FunctionalInterface

1. Given an ArrayList with 5 Employee(id**,**name**,**location**,**salary) objects**,**

write a program to extract the location details of each Employee and store it in an ArrayList**,** with the help of Function**.**

**CODE**

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

@FunctionalInterface

interface MyFunction {

    String apply(Employee emp);

}

class Employee {

*int* id;

    String name;

    String location;

*double* salary;

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

*this*.*id* = id;

*this*.*name* = name;

*this*.*location* = location;

*this*.*salary* = salary;

    }

}

*public* class EmployeeLocation {

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

        ArrayList<Employee> employees = new ArrayList<>();

        employees.add(new Employee(1, "Vinnu", "Hyderabad", 50000));

        employees.add(new Employee(2, "Abde", "Bangalore", 55000));

        employees.add(new Employee(3, "Chinni", "Pune", 52000));

        employees.add(new Employee(4, "Abhi", "Delhi", 51000));

        employees.add(new Employee(5, "Shree", "Mumbai", 53000));

        MyFunction getLocation = (Employee emp) -> emp.*location*;

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

        for (Employee emp : employees) {

            locations.add(getLocation.apply(emp));

        }

        System.*out*.println("Locations:");

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

    }

}

**OUTPUT:**

Locations:

Hyderabad

Bangalore

Pune

Delhi

Mumbai

1. Given an ArrayList containing 10 numbers**,** write a program to calculate the sum of all the elements**,** with the help of Function**.**

**CODE:**

*import* java.util.ArrayList;

*import* java.util.Random;

@FunctionalInterface

interface Sum{

*int* sum(ArrayList<Integer> list);

}

*public* class ArraySum{

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

        Random rand = new Random();

        ArrayList<Integer> list = new ArrayList<>();

        for(*int* i = 1; i <= 10; i++) {

            list.add(rand.nextInt(1000));

        }

        Sum s  = ArraySum::display;

*int* res = s.sum(list);

        System.*out*.println("Sum of elements in the list: " + res);

    }

*public* *static* *int* display(ArrayList<Integer> list) {

*int* sum = 0;

        for(*int* num : list) {

            sum += num;

        }

        return sum;

    }

}

**OUTPUT:**

Sum of elements in the list: 7094

1. Given an ArrayList containing 10 words**,** write a program to filter the words which are palindrome**,** with the help of Predicate**.**

**CODE:**

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

*import* java.util.function.Predicate;

*import* java.util.stream.Collectors;

@FunctionalInterface

interface PalindromeCheck {

*boolean* isPalindrome(String s);

}

*public* class PalindromeExample {

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

        List<String> words = Arrays.asList(

            "madam", "hello", "racecar", "world",

            "level", "java", "noon", "python", "civic", "radar"

        );

        PalindromeCheck palindromeCheck =

            word -> word.equalsIgnoreCase(new StringBuilder(word).reverse().toString());

        Predicate<String> isPalindrome = palindromeCheck::isPalindrome;

        List<String> palindromeWords = words.stream()

                .filter(isPalindrome)

                .collect(Collectors.toList());

        System.*out*.println("Palindrome words: " + palindromeWords);

    }

}

**OUTPUT:**

Palindrome words: [madam, racecar, level, noon, civic, radar]

1. Create an Employee class with three *private* variables id, name and salary.

Create getters & setters and a parameterized constructor.

Create an ArrayList and store 5 to 10 Employee objects in it.

Write a program using Predicate, to filter and display the name of the employees whose salary is less than 10000.

