**public** **class** ReverseANumber\_1 {

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

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

System.***out***.println("Enter the number to be reversed");

**int** num = s.nextInt();

**int** rev = 0;

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

rev = rev \* 10 + num % 10;

num = num / 10;

}

System.***out***.println("The reversed number is " + rev);

}

}

**public** **class** ReverseANumber\_2 {

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

@SuppressWarnings("resource")

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

System.***out***.println("Enter the number to be reversed");

**int** num = s.nextInt();

// Using StringBuffer class

StringBuffer sb = **new** StringBuffer(String.*valueOf*(num));

//StringBuffer rev = sb.reverse();

System.***out***.println("The reversed number is " + sb.reverse());

}

}/\*

Enter the number to be reversed

1234

The reversed number is 4321\*/

**public** **class** ReverseANumber\_3 {

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

@SuppressWarnings("resource")

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

System.***out***.println("Enter the number to be reversed");

**int** num = s.nextInt();

StringBuilder sbl= **new** StringBuilder(String.*valueOf*(num));

//StringBuilder rev= sbl.reverse();

System.***out***.println("The reversed number is " + sbl.reverse());

}

}

/\*Enter the number to be reversed

4321

The reversed number is 1234\*/

**public** **class** ReverseInteger {

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

String a="12345",b = "";

//int a;

**for** (**int** i=(a.length()-1); i>=0; i--) {

b=b+a.charAt(i);

}

//long c=Integer.parseInt(b);

**int** c=Integer.*parseInt*(b);

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

}

}================

**public** **class** RoboClassDemo {

**static** {

//System.setProperty("webdriver.gecko.driver",".//Software//geckodriver.exe");

System.*setProperty*("webdriver.chrome.driver", ".//Software//chromedriver.exe");

}

**public** **static** **void** main(String[] args) **throws** InterruptedException, AWTException {

//WebDriver driver = new FirefoxDriver();

WebDriver driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("https://www.google.com/");

Thread.*sleep*(3000);

WebElement SearchBox = driver.findElement(By.*name*("q"));

SearchBox.sendKeys("Java", Keys.***ENTER***);

Thread.*sleep*(2000);

// locating the Java WebElement on the right

WebElement Java\_WebElmnt = driver.findElement(By.*xpath*("//div[@class='kno-ecr-pt kno-fb-ctx PZPZlf gsmt']/span[contains(.,'Java')]"));

String Java\_txt = Java\_WebElmnt.getText();

System.***out***.println(Java\_txt);

Thread.*sleep*(2000);

// Double clicking on the Java text to copy it

Actions a = **new** Actions(driver);

a.doubleClick(Java\_WebElmnt).perform();

// Creating an object r of the Robot class to perform keyboard actions

Robot r = **new** Robot();

r.keyPress(KeyEvent.***VK\_CONTROL***); // pressing the control key

Thread.*sleep*(1000);

r.keyPress(KeyEvent.***VK\_C***); // Pressing the C key(ctrl+C)

Thread.*sleep*(1000);

r.keyPress(KeyEvent.***VK\_T***); // To open a new tab

Thread.*sleep*(1000);

r.keyPress(KeyEvent.***VK\_V***); // To paste the copied text

Thread.*sleep*(1000);

r.keyPress(KeyEvent.***VK\_ENTER***); // Hitting the enter key

// Releasing the keys pressed

r.keyRelease(KeyEvent.***VK\_CONTROL***); // pressing the control key

Thread.*sleep*(1000);

r.keyRelease(KeyEvent.***VK\_C***); // Pressing the C key(ctrl+C)

Thread.*sleep*(1000);

r.keyRelease(KeyEvent.***VK\_T***); // To open a new tab

Thread.*sleep*(1000);

r.keyRelease(KeyEvent.***VK\_V***); // To paste the copied text

Thread.*sleep*(1000);

r.keyRelease(KeyEvent.***VK\_ENTER***);

driver.close();

}

}

**public** **class** throwException\_ {

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

{

// Use of unchecked Exception

**try** {

**double** x=3/0;

**throw** **new** ArithmeticException();

}

**catch** (ArithmeticException e)

{

e.printStackTrace();

System.***out***.println("inside catch");

}

**finally**{

System.***out***.println("inside finally");

}

}

}

**public** **class** tooltipDemo {

**public** **static** WebDriver *driver*;

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

//Set system properties for geckodriver This is required since Selenium 3.0

System.*setProperty*("webdriver.chrome.driver",

"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe");

// Create a new instance of the Firefox driver

*driver* = **new** ChromeDriver();

//CASE 2 : Using Actions class method

*driver*.get("http://demoqa.com/tooltip-and-double-click/");

System.***out***.println("demoqa webpage Displayed");

//Maximise browser window

*driver*.manage().window().maximize();

//Instantiate Action Class

Actions actions = **new** Actions(*driver*);

//Retrieve WebElement

WebElement element = *driver*.findElement(By.*id*("tooltipDemo"));

// Use action class to mouse hover

actions.moveToElement(element).perform();

//WebElement toolTip = driver.findElement(By.cssSelector(".tooltiptext"));

String toolTip = element.getAttribute("title");

// To get the tool tip text and assert

//String toolTipText = toolTip.getText();

System.***out***.println("toolTipText-->"+toolTip);

//Verification if tooltip text is matching expected value

**if**(toolTip.equalsIgnoreCase("We ask for your age only for statistical purposes.")){

System.***out***.println("Pass\* : Tooltip matching expected value");

}**else**{

System.***out***.println("Fail : Tooltip NOT matching expected value");

}

// Close the main window

*driver*.close();

}

}---------

**public** **class** toolTipDemo2 {

**public** **static** WebDriver *driver*;

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

//Set system properties for geckodriver This is required since Selenium 3.0

System.*setProperty*("webdriver.chrome.driver",

"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe");

// Create a new instance of the Firefox driver

*driver* = **new** ChromeDriver();

//CASE 2 : Using Actions class method

//driver.get("https://accounts.google.com/signup/v2/webcreateaccount?service=mail&continue=https%3A%2F%2Fmail.google.com%2Fmail%2F&ltmpl=default&flowName=GlifWebSignIn&flowEntry=SignUp");

/\*WebElement logo=driver.findElement(By.id("logo"));

//Instantiate Action Class

Actions actions = new Actions(driver);

actions.moveToElement(logo).build().perform();

String toolTip= logo.getAttribute("title");

System.out.println("ToolTip of google logo is = "+toolTip);\*/

*driver*.get("https://mail.google.com/mail/u/2/#inbox");

//Maximise browser window

*driver*.manage().window().maximize();

Thread.*sleep*(4000);

// closing the popup

WebElement imgPrimary= *driver*.findElement(By.*xpath*("//div[@class='aKz'][contains(text(),'Primary')]"));

Actions actions = **new** Actions(*driver*);

actions.moveToElement(imgPrimary).build().perform();

String tooltip=imgPrimary.getAttribute("title");

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

}

}

**public** **class** WriteToFile {

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

FileWriter filewriter = **new** FileWriter("E:\\\\\\\\MyWorkspace\_Completed\\\\\\\\ReadFromTxtFile.txt");

BufferedWriter bufferedWriter = **new** BufferedWriter(filewriter);

bufferedWriter.write("Dont? Close the BufferedWriter object to prevent memory leak");

bufferedWriter.close();

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

}

}

/\*

\* String textToWrite = "I am learning Java from SoftwareTestingMaterial"; //

\* Import BufferedWriter and FileWriter classes BufferedWriter writes text to a

\* character-output stream, buffering characters so as to provide for the

\* efficient writing of single characters, arrays, and strings. //Import

\* FileWriter and BufferedWriter class BufferedWriter writer = new

\* BufferedWriter(new

\* FileWriter("E:\\MyWorkspace\_Completed\\ReadFromTxtFile.txt"));

\* writer.write(textToWrite); // Close the BufferedWriter object to prevent

\* memory leak writer.close(); } }

\*/========

**public** **class** impthirdLargestUsingSorting {

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

//There should be atleast 3 elements in an array

**int** arr[] = { 14, 46, 47, 86, 92, 52, 48, 36, 66, 85 };

**int** size=arr.length;

Arrays.*sort*(arr);

System.***out***.println( "sorted Array is :" + arr);

**int** max = arr[size-3];

System.***out***.println("Largest element is : "+arr[size-1]);

System.***out***.println("2nd largest element is :"+arr[size-2]);

System.***out***.println("3rd largest element is :"+max);

System.***out***.println("Smallest element is : "+arr[size-size]);

System.***out***.println("2nd Smallest element is : "+((arr[size-(size-1)])));

System.***out***.println("3rd Smallest element is : "+((arr[size-(size-2)])));

}

}=============

**public** **class** MaxAndMin\_ValueofArray {

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

**int** a[]= {50,60,30,20,40,70,10};

//int max=a[0];

**int** min=a[0];

**for**(**int** i=1; i<a.length;i++)

{

//if(a[i]>max)

**if**(a[i]<min)

{

//max=a[i];

min=a[i];

}

}

//System.out.println("Maximum element of array is = "+max);

System.***out***.println("Minimum element of array is = "+min);

}

}

**public** **class** SecondLargestNumbInArray {

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

**int** arr[] = { 14, 46, 47, 86, 92, 52, 48, 36, 66, 85 };

**int** largest = arr[0];

**int** secondLargest = arr[0];

System.***out***.println("The given array is:");

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

System.***out***.print(arr[i] + "\t");

}

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

**if** (arr[i] > largest) {

secondLargest = largest;

largest = arr[i];

} **else** **if** (arr[i] > secondLargest) {

secondLargest = arr[i];

}

}

System.***out***.println("\nSecond largest number is:" + secondLargest);

}

}===========

**public** **class** thirdLargestArray {

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

**int** largest, second, third;

// There should be atleast 3 elements in an array

**int** arr[] = { 14, 46, 47, 86, 92, 52, 48, 36, 66, 85 };

largest = arr[0];

System.***out***.println("The given array is:");

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

System.***out***.print(arr[i] + "\t");

}

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

**if** (arr[i] > largest) {

largest = arr[i];

}

// Find secondLargest

second = arr[0];

**for** (**int** j = 0; j < arr.length; j++) {

**if** (arr[j] > second && arr[j] < largest) {

second = arr[j];

}

// Find thirdLargest

third = arr[0];

**for** (**int** l = 0; l < arr.length; l++) {

**if** (arr[l] > third && arr[l] < second) {

third = arr[l];

}

// System.out.println("\n third largest number is:" + third);

}

System.***out***.println("\n Number are:" + largest + " " + second + " " + third);

}

}

}

}

**public** **class** ThirdLargestNumberInAnArray {

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

**int** temp, size;

**int** arr[] = {10, 20, 25, 63, 96, 57};

size = arr.length;

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

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

**if**(arr[i]>arr[j]){

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

//System.out.println("The Array is " + arr[]);

System.***out***.println("Third largest number is:: "+arr[size-3]);

System.***out***.println("Second largest number is:: "+arr[size-2]);

System.***out***.println("Largest number is:: "+arr[size-1]);

}

}

**public** **class** RoboClassDemo2 {

**static** {

//System.setProperty("webdriver.chrome.driver", ".\\Software\\chromedriver.exe");

System.*setProperty*("webdriver.chrome.driver", ".//Software//chromedriver.exe");

}

**public** **static** **void** main(String[] args) **throws** InterruptedException, AWTException {

WebDriver driver = **new** ChromeDriver(); // upcasting

driver.manage().window().maximize();

driver.get("https://www.google.com/");

Thread.*sleep*(2000);

WebElement SrchBox = driver.findElement(By.*name*("q"));

SrchBox.sendKeys("Java", Keys.***ENTER***);

Thread.*sleep*(2000);

// this is the Java icon on the right side.

