" ");

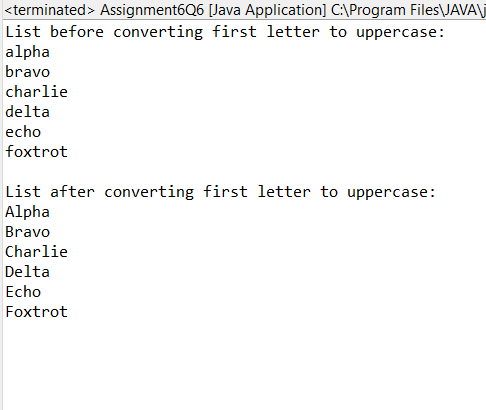
}

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

}

}

**Output:**

****

**7. Convert every key-value pair of the map into a string and append them all into a single string, in iteration order. HINT: Use Map.entrySet() method & a StringBuilder to construct the result String.**

**Description:-**

Write a java program using Map.entrySet() method & a StringBuilder which will return a string by appending all the key value pairs of a map into a single string ,in insertion order.

**Specifications:**

public class Assignment4Q7 {  
    public static void main(String[] args) {}  
    public String convertKeyValueToString(HashMap<String, Integer> map) {}  
}

**Code:**

**import** java.util.HashMap;

**import** java.util.Map;

**public** **class** Assignment6Q7 {

**interface** ConvertKeyValueToString{

**public** String convertKeyValueToString(HashMap<String, Integer>map);

}

**public** **static** ConvertKeyValueToString convertKeyValueToString(){

**return** (map)->{

String result="";

**for**(Map.Entry<String,Integer>mp: map.entrySet()){

String key = mp.getKey();

**int** value = mp.getValue();

result += key + Integer.*toString*(value);

}

**return** result;

};

}

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

HashMap<String, Integer>hashMap = **new** HashMap<>();

hashMap.put("alpha",1);

hashMap.put("bravo",2);

hashMap.put("charlie",3);

hashMap.put("delta",4);

hashMap.put("echo",5);

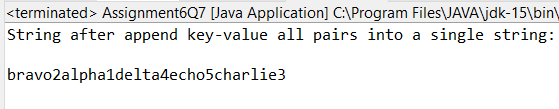
System.***out***.println("String after append key-value all pairs into a single string:\n");

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

}

}

**Output:**

****

**Q8. Create a new thread that prints the numbers from the list. Use class Thread & interface Consumer.**

**Description:-**

Write a java program which will print the list of number using Thread and interface Consumer.

**Code:**

**import** java.util.Arrays;

**import** java.util.List;

**class** Thread **extends** java.lang.Thread{

**public** **void** Display\_Numbers(List<Integer>list){

**for** (**int** i:list){

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

}

}

}

**public** **class** Assignment6Q8 {

@SuppressWarnings("deprecation")

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

Thread thread = **new** Thread();

thread.start();

List<Integer>list = Arrays.*asList*(1,2,3,4,5,6,7,8,9,10);

System.***out***.println("Printing numbers from the list: ");

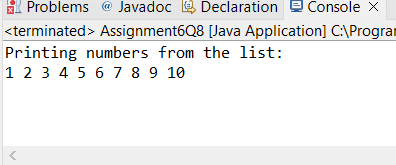
thread.Display\_Numbers(list);

thread.~~stop~~();

}

}

**Output:**

****