**CODE:**

*import* java.util.ArrayList;

*import* java.util.List;

*import* java.util.function.Predicate;

*import* java.util.stream.Collectors;

class EmployeeDetails{

*private* *int* id;

*private* String name;

*private* *int* salary;

*public* EmployeeDetails(*int* id, String name, *int* salary) {

*this*.*id* = id;

*this*.*name* = name;

*this*.*salary* = salary;

    }

*public* *int* getId() {

        return id;

    }

*public* String getName() {

        return name;

    }

*public* *int* getSalary() {

        return salary;

    }

*public* *void* setId(*int* id) {

*this*.*id* = id;

    }

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

*this*.*name* = name;

    }

*public* *void* setSalary(*int* salary) {

*this*.*salary* = salary;

    }

}

*public* class EmployeeName {

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

        ArrayList<EmployeeDetails> employees = new ArrayList<>();

        employees.add(new EmployeeDetails(1, "vinnu", 1000));

        employees.add(new EmployeeDetails(2, "abde", 20000));

        employees.add(new EmployeeDetails(3, "chinni", 30000));

        employees.add(new EmployeeDetails(4, "abhi", 4000));

        Predicate<EmployeeDetails> filterByName = emp -> emp.getSalary()<10000;

        List<String> names = employees.stream()

        .filter(filterByName)

        .map(EmployeeDetails::getName)

        .collect(Collectors.toList());

        System.*out*.println(names);

    }

}

**OUTPUT:**

[vinnu, abhi]

1. Given an ArrayList containing 10 numbers**,** write a program to filter the perfect square numbers**.**

Example **for** perfect square numbers**:** 0**,** 1**,** 4**,** 9**,** 16**,** 25**,** 36**,** 49**,** 64 etc**..**

**CODE:**

*import* java.util.ArrayList;

*import* java.util.Arrays;

*import* java.util.List;

*import* java.util.stream.Collectors;

*public* class PerfectSquare{

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

        ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(1,2,4,6,9,10,16,20,25));

        List<Integer> perfectsquare = numbers.stream()

            .filter(n -> Math.sqrt(n) \* Math.sqrt(n) == n)

            .collect(Collectors.toList());

        System.*out*.println("Perfect squares in the list: " + perfectsquare);

    }

}

**OUTPUT:**

Perfect squares in the list: [1, 4, 9, 16, 25]

1. Given an ArrayList containing 10 words, write a program to reverse each word and

update the same ArrayList , with the help of Consumer.

**CODE:**

*import* java.util.ArrayList;

*import* java.util.Arrays;

*import* java.util.function.Consumer;

*public* class ReverString {

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

        ArrayList<String> words = new ArrayList<>(Arrays.asList("Apple", "Banana", "Cherry", "Date"));

        Consumer<String> reverseAndPrint = word -> {

                String reversed = new StringBuilder(word).reverse().toString();

                System.*out*.println(reversed);

        };

        words.forEach(reverseAndPrint);

    }

}

**OUTPUT:**

elppA

ananaB

yrrehC

etaD

1. Given an ArrayList containing 10 numbers, write a program using Consumer methods to display each number and whether is it odd or even. Example: For number 2, it should print "2 even" For number 5, it should print "5 odd"

**CODE:**

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

*import* java.util.function.Consumer;

*public* class OddEvenConsumer {

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

        ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 5));

        Consumer<Integer> displayOddEven = num -> {

            if (num % 2 == 0)

                System.*out*.println(num + " even");

            else

                System.*out*.println(num + " odd");

        };

        numbers.forEach(displayOddEven);

    }

}

**OUTPUT:**

1 odd

2 even

3 odd

4 even

5 odd

1. Write a program using Supplier, which generates and returns an ArrayList containing first 10 prime numbers.

**CODE:**

*import* java.util.ArrayList;

*import* java.util.function.Supplier;

*public* class PrimeSupplierExample {

*private* *static* *boolean* isPrime(*int* n) {

        if (n <= 1) return false;

        for (*int* i = 2; i <= Math.sqrt(n); i++) {

            if (n % i == 0) return false;

        }

        return true;

    }

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

        Supplier<ArrayList<Integer>> primeSupplier = () -> {

            ArrayList<Integer> primes = new ArrayList<>();

*int* num = 2;

            while (primes.size() < 10) {

                if (isPrime(num)) {

                    primes.add(num);

                }

                num++;

            }

            return primes;

        };

        ArrayList<Integer> primeNumbers = primeSupplier.get();

        System.*out*.println(primeNumbers);

    }

}

**OUTPUT:**

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29]