WebElement we = driver.findElement(By.*xpath*("//div[@class='kno-ecr-pt PZPZlf gsmt']/span[contains(.,'Java')]"));

//"//div[@class='kno-ecr-pt PZPZlf gsmt']/span[contains(.,'Java')]"

String java = driver.findElement(By.*xpath*("//div[@class='kno-ecr-pt kno-fb-ctx PZPZlf gsmt']")).getText();

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

Actions a = **new** Actions(driver);

a.doubleClick(we).perform();

// Creating an object r of the Robot class

Robot r = **new** Robot();

r.keyPress(KeyEvent.***VK\_CONTROL***);

Thread.*sleep*(2000);

r.keyPress(KeyEvent.***VK\_C***); // ctrl+C copy the text.

Thread.*sleep*(2000);

r.keyPress(KeyEvent.***VK\_T***); //open a new tab

Thread.*sleep*(5000);

r.keyPress(KeyEvent.***VK\_V***); //paste the copied text

Thread.*sleep*(5000);

r.keyPress(KeyEvent.***VK\_ENTER***);

Thread.*sleep*(4000);

// Release all the objects

r.keyRelease(KeyEvent.***VK\_ENTER***);

r.keyRelease(KeyEvent.***VK\_V***);

r.keyRelease(KeyEvent.***VK\_T***);

r.keyRelease(KeyEvent.***VK\_C***);

r.keyRelease(KeyEvent.***VK\_CONTROL***);

}

}

============

**public** **class** ThirdLargestNumberInAnArray {

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

**int** temp, size;

**int** arr[] = {10, 20, 25, 63, 96, 57};

size = arr.length;

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

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

**if**(arr[i]>arr[j]){

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

//System.out.println("The Array is " + arr[]);

System.***out***.println("Third largest number is:: "+arr[size-3]);

System.***out***.println("Second largest number is:: "+arr[size-2]);

System.***out***.println("Largest number is:: "+arr[size-1]);

}

}=====================

METHOD OVERRIDING

**class** Bank {

**int** getRateOfInterest() {

**return** 0;

}

}

**class** SBI **extends** Bank {

**int** getRateOfInterest() {

**return** 9;

}

}

**class** ICICI **extends** Bank {

**int** getRateOfInterest() {

**return** 10;

}

}

**class** AXIS **extends** Bank {

**int** getRateOfInterest() {

**return** 11;

}

}

**class** MethodOverriding2 {

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

{

SBI s = **new** SBI();

ICICI i = **new** ICICI();

AXIS a = **new** AXIS();

System.***out***.println("SBI Rate of Interest: " + s.getRateOfInterest());

System.***out***.println("ICICI Rate of Interest: " + i.getRateOfInterest());

System.***out***.println("AXIS Rate of Interest: " + a.getRateOfInterest());

}

}

/\*SBI Rate of Interest: 9

ICICI Rate of Interest: 10

AXIS Rate of Interest: 11\*/

**public** **class** MaxAndMin\_ValueofArray {

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

**int** a[]= {50,60,30,20,40,70,10};

//int max=a[0];

**int** min=a[0];

**for**(**int** i=1; i<a.length;i++)

{

//if(a[i]>max)

**if**(a[i]<min)

{

//max=a[i];

min=a[i];

}

}

//System.out.println("Maximum element of array is = "+max);

System.***out***.println("Minimum element of array is = "+min);

}

}

**public** **class** Imp\_DuplicateCharsAndCount {

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

String str = "tasdfasdfafk asd234asda";

Map<Character, Integer> hm = **new** HashMap<Character, Integer>();

**char**[] arr = str.toCharArray();

{

**for** (**char** value: arr) {

**if** (Character.*isAlphabetic*(value)) {

**if** (hm.containsKey(value)) {

hm.put(value, hm.get(value) + 1);

} **else** {

hm.put(value, 1);

}

}

}

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

**for** (**char** x:hm.keySet()) {

**if**(hm.get(x)> 1)

{

System.***out***.println("Duplicate element is : "+x);

}

}

}

}

}

///{a=6, s=4, d=4, f=3, k=1}

/\* public static void main(String[] args) throws IOException {

String ch;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the string");

ch=br.readLine();

int count=0,len=0;

do

{

try

{

char name[]=ch.toCharArray();

len=name.length;

count=0;

for(int j=0;j<len;j++)

{

if((name[0]==name[j])&&((name[0]>=65&&name[0]<=91)||(name[0]>=97&&name[0]<=123)))

count++;

}

if(count!=0)

System.out.println(name[0]+" "+count+" times");

ch=ch.replace(""+name[0],"");

}

catch(Exception ex){}

}

while(len!=1);

}}

\*/

**public** **class** Imp\_DuplicatesInArrayUsingHashMap {

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

**int**[] inputArr = { 2, 5, 6, 7, 7, 11, 8, 2, 7, 9, 5, 7 };

Imp\_DuplicatesInArrayUsingHashMap.*findDuplicatesUsingHashMap*(inputArr);

}

**private** **static** **void** findDuplicatesUsingHashMap(**int** [] arr)

{

Map<Integer, Integer> hm= **new** HashMap<Integer, Integer>();

**for**(**int** x: arr)

{

**if**(!hm.containsKey(x))

{

hm.put(x, 1);

}**else** {

hm.put(x, hm.get(x)+ 1);

}

} //// Using keySet() to get the set view only of all the keys

**for** (Integer x:hm.keySet()) {

**if**(hm.get(x)> 1)

{

System.***out***.println("Duplicate element is : "+x);

}

}

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

}

}

**public** **class** IteratorStrings\_SortingAscendingOrder {

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

ArrayList<String> arrayList1 = **new** ArrayList<String>();

arrayList1.add("Subash");

arrayList1.add("Amrita");

arrayList1.add("Varun");

arrayList1.add("Shreyas");

// Create an iterator object itr ListIterator to traverse the list

Iterator<String> itr = arrayList1.iterator();

// Traversing the list in forward direction

System.***out***.println("Displaying list items in forward direction");

**while** (itr.hasNext()) {

System.***out***.print(itr.next() + ", ");

/\*

\* // ListIterator to traverse the list Iterator<String>

\* itr1=arrayList1.iterator(); // Traversing the list in backward direction

\* System.out.println("Displaying list items in backward direction");

\*

\* while (itr.hasPrevious()) {

\* System.out.println(itr.previous()+", ");

\*/

}

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

//Sorting an ArrayList in increasing order

Collections.*sort*(arrayList1);

System.***out***.println("Sorting List items in Ascending order"+ arrayList1);

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

//Sorting an ArrayList in descending order

Collections.*sort*(arrayList1, Collections.*reverseOrder*());

System.***out***.println("Sorting ArrayList items in descending order "+ arrayList1);

}

}

**public** **class** LinkedHashMaps {

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

// HashMap Declaration

LinkedHashMap<Integer, String> lhmap = **new** LinkedHashMap<Integer, String>();

//Adding elements to LinkedHashMap

lhmap.put(22, "Abey");

lhmap.put(33, "Dawn");

lhmap.put(1, "Sherry");

lhmap.put(1, "Sherry");

lhmap.put(2, "Karon");

lhmap.put(100, "Jim");

lhmap.put(**null**, **null**);

lhmap.put(55, **null**);

lhmap.put(56, **null**);

// Generating a Set of entries

Set set = lhmap.entrySet();

// Displaying elements of LinkedHashMap

Iterator iterator = set.iterator();

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

Map.Entry me = (Map.Entry)iterator.next();

System.***out***.print("Key is: "+ me.getKey() +

"& Value is: "+me.getValue()+"\n");

}

System.***out***.print(lhmap.get(100));

}

}

/\*Key is: 22& Value is: Abey No duplicates, single null key and multiple null values, maintains insertion order

Key is: 33& Value is: Dawn data is stored in Key and value pairs.

