Test Assignment - By Pankaj Ku Chouhan

1. Given a string and an int n, return a string made of the first n characters of the string, followed by the first n-1 characters of the string, and so on. You may assume that n is between 0 and the length of the string, inclusive (i.e. n >= 0 and n <= str.length()). repeatFront("Chocolate", 4) → "ChocChoChC" repeatFront("Chocolate", 3) → "ChoChC" repeatFront("Ice Cream", 2) → "IcI" public String repeatFront(String str, int n) { }

Program:

| package com.techment.two;  class RepeatFront{  private String words;  //created function taking string input and limit returning string  String RepeatImpl(String words,int limit)  {  String str = "";  int k=0;  for(int i = limit;i>=0;i--)  {  //extracting string here and concatenate simultaneously  str+=words.substring(k,i);  }      return str;    }   }    public class RepeatFrontByN {   public static void main(String[] args) {  *// TODO Auto-generated method stub*   // creating obj  RepeatFront obj = new RepeatFront();  System.out.println("Chocolate for 4 Chars is " + obj.RepeatImpl("Chocolate", 4));  System.out.println("Chocolate for 3 Chars is " + obj.RepeatImpl("Chocolate", 3));  System.out.println("Icecream for 2 Chars is " + obj.RepeatImpl("Icecream", 2));                }  } |
| --- |

Output:

| Chocolate for 4 Chars is ChocChoChC Chocolate for 3 Chars is ChoChC Icecream for 2 Chars is IcI |
| --- |

2 ) Given a string and an int n, return a string made of n repetitions of the last n characters

of th

e string. You may assume that n is between 0 and the length of the string, inclusive.

repeatEnd("Hello", 3)

→

"llollollo"

repeatEnd("Hello", 2)

→

"lolo"

repeatEnd("Hello", 1)

→

"o"

public String repeatEnd(String str, int n) {

}

Program :

| package com.techment.two;  class RepeatFront{  private String words;    String RepeatImpl(String words,int limit)  {  String str = "";  int k=0;  for(int i = limit;i>=0;i--)  {   str+=words.substring(k,i);  }      return str;    }   }    public class RepeatFrontByN {   public static void main(String[] args) {  *// TODO Auto-generated method stub*     RepeatFront obj = new RepeatFront();  System.out.println("Chocolate for 4 Chars is " + obj.RepeatImpl("Chocolate", 4));  System.out.println("Chocolate for 3 Chars is " + obj.RepeatImpl("Chocolate", 3));  System.out.println("Icecream for 2 Chars is " + obj.RepeatImpl("Icecream", 2));                }  } |
| --- |

Output :

| Chocolate for 4 Chars is ChocChoChC Chocolate for 3 Chars is ChoChC Icecream for 2 Chars is IcI |
| --- |

3 ) ArrayList containing

the firstName,lastName and weight of the wrestler that will be used to create a

stream

Task:

create a WWE class which has the following private member

firstName String,

lastName String,

weight int

>Define parameterized Constructor

-

>

Define Setter and getter

Create a WWEWrestlerImplementation class which performs operation as per given requirement using

stream api

a)count the number of wrestlers

b)sum of all weight of all wrestlers whose weight is above 200

c) create a meth

od printFirstName(List <WWE> list) that returns the first name of the all wrestler

Program:

