1. Write a program to depict the usage of the Lambda expression.

. **package** com.ust.assessment7;

//Write a program to depict the usage of the Lambda expression.

**interface** Addable{

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

}

**public** **class** Question1 {

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

// Multiple parameters in lambda expression

Addable ad1=(a,b)->(a+b);

System.***out***.println(ad1.add(10,20));

Addable ad2=(**int** a,**int** b)->(a+b);

System.***out***.println(ad2.add(100,200));

}

}



1. Write a program using the StringJoiner with example.

. package com.ust.assessment7;

import java.util.ArrayList;

import java.util.StringJoiner;

//Write a program using the StringJoiner with example.

public class Question2 {

public static void main(String[] args)

{

ArrayList<String> al = new ArrayList<>();

al.add("Ram");

al.add("Shyam");

al.add("Alice");

al.add("Bob");

StringJoiner sj1 = new StringJoiner(",");

sj1.setEmptyValue("sj1 is empty");

System.out.println(sj1);

sj1.add(al.get(0)).add(al.get(1));

System.out.println(sj1);

System.out.println("Length of sj1 : " + sj1.length());

StringJoiner sj2 = new StringJoiner(":");

sj2.add(al.get(2)).add(al.get(3));

sj1.merge(sj2);

System.out.println(sj1.toString());

System.out.println("Length of new sj1 : " + sj1.length());

}

}



1. Write a program to depict the Optional class.

. **package** com.ust.assessment7;

**import** java.util.Optional;

//Write a program to depict the Optional class.

**public** **class** Question3 {

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

String[] str = **new** String[10];

Optional<String> checkNull = Optional.*ofNullable*(str[5]);

**if**(checkNull.isPresent()){

String lowercaseString = str[5].toLowerCase();

System.***out***.print(lowercaseString);

}**else**

System.***out***.println("string value is not present");

}

}



1. Show case the usage of forEach()

. package com.ust.assessment7;

import java.util.ArrayList;

import java.util.List;

//Show case the usage of forEach()

public class Question4 {

public static void main(String[] args) {

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

gamesList.add("Football");

gamesList.add("Cricket");

gamesList.add("Chess");

gamesList.add("Hocky");

System.out.println("------------Iterating by passing method reference---------------");

gamesList.forEach(System.out::println);

}

}



1. Write a program to showcase the usage of streams and method references.

. package com.ust.assessment7;

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Collectors;

//Write a program to showcase the usage of streams and method references.

interface Sayable{

void say();

}

class Product{

int id;

String name;

float price;

public Product(int id, String name, float price) {

this.id = id;

this.name = name;

this.price = price;

}

}

public class Question5 {

public static void saySomething(){

System.out.println("Hello, this is static method.");

}

public static void main(String[] args) {

// Referring static method

Sayable sayable = Question5::saySomething;

// Calling interface method

sayable.say();

List<Product> productsList = new ArrayList<Product>();

//Adding Products

productsList.add(new Product(1,"HP Laptop",25000f));

productsList.add(new Product(2,"Dell Laptop",30000f));

productsList.add(new Product(5,"Apple Laptop",90000f));

List<Float> productPriceList2 =productsList.stream()

.filter(p -> p.price > 30000)// filtering data

.map(p->p.price) // fetching price

.collect(Collectors.toList()); // collecting as list

System.out.println(productPriceList2);

}

}