Key is: 1& Value is: Sherry

Key is: 2& Value is: Karon

Key is: 100& Value is: Jim

Key is: null& Value is: null

Key is: 55& Value is: null

Key is: 56& Value is: null

Jim\*/===============

**package** Collections;

**import** java.util.ArrayList;

**public** **class** RemoveDuplicatesFromArray {

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

ArrayList<Object> al = **new** ArrayList<Object>();

al.add("java");

al.add('a');

al.add('a');

al.add('b');

al.add('a');

al.add("java");

al.add(10.3);

al.add('c');

al.add(14);

al.add("java");

al.add(12);

System.***out***.println("Before Removing Duplicate elements:"+al);

**for**(**int** i=0;i<al.size();i++){

**for**(**int** j=i+1;j<al.size();j++){

**if**(al.get(i).equals(al.get(j))){

al.remove(j);

j--;

}

}

}

System.***out***.println("After Removing duplicate elements:"+al);

}

}

**public** **class** TreeSetDemo {

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

{

TreeSet<String> ts1 = **new** TreeSet<String>();

// Elements are added using add() method

ts1.add("C");

ts1.add("B");

ts1.add("A");

//ts1.add(null); //Throws null pointer exception if null value is added

//Exception in thread "main" java.lang.NullPointerException

// Duplicates will not get insert

ts1.add("C");

// Elements get stored in default natural

// Sorting Order(Ascending)

System.***out***.println(ts1);

}

}

/\*Output:

[A, B, C]\*/

**public** **class** BrokenLinks {

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

System.*setProperty*("webdriver.chrome.driver",

"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe");

//WebDriver driver = new FirefoxDriver();

WebDriver driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("http://newtours.demoaut.com");

driver.manage().timeouts().implicitlyWait(25, TimeUnit.***SECONDS***);

//capture links from webpage

List<WebElement> links=driver.findElements(By.*tagName*("a"));

//Number of links

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

**for**(**int** i=0; i<=links.size();i++)

{ //by using href attribute we can get the URL of required link

WebElement element= links.get(i);

String url=element.getAttribute("href");

//create a linkurl object

URL linkURL= **new** URL(url);

//create a connection using url object link

HttpURLConnection httpConn= (HttpURLConnection)linkURL.openConnection();

Thread.*sleep*(2000);

httpConn.connect();

**int** resCode= httpConn.getResponseCode();

**if** (resCode>400)

{

System.***out***.println(url+"--"+"is a broken link"+" and response code is = "+resCode);

}**else**

{

System.***out***.println(url+"--"+"is a valid link"+ " and response code is = "+resCode);

}

}

}

}====================

**public** **class** dragAndDrop\_Action {

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

WebDriver driver;

System.*setProperty*("webdriver.chrome.driver", ".//Software//chromedriver.exe");

driver = **new** ChromeDriver();

driver.get("https://jqueryui.com/droppable/");

driver.manage().window().maximize();

Actions action = **new** Actions(driver);

String logoTxt= driver.findElement(By.*xpath*("//h2[@class='logo']/a")).getText();

System.***out***.println("Before the frame, the logo text is displayed as "+logoTxt);

Thread.*sleep*(2000);

WebElement fram=driver.findElement(By.*xpath*("//iframe[@class='demo-frame']"));

driver.switchTo().frame(fram);

//driver.switchTo().frame(0);

WebElement drag = driver.findElement(By.*id*("draggable"));

Thread.*sleep*(4000);

WebElement drop = driver.findElement(By.*id*("droppable"));

action.dragAndDrop(drag, drop).build().perform();

System.***out***.println("I am inside frame");

Thread.*sleep*(2000);

driver.switchTo().defaultContent();

String logoTxt2= driver.findElement(By.*xpath*("//h2[@class='logo']/a")).getText();

System.***out***.println("After closing the frame, the logo text is displayed as "+logoTxt2);

driver.quit();

}

}===================

**public** **class** FramesDemoNew {

@Test

**public** **void** testFrames() {

WebDriver driver= **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///C:/Users/Subhash%20patil/Desktop/myframes.html");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**int** total\_frames=driver.findElements(By.*tagName*("iframe")).size();

System.***out***.println("Total frames are "+total\_frames);

WebElement myFrame=driver.findElement(By.*xpath*("//iframe[@title='selenium\_news']"));

driver.switchTo().frame("w3c");

driver.findElement(By.*xpath*("//a[@title='Search W3Schools']")).click();

driver.findElement(By.*name*("search")).sendKeys("CSS");

driver.switchTo().defaultContent(); //switches to the parent window

driver.findElement(By.*xpath*("html/body/a")).click();

}

}

**public** **class** FramesDemo {

@Test

**public** **void** testFrames() {

WebDriver driver= **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///C:/Users/Subhash%20patil/Desktop/myframes.html");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**int** total\_frames=driver.findElements(By.*tagName*("iframe")).size();

System.***out***.println("Total frames are "+total\_frames);

driver.switchTo().frame("w3c");

driver.findElement(By.*xpath*("//a[@title='Search W3Schools']")).click();

driver.findElement(By.*name*("search")).sendKeys("CSS");

driver.switchTo().defaultContent(); //switches to the parent window

driver.findElement(By.*xpath*("html/body/a")).click();

}

}

**public** **class** CountWordsInString {

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

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

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

String str = s.nextLine();

// calling the method to count the number of words in a string

**int** cnt = *countWordsUsingSplit*(str);

System.***out***.println("Number of words in string are - " + cnt);

}

**public** **static** **int** countWordsUsingSplit(String input) {

**if** (input == **null** || input.isEmpty()) {

**return** 0;

}

String[] words = input.split("\\s+");

**return** words.length;

}

}=============

**package StringFunctions;**

**/\*public class DuplicateCharactersInString {**

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

**String string1 = "Great responsibility";**

**int count;**

**//Converts given string into character array**

**char string[] = string1.toCharArray();**

**System.out.println("Duplicate characters in a given string: ");**

**//Counts each character present in the string**

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

**count = 1;**

**for(int j = i+1; j <string.length; j++) {**

**if(string[i] == string[j] && string[i] != ' ') {**

**count++;**

**//Set string[j] to 0 to avoid printing visited character**

**string[j] = '0';**

**}**

**}**

**//A character is considered as duplicate if count is greater than 1**

**if(count > 1 && string[i] != '0')**

**System.out.println(string[i]);**

**}**

**}**

**}**

**\*/**

**//This Java program is used to find duplicate characters in string.**

**public class DuplicateCharactersInString {**

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

**String str = "Great responsibiliiiity";**

**//String str = "w3schoolls";**

**// char string[] = str.toCharArray();**

**System.out.println("Duplicate Characters are:");**

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

**for (int j = i + 1; j < str.length(); j++) {**

**int cnt = 1;**

**if (str.charAt(i) == str.charAt(j) ) {**

**cnt= cnt+1;**

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

**// break;**

**}**

**}**

**}**

**}**

**}**

**/\*Duplicate Characters are: s o**

**Here in this program, a Java class name DuplStr is declared which is having the main() method. All Java program needs one main() function from where it starts executing program. Inside the main(), the String type variable name str is declared and initialized with string w3schools. Next an integer type variable cnt is declared and initialized with value 0. This cnt will count the number of character-duplication found in the given string.**

**The statement: char [] inp = str.toCharArray(); is used to convert the given string to character array with the name inp using the predefined method toCharArray(). The System.out.println is used to display the message "Duplicate Characters are as given below:". Now the for loop is implemented which will iterate from zero till string length. Another nested for loop has to be implemented which will count from i+1 till length of string.**

**Inside this two nested structure for loops, you have to use an if condition which will check whether inp[i] is equal to inp[j] or not. If the condition becomes true prints inp[j] using System.out.println() with s single incrementation of variable cnt and then break statement will be encountered which will move the execution out of the loop.**

**\*/**

**/\*Output:**

**Duplicate characters in a given string:**

**r**

**e**

**t**

**s**

**i\*/**

**public** **class** DuplicateStrings {

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

String str1 = "w3schoooooolssoo";

**int** cnt = 0;

**char**[] str = str1.toCharArray();

System.***out***.println("Duplicate Characters are:");

**for** (**int** i = 0; i < str1.length(); i++) {

**for** (**int** j = i + 1; j < str1.length(); j++) {

**if** (str[i] == str[j]) {

System.***out***.println(str[j]);

cnt++;

//break;

}

}

}

}

}===========

**public** **class** DuplicateWordsInString {

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

String value = "This is testing Program testing Program";

String words[]=value.split("\\s");

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

**for** (String word: words) {

**if** (hm.containsKey(word)) {

hm.put(word, hm.get(word) + 1);

} **else** {

hm.put(word, 1);

}

}

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

**for** (String word:hm.keySet())

{

**if**(hm.get(word)> 1)

{

System.***out***.println("Duplicate word is : "+word);

}

}

}

}

**public** **class** DuplWordsInString {

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

String str = "testingtti";

// int cnt=str.length();

HashMap<Character, Integer> hm = **new** HashMap<Character, Integer>();

**char** [] charArray= str.toCharArray();

**for** (**char** c:charArray) {

**if** (hm.containsKey(c)) {

hm.put(c, hm.get(c) + 1);

} **else** {

hm.put(c, 1);

}

}

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

**for** (**char** ch:hm.keySet())

{

**if**(hm.get(ch)> 1)

{

System.***out***.println("Duplicate character is : "+ch);

}

}

}

}

/\*{s=1, t=4, e=1, g=1, i=2, n=1}

Duplicate character is : t

Duplicate character is : i\*/

**public** **class** DuplicateCharactersInString {

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

String str = "Great responsibiliiiity";

//String str = "w3schoolls";

// char string[] = str.toCharArray();

System.***out***.println("Duplicate Characters are:");

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

**for** (**int** j = i + 1; j < str.length(); j++) {

**int** cnt = 1;

**if** (str.charAt(i) == str.charAt(j) ) {

cnt= cnt+1;

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

// break;

}

}

}

}

}

/\*Duplicate Characters are: s o

Here in this program, a Java class name DuplStr is declared which is having the main() method. All Java program needs one main() function from where it starts executing program. Inside the main(), the String type variable name str is declared and initialized with string w3schools. Next an integer type variable cnt is declared and initialized with value 0. This cnt will count the number of character-duplication found in the given string.

The statement: char [] inp = str.toCharArray(); is used to convert the given string to character array with the name inp using the predefined method toCharArray(). The System.out.println is used to display the message "Duplicate Characters are as given below:". Now the for loop is implemented which will iterate from zero till string length. Another nested for loop has to be implemented which will count from i+1 till length of string.

Inside this two nested structure for loops, you have to use an if condition which will check whether inp[i] is equal to inp[j] or not. If the condition becomes true prints inp[j] using System.out.println() with s single incrementation of variable cnt and then break statement will be encountered which will move the execution out of the loop.

\*/

/\*Output:

Duplicate characters in a given string:

r

e

t

s

i\*/======================

**public** **class** Fibonacci

{

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

{

**int** n, a = 0, b = 0, c = 1;

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

System.***out***.print("Enter value of n:");

n = s.nextInt();

System.***out***.print("Fibonacci Series:");

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

{

a = b;

b = c;

c = a + b;

System.***out***.print(a+", ");

}

}

}

//Enter value of n:12

//Fibonacci Series:0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,

/\*Enter value of n:12

Fibonacci Series:0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \*/

/\*Java Program to Generate Fibonacci Numbers

This is a Java Program to Generate Fibonacci Numbers.