| package com.techment.third;  import java.util.ArrayList; import java.util.Comparator; import java.util.List; import java.util.stream.Collectors;  class WWE {  //Creating private variable here  private String firstName;  private String lastName;  private long weight;  // Constructor here  public WWE(String firstName, String lastName, long weight) {  super();  this.firstName = firstName;  this.lastName = lastName;  this.weight = weight;  }  // Getters and Setters  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 long getWeight() {  return weight;  }   public void setWeight(long weight) {  this.weight = weight;  }  }  // Here i have done all Implemetations  public class WWEWrestlerImplementation {   public static void main(String[] args) {   ArrayList<WWE> wrestlers = new ArrayList<WWE>();   WWE wrestler1 = new WWE("Great", "Khali", 400);  WWE wrestler2 = new WWE("The", "Undertaker", 390);  WWE wrestler3 = new WWE("Jhon", "Cena", 189);  WWE wrestler4 = new WWE("Roman", "Reigns", 78);  WWE wrestler5 = new WWE("Randy", "Orton", 267);  WWE wrestler6 = new WWE("Kurt", "Angle", 99);   wrestlers.add(wrestler1);  wrestlers.add(wrestler2);  wrestlers.add(wrestler3);  wrestlers.add(wrestler4);  wrestlers.add(wrestler5);  wrestlers.add(wrestler6);   *// Counting the Number of Wrestlers*   *//Numbers of Wrestlers*   long total = wrestlers.stream().count();  System.out.println("Total wrestlers are : " + total);    *//Total weights of wrestlers above 200*   long totalWeights = wrestlers.stream().filter(i->(i.getWeight()>=200)).collect(Collectors.summarizingLong(WWE::getWeight)).getSum();  System.out.println("Total weights of wrestlers having more than 200 : " + totalWeights);  // Here i am Printing all wrestlers by firstname  System.out.println("\nFirst name of all wrestlers is ");  List<WWE> wrestlersList = new ArrayList(wrestlers);    WWEWrestlerImplementation obj = new WWEWrestlerImplementation();    List<String> firstNameList = obj.printFirstName(wrestlersList);    for(String f:firstNameList)  {  System.out.println(f);  }    System.out.println("\nMinimum weight among wrestlers is ");  WWE minWeight = obj.minimumWeight(wrestlersList);  System.out.println("Name : "+minWeight.getFirstName()+" "+minWeight.getLastName()+" weights :"+minWeight.getWeight());  }          List<String> printFirstName(List<WWE> wrestlersList)  {  return wrestlersList.stream().map(i->i.getFirstName()) .collect(Collectors.toList());  }    WWE minimumWeight(List<WWE> wrestlersList)  {    return wrestlersList.stream().min(Comparator.comparingLong(WWE::getWeight)).get();    }        } |
| --- |

Output :

| Total wrestlers are : 6 Total weights of wrestlers having more than 200 : 1057  First name of all wrestlers is  Great The Jhon Roman Randy Kurt  Minimum weight among wrestlers is  Name : Roman Reigns weights :78 |
| --- |

4 ) ) : Write a java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On entering the choice, an appropriate message with “stop” or “ready” or “go” should appear in the console .Initially there is no message shown

Program :

| package com.techment.fourth;  import java.util.Scanner;  public class TrafficLight {   public static void main(String[] args) {  *// TODO Auto-generated method stub*   System.out.println("Enter 1 for Red\nEnter 2 for Yellow\nEnter 3 for Green\n ");  Scanner scanner = new Scanner(System.in);  int choice = scanner.nextInt();    switch(choice)  {  case 1:  System.out.println("Stop");  break;  case 2:  System.out.println("ready");  break;    default:  System.out.println("go");  break;  }      }  } |
| --- |

Output :

| Enter 1 for Red Enter 2 for Yellow Enter 3 for Green   3 go |
| --- |

5 ) Create a method which accepts the id and the age of people as a Map and decide if they are eligible for vote. A person is eligible for vote if his age is greater than 18. Add the IDs of all the eligible persons to list and return the list.

Program :

| package com.techment.five;  import java.text.ParseException; import java.text.SimpleDateFormat; import java.time.LocalDate; import java.util.ArrayList; import java.util.HashMap; import java.util.List; import java.util.Map; import java.util.Map.Entry; import java.util.Scanner;  class Person {  private int id;  private String age;  //Constructors   public Person(int id, String age) {  super();  this.id = id;  this.age = age;  }  //Getters and Setters  public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   public String getAge() {  return age;  }   public void setAge(String age) {  this.age = age;  }  }  public class EligiblePerson {   public static void main(String[] args) throws ParseException {  *// TODO Auto-generated method stub*  try {  // Initialised new Empty Map here  Map<Integer, String> allPersonsMap = new HashMap<Integer, String>();  EligiblePerson obj = new EligiblePerson();   Scanner scanner = new Scanner(System.in);  // taking input here  System.out.println("Enter number of persons ");  int persons = scanner.nextInt();   for (int i = 0; i < persons; i++) {  System.out.println("Enter id of " + (i + 1) + " person");  int id = scanner.nextInt();   System.out.println("Enter dob in this format (DD/MM/YYYY) of " + (i + 1) + " person");  String age = scanner.next();   allPersonsMap.put(id, age);   }    // Assigning the lists here  List<Integer> allAge = obj.votersList(allPersonsMap);      System.out.println("List of eligible list are ");  for (Integer s : allAge) {  System.out.println(s);  }          } catch (Exception e) {  System.out.println(e);  }   }  List<Integer> allAge = new ArrayList<Integer>();   private List<Integer> votersList(Map<Integer, String> allPersonsMap) throws ParseException {   for (Entry<Integer, String> entry : allPersonsMap.entrySet())  {  int today = new SimpleDateFormat("dd/MM/yyyy").parse("18/07/2021").getYear();  int yourAge = new SimpleDateFormat("dd/MM/yyyy").parse(entry.getValue()).getYear();  // Comparing here two dates present and user input  if(today-yourAge>=18)  {  // adding here into the list  allAge.add(entry.getKey());  }    }  return allAge;  }  } |
| --- |