The number is said to be in a Fibonacci series if each subsequent

number is the sum of the previous two numbers.\*/

**public** **class** HashCodeInJava {

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

String str1 = "";

String str2 = **new** String();

System.***out***.println(str1.hashCode()); //0

System.***out***.println(str2.hashCode()); //0

String str3 = "subash";

String str4 = **new** String("subash");

System.***out***.println(str3.hashCode()); //-891546570

System.***out***.println(str4.hashCode()); //-891546570

}

}

**public** **class** pallindrome {

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

String a, b = "";

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

System.***out***.print("Enter the string you want to check:"); //madam, malayalam, or NeveroddorEVen

a = s.nextLine();

**int** n = a.length();

**for** (**int** i = n - 1; i >= 0; i--) {

b = b + a.charAt(i);

}

**if** (a.equalsIgnoreCase(b)) {

System.***out***.println("The string is palindrome.");

} **else** {

System.***out***.println("The string is not palindrome.");

}

}

}

/\*

VBScript code to check if a string is Palindrome or not

'Program to check if the given string is a Palindrome or not

MyStr=Ucase(inputbox("Enter the String:"))

RevStr=strreverse(MyStr)

if strcomp(MyStr,RevStr)=0 then

msgbox "It is a Palindrome"

else

msgbox "It is not a Palindrome"

end if

\*/

**public** **class** RegExCaseSensitive {

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

{

//public int cnt;

// the string we want to search in

String str = "happYyy"; // happy new year";

// search for this simple pattern

String sub\_str ="Y"; // " ye.\* ";

// set everything up

//Pattern p = Pattern.compile(sub\_str); //Case sensitive

Pattern p = Pattern.*compile*(sub\_str, Pattern.***CASE\_INSENSITIVE***);

Matcher m = p.matcher(str);

**int** res=0;

// now see if we find a match

**while** (m.find()) {

res++;

}

System.***out***.println("Found a match, "+ res+" times ");

}

}

==============

**public** **class** RegExMethod {

// Method that returns the count of the given

// character in the string

//public static int RegExcount(String s, String ch) {

**public** **static** **int** RegExcount(String s, **char** ch) {

// Use Matcher class of java.util.regex

// to match the character

Matcher matcher=Pattern.*compile*(String.*valueOf*(ch)).matcher(s);

//System.out.println((String.valueOf(ch)));

**int** res=0;

**while**(matcher.find()) {

res++;

}

**return** res;

}

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

/\*String str = "geeksforgeeks";

char c = 'e';\*/

String str = "subash is a bad bad bad boy";

**char** c = 's';

//String c = "bad";

System.***out***.println(*RegExcount*(str, c));

}

}

**public** **class** ReverseLine {

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

String strLine = "Java Reverse string by word example";

//specify delimiter as " " space

StringTokenizer st = **new** StringTokenizer(strLine, " ");

String strReversedLine = "";

**while**(st.hasMoreTokens())

{

strReversedLine = st.nextToken() + " " + strReversedLine;

}

System.***out***.println("Reversed string by word is : " + strReversedLine);

}

}

/\*

Output would be

Reversed string by word is : example word by string Reverse Java

\*/

**public** **class** ReverseString {

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

System.***out***.println("Enter string to reverse:");

@SuppressWarnings("resource")

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

String str = read.nextLine();

String reverse = "";

**for** (**int** i = str.length() - 1; i >= 0; i--) {

reverse = reverse + str.charAt(i);

}

System.***out***.println("Reversed string is : " + reverse);

}

}

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

{

System.***out***.println("Enter string to reverse:");

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

String str = read.nextLine();

StringBuilder sb = **new** StringBuilder(str);

// StringBuffer sb = new StringBuffer(str);

System.***out***.println("Reversed string is:");

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

}

}

**public** **class** StringImmutable {

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

String s= **new** String("Hello");

//s.concat("Java");

s=s+"Java";

System.***out***.println(s);// prints Hello

}

}

**public** **class** reverseStringUsingStringBuilder

{

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

{

System.out.println("Enter string to reverse:");

Scanner read = new Scanner(System.in);

String str = read.nextLine();

StringBuilder sb = new StringBuilder();

for(int i = str.length() - 1; i >= 0; i--)

{

sb.append(str.charAt(i));

}

System.out.println("Reversed string is:");

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

}

}\*/

//==============================================

//{

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

{

System.***out***.println("Enter string to reverse:");

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

String str = read.nextLine();

StringBuilder sb = **new** StringBuilder(str);

// StringBuffer sb = new StringBuffer(str);

System.***out***.println("Reversed string is:");

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

}

}

**public** **class** StringManipulation {

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

// index starts from 0 and string is a collection of character written in double

// quotes.

String str = "The rains have started here, subash";

String str1 = "The rains have started here, subash";

String str3 = "The rains Have started here, subash";

String str2 = "The rains have started here, subash Patil";

// code to get the length of string

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

// to find the character at 5th location

System.***out***.println(str.charAt(5));

// to find th index of a specific character

System.***out***.println("This is "+ str.indexOf('s',str.indexOf('s')));

// now i want to get the index of 2nd s in teh string starting from index

// position of first s

System.***out***.println(str.indexOf('s', 8));

// System.out.println(str.indexOf('s', 15));

System.***out***.println(str.indexOf("have")); // occurrence of have in the string

System.***out***.println(str.indexOf("hello")); // since hello is not present, it returns -1

// string comparison returns true if equal else false

System.***out***.println(str.equals(str1));

// in str3 the letter H in Have is capital letter so we need to use ignorecase

// to return true

System.***out***.println(str.equalsIgnoreCase(str3));

// substring, u have to mention the start index and end index of "The rains"

System.***out***.println(str.substring(0, 9));

// trim

String s = " Hello World ";

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

System.***out***.println(s.replace(" ", ""));

String dat = "15-11-74"; // replace this to 15/11/74

System.***out***.println(dat.replace("-", "/"));

// split, return type of split is an array

String test = "Hello\_World\_Test\_Selenium";

String testval[] = test.split("\_");

System.***out***.println("total number of underscores are"+testval.length);

**for** (**int** i = 0; i < (testval.length-1); i++) {

System.***out***.println(testval[i]);

}

// concat output of below is caress

String s2 = "cares";

System.***out***.println(s2.concat("s"));

// VERY VERY IMPORTANT QUESTION

String x = "Hello";

String y = "World";

**int** a = 100;

**int** b = 200;

System.***out***.println(x + y); // HelloWorld

System.***out***.println(a + b); // 300

System.***out***.println(x + y + a + b); // Compiler executes from the left so O/P is HelloWorld100200

System.***out***.println(a + b + x + y); // Compiler executes from the left so O/P is 300HelloWorld

System.***out***.println(x + y + (a + b)); // x and y concatenates with resultant of a+b =O/P is HelloWorld300

// calling a method to return the number of Ls in String s="GOOGLLLLLEE";

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

}

/\*

\* public static int mylengthOfDuplicateChars() { String s = "GOOGLLLLLLLEE";

\* int first = s.indexOf('L'); int last = s.lastIndexOf('L'); int counter = 0;

\* for (int i = first; i <= last; i++) { // if(s.charAt(i)=='L')

\*

\* //(browserName.equals("chrome")) if (s.charAt(i) == ('L' | 'l')) counter++;

\* System.out.println("occurence of L at position " + i); } return counter;

\*

\* }

\*/

**public** **static** **int** mylengthOfDuplicateChars() {

String s = "GOOGLLLLLllEE";

/\*

\* int first = s.indexOf('L'); int last = s.lastIndexOf('L');

\*/

**int** counter = 0;

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

**if** ((s.charAt(i) == 'L') | (s.charAt(i) == 'l'))

{ counter=counter++;

System.***out***.println("occurence of L at position " + i);

}

}

**return** counter;

}

}

=============

**public** **class** StringsManipulation {

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

//index starts from 0 and string is a collection of character written in double quotes.

String str="The rains have started here, subash";

//code to get the length of string

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

//to find the character at 5th location

System.***out***.println(str.charAt(5));

//to find th index of a specific character

System.***out***.println(str.indexOf('s'));

//now i want to get the index of 2nd s in teh string starting from index position of first s

System.***out***.println("Third s's index "+str.indexOf('s', str.indexOf('s')+1));

System.***out***.println("Third s's index "+str.indexOf('s', str.indexOf('s', str.indexOf('s')+1+1)));

//finding the 2nd occurence of s in teh string

System.***out***.println(str.indexOf('s',str.indexOf('s')+1));

//finding the 3rd occurence of s in teh string

System.***out***.println(str.indexOf('s',str.indexOf('s',str.indexOf('s')+1+1+1)));

/\*public static int ordinalIndexOf(String str, String substr, int n) {

int pos = -1;

do {

pos = str.indexOf(substr, pos + 1);

} while (n-- > 0 && pos != -1);

return pos;

}\*/

/\*ordinalIndexOf(s, ss, 0) is equivalent to s.indexOf(ss)

ordinalIndexOf(s, ss, 1) is equivalent to s.indexOf(ss, s.indexOf(ss) + 1)

and so on...

Example input / output

Expression Result Explanation

ordinalIndexOf("abcd abcd", "bc", 0) 1 The 0th occurrence is found at index 1

ordinalIndexOf("abcd abcd", "bc", 1) 6 The 1st (zero-based) occurrence is found at index 6

ordinalIndexOf("abcd abcd", "bc", 2) -1 There's no 2nd (zero-based) occurrence, hence -1 is returned

ordinalIndexOf("aaaaa", "aaa", 1) 1 Matches may overlap\*/

}

}

**public** **class** trial {

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

String str="shisubais is issh";

//String s1=new String("shubash");

**int** first = str.indexOf("is");

**int** second = str.indexOf("is", first + 1);

**int** third = str.indexOf("is", second + 1);

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

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

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

/\*int s=str.indexOf("is", str.indexOf("is") + 1);

System.out.println(s);

System.out.println("Third s's index "+str.indexOf("is", str.indexOf("is", str.indexOf("is")+1+1)));

\*/

}

}

**public** **class** Strings {

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

StringBuilder sb = **new** StringBuilder("ABCDEFGHIGHLMNOPQRSTUVWXYZ");

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

System.***out***.println("substring: " + sb.substring(0,10));

sb = sb.insert(5, "abcd");

System.***out***.println("insert: " + sb);

sb = sb.replace(8,18,"\_middle\_");

System.***out***.println("replace: " + sb);

sb = sb.append("123456789");

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

System.***out***.println("indexOf: " + sb.indexOf("QRS"));

sb = sb.reverse();

System.***out***.println("reverse: " + sb);

System.***out***.println("IndexOf: " + sb.indexOf("\_"));

System.***out***.println("lastIndexOf: " + sb.lastIndexOf("\_"));

}

}

**public** **class** GoogleNext {

**static** {

System.setProperty("webdriver.chrome.driver", ".\\Software\\chromedriver.exe");

}

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

WebDriver driver = **new** ChromeDriver(); // upcasting

driver.manage().window().maximize();

driver.get("https://www.google.com/");

Thread.sleep(3000);

//Enter Java in the search box and hit enter

WebElement SrchBox = driver.findElement(By.name("q"));

SrchBox.sendKeys("Java", Keys.ENTER);

Thread.sleep(2000);

**int** i=0;

//locating the page link numbers

List<WebElement> page = driver.findElements(By.xpath("//tr[@valign='top']/td/a"));

Iterator<WebElement> itr= page.iterator();

**while**(itr.hasNext())

driver.findElement(By.xpath("//a/span[contains(.,'Next')]")).click();

System.out.print("The title is = "+ i + driver.getTitle());

i++;

System.out.println(i);

}

}

**public** **class** TestBase {

**public** **static** WebDriver *driver*;

**public** **static** Properties *prop*;

**public** TestBase() {

**try** {

*prop* = **new** Properties();

FileInputStream ip= **new** FileInputStream("E:\\MyWorkspace\_Completed\\FreeCRMTest\\src\\main\\java\\com\\crm\\qa\\config\\config.properties");

*prop*.load(ip);

}**catch** (FileNotFoundException e) {

e.printStackTrace();

}**catch** (IOException e) {

e.printStackTrace();

}

}

**public** **static** **void** initialization() {

String browserName= *prop*.getProperty("browser");

**if** (browserName.equals("chrome")) {

System.*setProperty*("webdriver.chrome.driver","E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe" );

//System.setProperty("webdriver.chrome.driver","E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\Old\_chromedriver.exe" ); // Old\_chromedriver.exe

//System.setProperty("webdriver.chrome.driver",".//FreeCRMTest//Software//Old\_chromedriver.exe" );

*driver*= **new** ChromeDriver();

}**else** **if** (browserName.equals("FF")) {

System.*setProperty*("webdriver.gecko.driver","E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\geckodriver.exe" );

*driver*= **new** FirefoxDriver();

}

*driver*.manage().window().maximize();

*driver*.manage().deleteAllCookies();

*driver*.manage().timeouts().pageLoadTimeout(TestUtil.*PAGE\_LOAD\_TIMEOUT*, TimeUnit.***SECONDS***);

//driver.manage().timeouts().pageLoadTimeout(60, TimeUnit.SECONDS);

//driver.manage().timeouts().implicitlyWait((prop.getProperty("IMPLICIT\_WAIT")), TimeUnit.SECONDS);

*driver*.manage().timeouts().implicitlyWait(TestUtil.*IMPLICIT\_WAIT*, TimeUnit.***SECONDS***);

//if u want to navigate to the origin of this code TestUtil.IMPLICIT\_WAIT use ctrl+mouseclick

*driver*.get(*prop*.getProperty("url"));

}

}

ConvertToString

**public** **class** ConvertString {

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

String string1 = "1000";

String string2 = "2000";

// concating strings....

System.***out***.println(string1 + string2);

// convert the strings to ints.

**int** i1 = Integer.*parseInt*(string1);

**int** i2 = Integer.*parseInt*(string2);

System.***out***.println(i1 + i2);

*integerToString*();

*integerToString2*();

*integerToChar*();

}

**private** **static** **void** integerToString() {

**int** num = 65;

System.***out***.println("Conversion of Integer to String = " + Integer.*toString*(num));

}

**public** **static** **void** integerToString2() {

**int** number = 1234;

String str = String.*valueOf*(number);

System.***out***.println("Conversion of Integer to String = " + str);

}

**public** **static** **void** integerToChar() {

**int** a = 65;

**char** ch = (**char**) a;

System.***out***.println("The character value of an integer is: " + ch);

}

}

/\*

\* 10002000 3000

\*/

Add elements to ArrayList

**public** **class** findElementsConcept {

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

WebDriver driver;

ArrayList<String> arrayList1 = **new** ArrayList<String>();

// System.setProperty("webdriver.chrome.driver","E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\Old\_chromedriver.exe"

// ); // Old\_chromedriver.exe

System.*setProperty*("webdriver.chrome.driver",

"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe"); // Old\_chromedriver.exe

// System.setProperty("webdriver.chrome.driver",".//FreeCRMTest//Software//Old\_chromedriver.exe"

// );

driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.manage().deleteAllCookies();

driver.get("https://www.ebay.com");

// driver.manage().timeouts().pageLoadTimeout(TestUtil.PAGE\_LOAD\_TIMEOUT,

// TimeUnit.SECONDS);

driver.manage().timeouts().pageLoadTimeout(60, TimeUnit.***SECONDS***);

// driver.manage().timeouts().implicitlyWait((prop.getProperty("IMPLICIT\_WAIT")),

// TimeUnit.SECONDS);

driver.manage().timeouts().implicitlyWait(60, TimeUnit.***SECONDS***);

// if u want to navigate to the origin of this code TestUtil.IMPLICIT\_WAIT use

// ctrl+mouseclick

// count the number of links on the page

// get the text of each link on the page

List<WebElement> LnkList = driver.findElements(By.*tagName*("a"));

**int** totalLinks = LnkList.size();

// size of link list

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

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

String linkText = LnkList.get(i).getText();

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

arrayList1.add(linkText);

}

/\*

\* Iterator<String> itr= arrayList1.iterator();

\* while(itr.hasNext())

\* System.out.println();

\*/

Collections.*sort*(arrayList1);

System.***out***.println("Sorting Elements in the page in Ascending Order " + arrayList1);

/\*

\* Collections.sort(arrayList1, Collections.reverseOrder());

\* System.out.println("Sorting Elements in the page in descending Order "+

\* arrayList1);

\*/

}

}

getSystemDate

**public** **class** getSystemDate {

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

// Create object of SimpleDateFormat class and decide the format

DateFormat dateFormat = **new** SimpleDateFormat("MM/dd/yyyy HH:mm:ss");

// get current date time with Date()

Date date = **new** Date();

// Now format the date

String myDate = dateFormat.format(date);

// Print the Date

System.***out***.println("Current date and time is " + myDate);

}

}

// Current date and time is 06/10/2020 13:32:11

**public** **class** HighlighterClass {

@Test

**public** **void** highlighterElement() {

//System.setProperty("webdriver.gecko.driver", "D:\\Selenium Environment\\Drivers\\geckodriver.exe");

System.*setProperty*("webdriver.chrome.driver",

"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe");

//WebDriver driver = new FirefoxDriver();

WebDriver driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("https://www.gmail.com");

WebElement ele = driver.findElement(By.*xpath*("//\*[@id='identifierId']"));

//Call the highlighterMethod and pass webdriver and WebElement which you want to highlight as arguments.

highlightObject(driver,ele);

//ele.sendKeys("SoftwareTestingMaterial.com");

//ALTERNATIVE METHOD TO ENTER TEXT OTHER THAN SENDKEYS

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].value='SoftwareTestingMaterial.com';", ele);

}

//Creating a custom function

**public** **void** highlightObject(WebDriver driver, WebElement element){

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("arguments[0].setAttribute('style', 'background: yellow; border: 2px solid red;');", element);

}

}

**public** **class** methodChaining {

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

StringBuilder sb = **new** StringBuilder("T");

// Below line uses method chaining technique

sb.insert(0,"Shubash ").append(" Patil, ").append(" QA");

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

// Below line also uses method chaining technique

String str = sb.substring(18).replace("QA", "Automation").toUpperCase();

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

}

}

/\*Shubash T Patil, QA

AUTOMATION

\*/

/\*Method Chaining is the practice of calling different methods in a single line instead of calling different

methods with the same object reference separately. Under this procedure, we have to write the object reference

once and then call the methods by separating them with a (dot.). \*/

**class** Person {

String name;

**int** age;

// This method can be used in chaining

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

**this**.name = name;

**return** **this**;

}

// This method can also be used in chaining

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

**this**.age = age;

**return** **this**;

}

**public** **void** getPersonDetails() {

System.***out***.println("Person name : " + name + ", age : " +age+" year.");

}

}

**class** MethodChainingDemo {

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

Person p= **new** Person();

// Below line uses method chaining technique

p.setName("Shubash").setAge(46).getPersonDetails();

}

}

**public** **class** MultiPopupClose {

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

System.*setProperty*("webdriver.gecko.driver", ".//Software//geckodriver.exe");

//System.setProperty("webdriver.chrome.driver",".//Software//chromedriver.exe");

//WebDriver driver = new ChromeDriver();

WebDriver driver = **new** FirefoxDriver();

driver.get("http://www.popuptest.com/popuptest2.html");

driver.manage().window().maximize();

Thread.*sleep*(5000);

String ParentWindow= driver.getWindowHandle();

System.***out***.println("Parent Window = "+ParentWindow);

Thread.*sleep*(6000);

// String firstWindow = null, SecondWindow = null, ThirdWindow = null, FourthWindow=null ;

Set<String> allWindowHandles = driver.getWindowHandles();

allWindowHandles.remove(ParentWindow);

**int** i=1;

**for**(String handle : allWindowHandles) {

System.***out***.println("Window handle - >#" + i++ +" " + handle);

driver.switchTo().window(handle);

Thread.*sleep*(8000);

System.***out***.println("Switched to Window"+ handle);

//close the parent window

driver.close();

System.***out***.println("closed the Window"+ handle);

Thread.*sleep*(8000);

}

//and another popup launches when you close the main window

Set<String> allWindowHandles2 = driver.getWindowHandles();

**int** j=1;

**for**(String handle : allWindowHandles2) {

System.***out***.println("Window handle - >#" + j++ +" " + handle);

driver.switchTo().window(handle);

Thread.*sleep*(8000);

System.***out***.println("Switched to last child Window"+ handle);

//close the parent window

driver.close();

System.***out***.println("closed the last child Window"+ handle);

Thread.*sleep*(8000);

}

}

}

**public** **class** ReadFromTextFile {

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

// FileReader is meant for reading streams of characters.

// For reading streams of raw bytes, consider using a FileInputStream

FileReader fileReader = **new** FileReader("E:\\MyWorkspace\_Completed\\ReadFromTxtFile.txt");

/\*

\* BufferedReader reads text from a character-input stream, buffering characters

\* so as to provide for the efficient reading of characters, arrays, and lines.