Output :

Enter number of persons

3

| Enter id of 1 person 1 Enter dob in this format (DD/MM/YYYY) of 1 person 12/03/2019 Enter id of 2 person 2 Enter dob in this format (DD/MM/YYYY) of 2 person 03/04/1992 Enter id of 3 person 3 Enter dob in this format (DD/MM/YYYY) of 3 person 12/03/2008 List of eligible list are  2 |
| --- |

6 ) Exception Handling Declare a class Called Person Fields id,name,age

a)Validate the age of a person and display proper message by using user defined exception. Age of a person should be above 15. b)if the size of name is less than 3 throw user defined Exception. Take values from user input

Program :

| package com.techment.six;  import java.util.Scanner;  class Person{  private int id;  private String name;  private int age;  public Person(int id, String name, int age) {  super();  this.id = id;  this.name = name;  this.age = age;  }  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 int getAge() {  return age;  }  public void setAge(int age) {  this.age = age;  }     }  class InvalidAgeException extends Exception  {   public InvalidAgeException (String str)   {   *// calling the constructor of parent Exception*   super(str);   }  }    class InvalidNameException extends Exception  {   public InvalidNameException (String str)   {   *// calling the constructor of parent Exception*   super(str);   }  }   public class AgeValidate {   public static void main(String[] args) throws InvalidNameException, InvalidAgeException {  *// TODO Auto-generated method stub*     try {  Scanner scanner = new Scanner(System.in);  System.out.println("Enter id ");  int id = scanner.nextInt();    System.out.println("Enter name ");    String name = scanner.nextLine();  name+=scanner.nextLine();      if(name.length()<=3)  throw new InvalidNameException("Name should be more than 3 chars ");    System.out.println("Enter age ");  int age = scanner.nextInt();    if(age==0 || age<15)  throw new InvalidAgeException("Age should be more than 15");    Person person = new Person(id,name,age);    System.out.println("Name is "+person.getName());  System.out.println("Age is "+person.getAge());      }catch(InvalidAgeException e) {  System.out.println(e.getMessage());  }catch(InvalidNameException e)  {  System.out.println(e.getMessage());  }catch(Exception e)  {  System.out.println(e.getMessage());  }      }  } |
| --- |

Output 1 : Checking name exception here

| Enter name  Pan Name should be more than 3 chars |
| --- |

Output 2 : Checking age here

| Enter id  1 Enter name  Pankaj Enter age  12 Age should be more than 15 |
| --- |

Output 3 : All Successfull

| Enter id  1 Enter name  Pankaj Enter age  22 Name is Pankaj Age is 22 |
| --- |

7 ) Create an ArrayList, Divide the ArrayList into 2 ArrayList .if the size of ArrayList is 10 means it should divide 5 and 5 each. If the size is 11 means it should divide 6 and 5.and find the maximum number from the divide list.

Program :

| package com.techment.seven;  import java.util.ArrayList; import java.util.Scanner;    public class ArrayListDivide {   public static void main(String[] args) {  *// TODO Auto-generated method stub*   Scanner scanner = new Scanner(System.in);  System.out.println("Enter the size ");  int size = scanner.nextInt();   ArrayList<Integer> original = new ArrayList<Integer>();   for (int i = 0; i < size; i++) {  System.out.println("Enter data");  int data = scanner.nextInt();   original.add(data);  }     ArrayList<Integer> firstArray = new ArrayList<Integer>(original.subList(0, (size) / 2));  ArrayList<Integer> secondArray = new ArrayList<Integer>(original.subList((size) / 2,size));   int max1 = firstArray.stream().max(Integer::compare).get();  int max2 = secondArray.stream().max(Integer::compare).get();  System.out.println(max1+" "+max2);   }  } |
| --- |

Output :

| Enter the size  10 Enter data 23 Enter data 243 Enter data 565 Enter data 76 Enter data 19 Enter data 76 Enter data 45 Enter data 64 Enter data 35 Enter data 86 565 86 |
| --- |