\*/

BufferedReader bufferedReader = **new** BufferedReader(fileReader);

String txtFromFile = **null**;

// use the while loop to read the file until there are no more lines

**while** ((txtFromFile = bufferedReader.readLine()) != **null**) {

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

}

fileReader.close();

// Close the buffer reader object to prevent memory leak

bufferedReader.close();

}

}

COLLECTIONS

**public** **class** ArrayListConcept {

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

//To an arrayList u can add any amount of number or any type of values

//stores values on basis of index

//1. create an object for ArrayList

ArrayList ar= **new** ArrayList();

//2. Add items to the ArrayList

ar.add(100); //0

ar.add(200); //1

ar.add("Subash"); //2

System.***out***.println(ar.size()); //3

ar.add(400); //3

ar.add(500); //4

System.***out***.println(ar.size()); //5

ar.add("Tom"); //5

ar.add("Hello"); //6

ar.add(12.33); //7

ar.add('M'); //8

ar.add(600);

System.***out***.println(ar.size()); //10

System.***out***.println(ar.get(2)); //Subash

ar.remove(9);

System.***out***.println(ar.size()); //9

System.***out***.println(ar.get(4)); //500

System.***out***.println(ar.get(8)); //M

//to print all the values of an arrayList use the for loop

**for**( **int** i=0; i<=ar.size(); i++)

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

//To add only integer values to your arrayList, you need to add <Integer> on both left and right as below

ArrayList<Integer> ar1 = **new** ArrayList<Integer>();

ar1.add(100);

//ar1.add("Tom"); //This will throw an error if u try to add string value

//System.out.println(ar1.get(0));

//To add only String values to your arrayList, you need to add <String> on both left and right as below

ArrayList<String> ar2= **new** ArrayList<String>();

ar2.add("Subash ");

//System.out.println(ar2.get(0));

}

}

@@@@@@@@@@@@@@@@@@@

**public** **class** ArrayListExample {

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

/\* Creating ArrayList of type "String" which means

\* we can only add "String" elements

\*/

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

/\*This is how we add elements to an ArrayList\*/

obj.add("Ajeet");

obj.add("Harry");

obj.add("Chaitanya");

obj.add("Steve");

obj.add("Anuj");

obj.add("Ajeet");

// Displaying elements

System.***out***.println("Original ArrayList:");

**for**(String str:obj)

{

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

}

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

/\* \* Add element at the given index

\* obj.add(0, "Rahul") - Adding element "Rahul" at first position

\* obj.add(1, "Justin") - Adding element "Justin" at second position\*/

obj.add(0, "Rahul");

obj.add(1, "Justin");

// Displaying elements

System.***out***.println("ArrayList after add operation:");

**for**(String str:obj)

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

//Remove elements from ArrayList like this

obj.remove("Chaitanya"); //Removes "Chaitanya" from ArrayList

obj.remove("Harry"); //Removes "Harry" from ArrayList

// Displaying elements

System.***out***.println("ArrayList after remove operation:");

**for**(String str:obj)

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

//Remove element from the specified index

obj.remove(1); //Removes Second element from the List

// Displaying elements

System.***out***.println("Final ArrayList:");

**for**(String str:obj)

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

}

}

/\*

Original ArrayList:

Ajeet

Harry

Chaitanya

Steve

Anuj

ArrayList after add operation:

Rahul

Justin

Ajeet

Harry

Chaitanya

Steve

Anuj

ArrayList after remove operation:

Rahul

Justin

Ajeet

Steve

Anuj

Final ArrayList:

Rahul

Ajeet

Steve

Anuj\*/

@@@@@@@@@@@@@@@@@@@

w@@@@@@@@@@@@@@@

**public** **abstract** **class** HashMap\_Demo {

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

{

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

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

map.put("vishal", 10);

map.put("sachin", 30);

map.put("vaibhav", 20);

map.put("vaibhav", 20); //No duplicates allowed

map.put(**null**, **null**); //Single null key allowed

map.put("one", **null**); //Multiple null values allowed

System.***out***.println("Size of map is: "

+ map.size());

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

**if** (map.containsKey("vishal")) {

Integer a = map.get("vishal");

System.***out***.println("value for key"

+ " \"vishal\" is:- "

+ a);

}

map.clear();

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

}

**public** **static** **void** print(Map<String, Integer> map)

{

**if** (map.isEmpty()) {

System.***out***.println("map is empty");

}

**else** {

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

}

}

}

@@@@@@@@@@@@@@@@@@

**public** **class** HashSet\_Demo {

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

{

HashSet<String> h = **new** HashSet<String>();

// Adding elements into HashSet usind add()

h.add("India");

h.add("Australia");

h.add("South Africa");

h.add("India");// adding duplicate elements

h.add(**null**);

h.add(**null**); //only one null can be inserted or added

// Displaying the HashSet

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

System.***out***.println("List contains India or not:" +

h.contains("India"));

// Removing items from HashSet using remove()

h.remove("Australia");

System.***out***.println("List after removing Australia:"+h);

// Iterating over hash set items

System.***out***.println("Iterating over list:");

Iterator<String> itr = h.iterator();

**while** (itr.hasNext())

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

}

}

/\*[null, South Africa, Australia, India] No duplicates, no insertion order, single null value

List contains India or not:true

List after removing Australia:[null, South Africa, India]

Iterating over list:

null

South Africa

India\*/@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@

**public** **class** HashTableConcept {

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

//It is also a collection object but values are stored in key and value pairs. It is a dynamic array

//here instead of add we have put method

Hashtable h= **new** Hashtable();

h.put("A", "Test");

h.put("B", "Hello");

h.put("C", "World");

// to get the size

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

h.put(1, 100);

h.put(2, 200);

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

System.***out***.println(h.get(2));

System.***out***.println(h.get("C"));

h.put(3,"Tom"); //you can also use key as int and value as string

System.***out***.println(h.get(3));

//now if we want to restrict key and value to Integer type, we use below

Hashtable <Integer, Integer> h1= **new** Hashtable<Integer, Integer>();

h1.put(1, 100);

System.***out***.println(h1.get(1));

//now if we want to restrict key and value to String type, we use below

Hashtable <Integer, String> h2= **new** Hashtable<Integer, String>();

h2.put(1, "Tom");

System.***out***.println(h2.get(1));

}

}

@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@

**public** **class** HashtableExample {

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

Enumeration names;

String key;

// Creating a Hashtable

Hashtable<String, String> hashtable =

**new** Hashtable<String, String>();

// Adding Key and Value pairs to Hashtable

hashtable.put("Father","Subash");

hashtable.put("Mother","Amrita");

hashtable.put("Son1","Varun");

hashtable.put("Son2","Shreyas");

hashtable.put("LName","Patil");

names = hashtable.keys();

**while**(names.hasMoreElements()) {

key = (String) names.nextElement();

System.***out***.println("Key: " +key+ " & Value: " +

hashtable.get(key));

}

}

}

/\*Key: Father & Value: Subash

Key: Mother & Value: Amrita

Key: Son2 & Value: Shreyas

Key: LName & Value: Patil

Key: Son1 & Value: Varun\*/

@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@

**public** **class** Imp\_DuplicateCharsAndCount {

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

String str = "tasdfasdfafk asd234asda";

Map<Character, Integer> hm = **new** HashMap<Character, Integer>();

**char**[] arr = str.toCharArray();

{

**for** (**char** value: arr) {

**if** (Character.*isAlphabetic*(value)) {

**if** (hm.containsKey(value)) {

hm.put(value, hm.get(value) + 1);

} **else** {

hm.put(value, 1);

}

}

}

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

**for** (**char** x:hm.keySet()) {

**if**(hm.get(x)> 1)

{

System.***out***.println("Duplicate element is : "+x);

}

}

}

}

}

///{a=6, s=4, d=4, f=3, k=1}

@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@

**public** **class** Imp\_DuplicatesInArrayUsingHashMap {

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

**int**[] inputArr = { 2, 5, 6, 7, 7, 11, 8, 2, 7, 9, 5, 7 };

Imp\_DuplicatesInArrayUsingHashMap.*findDuplicatesUsingHashMap*(inputArr);

}

**private** **static** **void** findDuplicatesUsingHashMap(**int** [] arr)

{

Map<Integer, Integer> hm= **new** HashMap<Integer, Integer>();

**for**(**int** x: arr)

{

**if**(!hm.containsKey(x))

{

hm.put(x, 1);

}**else** {

hm.put(x, hm.get(x)+ 1);

}

} //// Using keySet() to get the set view only of all the keys

**for** (Integer x:hm.keySet()) {

**if**(hm.get(x)> 1)

{

System.***out***.println("Duplicate element is : "+x);

}

}

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

}

}

@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@

**public** **class** Iterators\_SortingDescendingOrder {

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

//1. create an object for ArrayList

ArrayList<Integer> arrayList1 = **new** ArrayList<Integer>();

//2. Add items to ArrayList

arrayList1.add(15);

arrayList1.add(20);

arrayList1.add(10);

//iterator object to iterate through the collection

Iterator<Integer> itr = arrayList1.iterator();

**while** (itr.hasNext()) {

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

}

Collections.*sort*(arrayList1);

System.***out***.println("Arranged in increasing order"+ arrayList1);

Collections.*sort*(arrayList1, Collections.*reverseOrder*());

System.***out***.println("Arranged in decreasing order"+ arrayList1);

}

}

@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** IteratorStrings\_SortingAscendingOrder {

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

ArrayList<String> arrayList1 = **new** ArrayList<String>();

arrayList1.add("Subash");

arrayList1.add("Amrita");

arrayList1.add("Varun");

arrayList1.add("Shreyas");

// Create an iterator object itr ListIterator to traverse the list

Iterator<String> itr = arrayList1.iterator();

// Traversing the list in forward direction

System.***out***.println("Displaying list items in forward direction");

**while** (itr.hasNext()) {

System.***out***.print(itr.next() + ", ");

/\*

\* // ListIterator to traverse the list Iterator<String>

\* itr1=arrayList1.iterator(); // Traversing the list in backward direction

\* System.out.println("Displaying list items in backward direction");

\*

\* while (itr.hasPrevious()) {

\* System.out.println(itr.previous()+", ");

\*/

}

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

//Sorting an ArrayList in increasing order

Collections.*sort*(arrayList1);

System.***out***.println("Sorting List items in Ascending order"+ arrayList1);

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

//Sorting an ArrayList in descending order

Collections.*sort*(arrayList1, Collections.*reverseOrder*());

System.***out***.println("Sorting ArrayList items in descending order "+ arrayList1);

}

}

@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@

**public** **class** JavaExample {

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

/\* Creating ArrayList of type "String" which means

\* we can only add "String" elements

\*/

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

/\*This is how we add elements to an ArrayList\*/

obj.add("Ajeet");

obj.add("Harry");

obj.add("Chaitanya");

obj.add("Steve");

obj.add("Anuj");

// Displaying elements

System.***out***.println("Original ArrayList:");

**for**(String str:obj)

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

/\* Add element at the given index

\* obj.add(0, "Rahul") - Adding element "Rahul" at first position

\* obj.add(1, "Justin") - Adding element "Justin" at second position

\*/

obj.add(0, "Rahul");

obj.add(1, "Justin");

// Displaying elements

System.***out***.println("ArrayList after add operation:");

**for**(String str:obj)

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

//Remove elements from ArrayList like this

obj.remove("Chaitanya"); //Removes "Chaitanya" from ArrayList

obj.remove("Harry"); //Removes "Harry" from ArrayList

// Displaying elements

System.***out***.println("ArrayList after remove operation:");

**for**(String str:obj)

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

//Remove element from the specified index

obj.remove(1); //Removes Second element from the List

// Displaying elements

System.***out***.println("Final ArrayList:");

**for**(String str:obj)

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

}

}

@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@

**public** **class** LinkedHashMaps {

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

// HashMap Declaration

LinkedHashMap<Integer, String> lhmap = **new** LinkedHashMap<Integer, String>();

//Adding elements to LinkedHashMap

lhmap.put(22, "Abey");

lhmap.put(33, "Dawn");

lhmap.put(1, "Sherry");

lhmap.put(1, "Sherry");

lhmap.put(2, "Karon");

lhmap.put(100, "Jim");

lhmap.put(**null**, **null**);

lhmap.put(55, **null**);

lhmap.put(56, **null**);

// Generating a Set of entries

Set set = lhmap.entrySet();

// Displaying elements of LinkedHashMap

Iterator iterator = set.iterator();

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

Map.Entry me = (Map.Entry)iterator.next();

System.***out***.print("Key is: "+ me.getKey() +

"& Value is: "+me.getValue()+"\n");

}

System.***out***.print(lhmap.get(100));

}

}

/\*Key is: 22& Value is: Abey No duplicates, single null key and multiple null values, maintains insertion order

Key is: 33& Value is: Dawn data is stored in Key and value pairs.

Key is: 1& Value is: Sherry

Key is: 2& Value is: Karon

Key is: 100& Value is: Jim

Key is: null& Value is: null

Key is: 55& Value is: null

Key is: 56& Value is: null

Jim\*/@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@

**public** **class** LinkedHashSetExample {

/\*LinkedHashSet is also an implementation of Set interface,

\* it is similar to the HashSet and TreeSet except the below mentioned differences:

HashSet doesn’t maintain any kind of order of its elements.

TreeSet sorts the elements in ascending order.

LinkedHashSet maintains the insertion order. Elements gets sorted in the same sequence in which they

have been added to the Set.

Example of LinkedHashSet:

\*/

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

// LinkedHashSet of String Type

LinkedHashSet<String> lhset = **new** LinkedHashSet<String>();

// Adding elements to the LinkedHashSet

lhset.add("Z");

lhset.add("PQ");

lhset.add("N");

lhset.add("O");

lhset.add("KK");

lhset.add("KK");

lhset.add("KK");

lhset.add("FGH");

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

Iterator<String> itr=lhset.iterator();

System.***out***.println(lhset.contains("PQ"));

lhset.remove("KK");

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

**while** (itr.hasNext())

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

}

/\* // LinkedHashSet of Integer Type

LinkedHashSet<Integer> lhset2 = new LinkedHashSet<Integer>();

// Adding elements

lhset2.add(99);

lhset2.add(7);

lhset2.add(7);

lhset2.add(7);

lhset2.add(0);

lhset2.add(67);

lhset2.add(89);

lhset2.add(66);

System.out.println(lhset2);}\*/

}/\*[Z, PQ, N, O, KK, FGH] Insertion order is maintained and also no duplicates are added

[99, 7, 0, 67, 89, 66]\*/

@@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** RemoveDuplicatesFromArray {

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

ArrayList<Object> al = **new** ArrayList<Object>();

al.add("java");

al.add('a');

al.add('a');

al.add('b');

al.add('a');

al.add("java");

al.add(10.3);

al.add('c');

al.add(14);

al.add("java");

al.add(12);

System.***out***.println("Before Removing Duplicate elements:"+al);

**for**(**int** i=0;i<al.size();i++){

**for**(**int** j=i+1;j<al.size();j++){

**if**(al.get(i).equals(al.get(j))){

al.remove(j);

j--;

}

}

}

System.***out***.println("After Removing duplicate elements:"+al);

}

}

@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** RemoveDuplicatesFromArray {

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

ArrayList<Object> al = **new** ArrayList<Object>();

al.add("java");

al.add('a');

al.add('a');

al.add('b');

al.add('a');

al.add("java");

al.add(10.3);

al.add('c');

al.add(14);

al.add("java");

al.add(12);

System.***out***.println("Before Removing Duplicate elements:"+al);

**for**(**int** i=0;i<al.size();i++){

**for**(**int** j=i+1;j<al.size();j++){

**if**(al.get(i).equals(al.get(j))){

al.remove(j);

j--;

}

}

}

System.***out***.println("After Removing duplicate elements:"+al);

}

}

**public** **class** FramesDemo {

@Test

**public** **void** testFrames() {

WebDriver driver= **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///C:/Users/Subhash%20patil/Desktop/myframes.html");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**int** total\_frames=driver.findElements(By.*tagName*("iframe")).size();

System.***out***.println("Total frames are "+total\_frames);

driver.switchTo().frame("w3c");

driver.findElement(By.*xpath*("//a[@title='Search W3Schools']")).click();

driver.findElement(By.*name*("search")).sendKeys("CSS");

driver.switchTo().defaultContent(); //switches to the parent window

driver.findElement(By.*xpath*("html/body/a")).click();

}

}@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@S

**public** **class** AlphaNumericSpecial {

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

**int** count=0;

**int** num=0;

**int** spec=0;

**int** nums=0;

ArrayList<Character> arr= **new** ArrayList<>();

ArrayList<Character> arr1= **new** ArrayList<>();

ArrayList<Character> arr2= **new** ArrayList<>();

String s="AABae@io#b$u123";

**int** strLength= s.length();

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

**if** (Character.*isDigit*(s.charAt(i))) {

//if (s.charAt(i).contains("[0-9]"))){

num++;

//arr.add('s.charAt(i)');

arr.add(s.charAt(i));

}

**else** **if**(Character.*isAlphabetic*(s.charAt(i))) {

count++;

arr1.add(s.charAt(i));

}

**else** {

spec++;

arr2.add(s.charAt(i));

}

}

//==============================================

System.***out***.println("the digits are " + num +"and they are"+ arr);

// create object of StringBuilder class

StringBuilder sb = **new** StringBuilder();

// Appends characters one by one

**for** (Character ch : arr) {

sb.append(ch);

}

// convert in string

String string = sb.toString();

// print string

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

//=====================================================

System.***out***.println("no of alphabets are "+ count+"and they are"+ arr1);

StringBuilder sb1 = **new** StringBuilder();

// Appends characters one by one

**for** (Character ch1 : arr1) {

sb1.append(ch1);

}

// convert in string

String string2 = sb1.toString();

// print string

System.***out***.println(string2);

//============================================================

System.***out***.println("no of special characters are "+ spec+"and they are"+ arr2);

StringBuilder sb2 = **new** StringBuilder();

// Appends characters one by one

**for** (Character ch2 : arr2) {

sb2.append(ch2);

}

// convert in string

String string3 = sb2.toString();

// print string

System.***out***.println(string3);

}

}@@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** CountWordsInString {

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

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

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

String str = s.nextLine();

// calling the method to count the number of words in a string

**int** cnt = *countWordsUsingSplit*(str);

System.***out***.println("Number of words in string are - " + cnt);

}

**public** **static** **int** countWordsUsingSplit(String input) {

**if** (input == **null** || input.isEmpty()) {

**return** 0;

}

String[] words = input.split("\\s+");

**return** words.length;

}

}

/\*

\* In this solution, we will use the split() method of java.lang.String class to

\* count the number of words in a given sentence. This solution uses the regular

\* expression "\\s+" to split the String on whitespace. The split method returns

\* an array, the length of array is your number of words in given String.

\*

\* If you are new to regular expression in Java, the \s is a character class to

\* detect space including tabs, since \ needs to be escaped in Java, it becomes

\* \\s and because there could be multiple spaces between words we made this

\* regular expression greedy by adding +, hence \\s+ will find one more space

\* and split the String accordingly. See Core Java Volume 1 - Fundamentals by

\* Cay S. Horstmann to learn more about the split() method of String class. This

\* is also the simplest way to count the number of word in a given sentence.

\*/@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@ STRINGS

**public** **class** Demo1 {

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

String S1 = "Hello";

String S2 = **new** String("Bangalore");

System.***out***.println(S1+" "+S2);

System.***out***.println(S1);

}

}

//Output Hello Bangalore

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** Demo2 {

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

/\*String S1="Java";

String S2= new String("Developer");

String S3="Java";

String S4= new String("Developer");

System.out.println(S1==S3);

System.out.println(S2==S4);

String S5= new String("Developer");

System.out.println(S1==S5);

//}

/\*}

true

false

false\*/

/\*String S1="Java";

String S2= new String("Developer");

String S3="Java";

String S4= new String("Developer");

String S5= new String("Java");

System.out.println(S1==S3); //true

System.out.println(S2==S4); //false

System.out.println(S1==S5); //false

System.out.println(S1.equals(S3)); //true

System.out.println(S2.equals(S4)); //true

System.out.println(S1.equals(S5)); //true

System.out.println(S1.equals(S2)); //false

\*/

//String S2= new String("Developer");

//String S4= new String("Developer");

String S2= "Developer";

String S4= "Developer";

System.***out***.println(S2==S4); //false

System.***out***.println(S2.equals(S4)); //true

}

}

@@@@@@@@@@@@@@@@@@@@@@@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@

**public** **class** Demo3 {

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

/\*String S1="Java";

String S2= new String("Developer");

String S3="Java";

String S4= new String("Developer");

System.out.println(S1==S3);

System.out.println(S2==S4);

String S5= new String("Developer");

System.out.println(S1==S5);

}

}

true

false

false\*/

/\*String S1="Java";

String S2= new String("Developer");

String S3="Java";

String S4= new String("Developer");

String S5= new String("Java");

System.out.println(S1==S3); //true

System.out.println(S2==S4); //false

System.out.println(S1==S5); //false

System.out.println(S1.equals(S3)); //true

System.out.println(S2.equals(S4)); //true

System.out.println(S1.equals(S5)); //true

System.out.println(S1.equals(S2)); //false

\*/

String S2= **new** String("Developer");

String S4= **new** String("Developer");

System.***out***.println(S2);

System.***out***.println(S2==S4);

System.***out***.println(S2.equals(S4));

}

}

**public** **class** staleElementRefrenceExceptionDemo {

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

{

WebDriver driver;

System.*setProperty*("webdriver.chrome.driver",

"./Software/chromedriver.exe");

//"E:\\MyWorkspace\_Completed\\FreeCRMTest\\Software\\chromedriver.exe");

driver = **new** ChromeDriver();

driver.get("http://www.pavantestingtools.com/");

driver.manage().window().maximize();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***); //a[contains(text(),'Training')]

driver.findElement(By.*xpath*("//button[@id='close']//a[contains(text(),'X')]")).click();

Thread.*sleep*(2000);

WebElement link=driver.findElement(By.*xpath*("//a[contains(text(),'Training')]"));

link.click();

driver.navigate().back();

Thread.*sleep*(5000);

driver.findElement(By.*xpath*("//button[@id='close']//a[contains(text(),'X')]")).click();

/\*link=driver.findElement(By.xpath("//a[contains(text(),'Training')]"));

link.click();\*/

**try**

{ link.click();

}

**catch**(StaleElementReferenceException e)

{link=driver.findElement(By.*xpath*("//a[contains(text(),'Training')]"));

link.click();

}

}

}

**public** **class** staleElementRefrenceExceptionDemo2 {

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

WebDriver driver;

System.*setProperty*("webdriver.chrome.driver", "./Software/chromedriver.exe");

driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.get("https://www.freecrm.com/index.html");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***); // a[contains(text(),'Training')]

WebElement login = driver.findElement(By.*xpath*("//span[contains(text(),'Log In')]"));

login.click();

Thread.*sleep*(2000);

WebElement userName = driver.findElement(By.*xpath*("//input[@placeholder='E-mail address']"));

userName.sendKeys("subashtpatil@gmail.com");

WebElement pwd = driver.findElement(By.*xpath*("//input[@placeholder='Password']"));

pwd.sendKeys("test@123");

driver.navigate().refresh();

Thread.*sleep*(2000);

**try** {

userName.sendKeys("subashtpatil@gmail.com");

pwd.sendKeys("test@123");

**throw** **new** StaleElementReferenceException(**null**, **null**);

} **catch** (StaleElementReferenceException e) {

e.printStackTrace();

userName = driver.findElement(By.*xpath*("//input[@placeholder='E-mail address']"));

userName.sendKeys("subashtpatil@gmail.com");

pwd = driver.findElement(By.*xpath*("//input[@placeholder='Password']"));

pwd.sendKeys("test@123");

driver.findElement(By.*xpath*("//div[@class='ui fluid large blue submit button']")).click();

}

}

}

Constructor Overloading

**public** **class** A\_constOverldng {

**public** A\_constOverldng() {

System.***out***.println("parent class constructor");

}

**public** A\_constOverldng(**int** i) { //super() keyword calls this constructor

System.***out***.println("parent class constructor with value of i "+ i);

}

**public** A\_constOverldng(**int** i, **int** j) { //super() keyword calls this constructor

System.***out***.println("parent class constructor with value of i "+ i);

System.***out***.println("parent class constructor with value of j "+ j);

}

}

**public** **class** B\_constOverldng **extends** A\_constOverldng{ //B is child of A and also inherits it

**public** B\_constOverldng() {

**super**();

System.***out***.println("child class constructor");

}

**public** B\_constOverldng(**int** i) {

**super**(i);

System.***out***.println("child class constructor");

}

**public** B\_constOverldng(**int** i, **int** j) {

**super**(i, j);

System.***out***.println("child class constructor");

}

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

B\_constOverldng obj= **new** B\_constOverldng();

B\_constOverldng obj1= **new** B\_constOverldng(10);

B\_constOverldng obj2= **new** B\_constOverldng(10, 20);

}

}

/\*

parent class constructor

child class constructor

parent class constructor with value of i 10

child class constructor

parent class constructor with value of i 10

parent class constructor with value of j 20

child class constructor\*/

**package** oOPS;

**public** **class** ConstructorConcept {

**public** ConstructorConcept() {

System.***out***.println("Default constructor without any parameter");

}

**public** ConstructorConcept(**int** i) {

System.***out***.println("Single param constructor");

System.***out***.println("the value of i is "+i);

}

///Thius is also known as constructor overloading

**public** ConstructorConcept(**int** i, **int** j) {

System.***out***.println("two params constructor");

System.***out***.println("the value of i is "+i);

System.***out***.println("the value of j is "+j);

}

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

ConstructorConcept obj= **new** ConstructorConcept(); //default constructor with no parameters is called

ConstructorConcept obj1= **new** ConstructorConcept(10); //Calling the single parameter constructor

ConstructorConcept obj2= **new** ConstructorConcept(10, 20); //Calling the two parameter constructor

}

}

**public** **class** ConstructorWithThisKeyword {

// Class variables

String name;

**int** age;

**public** ConstructorWithThisKeyword(String name, **int** age) {

/\*

\* //local variables name=name; here local name is equal to local and global

\* name is unaffected this.global variable = local variable(whenever u want to

\* initialize ur global variable with the current value of constructor, v use

\* this k/w

\*/

**this**.name = name; // Now i can pass the name Tom from local to my class variable and initialize it

// using this keyword.

**this**.age = age;

}

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

ConstructorWithThisKeyword obj = **new** ConstructorWithThisKeyword("Tom", 30);// creating an object with name and

// age parameters

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

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

}

}

// Tom

// 30

**public** **class** MultilevelInheritance {

**protected** String str;

MultilevelInheritance() {

str = "One ";

}

}

**class** one **extends** MultilevelInheritance {

one() {

str = str.concat("two ");

}

}

**class** two **extends** one {

two() {

str = str.concat("three ");

}

}

**class** three **extends** two {

three() {

str = str.concat("four.");

}

**void** display() {

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

}

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

three obj = **new** three();

obj.display();

}

}

//[A, B, C]

/\*class MultilevelInheritanceMain {

public static void main(String args[]) {

three obj = new three();

obj.display();

}

}\*/

Method Overriding

**class** Bank {

**int** getRateOfInterest() {

**return** 0;

}

}

**class** SBI **extends** Bank {

**int** getRateOfInterest() {

**return** 9;

}

}

**class** ICICI **extends** Bank {

**int** getRateOfInterest() {

**return** 10;

}

}

**class** AXIS **extends** Bank {

**int** getRateOfInterest() {

**return** 11;

}

}

**class** MethodOverriding2 {

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

{

SBI s = **new** SBI();

ICICI i = **new** ICICI();

AXIS a = **new** AXIS();

System.***out***.println("SBI Rate of Interest: " + s.getRateOfInterest());

System.***out***.println("ICICI Rate of Interest: " + i.getRateOfInterest());

System.***out***.println("AXIS Rate of Interest: " + a.getRateOfInterest());

}

}

/\*SBI Rate of Interest: 9

ICICI Rate of Interest: 10

AXIS Rate of Interest: 11\*/

**package** MethodOverRiding;

//multiple inheritance is not supported in Java classes

//but a class can implement multiple interfaces

**class** A {

**void** msg() {

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

}

}

**class** B **extends** A {

**void** msg() {

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

}

}

**class** C **extends** A,B { //this is the line to be used actually for multiple inheritance error

//class C extends A {

//suppose if it were

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

C obj = **new** C();

obj.msg();// Now which msg() method of class A or class B would be invoked?

}

}

//Compile Time Error

Method Over Loading

**package** MethodOverLoading;

**public** **class** MainMethodOverLoading {

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

System.***out***.println("main with String[] args");

*main*();

*main*("main with String");

}

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

System.***out***.println("main with String");

}

**public** **static** **void** main(){

System.***out***.println("main without args");

}

}

**package** MethodOverLoading;

**public** **class** OverloadingExample {

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

/\*System.out.println(AdderClass.addition(11,12));

System.out.println(AdderClass.addition(11.3,15.6));\*/

System.***out***.println(AdderClass.*addition*(11,12));

System.***out***.println(AdderClass.*addition*(11.3,15.6));

}

}

**class** AdderClass{

**static** **int** addition(**int** a, **int** b){

**return** a+b;

}

**static** **double** addition(**double** a, **double** b){

**return** a+b;

}

}

/\*23

26.9\*/

**package** MethodOverLoading;

**class** Calculation {

**int** a,z;

**public** **void** addition(**int** x, **int** y) {

z = x + y;

System.***out***.println("The sum of the given numbers:" + z);

}

**public** **void** addition(**int** x, **int** y, **double** z) {

a = x +y+(**int**)z;

System.***out***.println("The sum of the given numbers:" + a);

}

}

**public** **class** My\_Calculation **extends** Calculation {

**public** **void** addition(**double** x, **int** y) {

z = (**int**)x + y;

System.***out***.println("The sum of the given numbers:" + z);

}

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

**int** a = 20, b = 10;

My\_Calculation demo = **new** Calculation();

demo.addition(a, b);

demo.addition(a, b, 10.5);

demo.addition(2.7, b); }

}

/\*

The sum of the given numbers:30

The difference between the given numbers:40

The product of the given numbers:12

\*/

Dynamic Polymorphism

**package** inheritance;

**public** **class** dynamicPolymorphism {

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

//parent class reference holds the child class object

Vehicle vh=**new** MotorBike();

vh.move(); // prints MotorBike can move and accelerate too!!

vh=**new** Vehicle();

vh.move(); // prints Vehicles can move!! }

}

}

**class** Vehicle{

**public** **void** move() {

System.***out***.println("Vehicles can move!!");

}

}

**class** MotorBike **extends** Vehicle{

**public** **void** move() {

System.***out***.println("MotorBike can move and accelerate too!!");

}

}

/\*MotorBike can move and accelerate too!!

Vehicles can move!!\*/