**Java**

**1) Reverse a string**

**public class Reverseprogram {**

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

**String word = "abcd";**

**for(int i=word.length()-1;i>=0;i--)s**

**{**

**System.out.print(word.charAt(i));**

**}}}**

**2) Following output:**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**public class Nested {**

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

**int i,j;**

**for(i=0;i<=5;i++)**

**{**

**for (j=0;j<=i;j++)**

**{**

**System.out.print("\*");**

**}**

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

**}}}**

**3) To create and throw custom exception**

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

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

**System.out.println("Enter a number b/w 1 and 10");**

**try**

**{**

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

**if(num<1 || num>10)**

**{**

**throw new ThrowCustomException("invalid num");**

**}}**

**catch (ThrowCustomException e)**

**{**

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

**}}}**

**4) Remove special character**

**public class Specialchar {**

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

**String rplc = "java%%\*\*";**

**String plainc =rplc.replaceAll("[^a-zA-Z0-9]","");**

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

}}

**int a[][]={{1,1},{2,2}};**

**5) Matrix Multiplication**

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

**int b[][]={{1,1},{2,2}};**

**int c[][]=new int[2][2]**

**for(int i=0;i<2;i++){**

**for(int j=0;j<2;j++){**

**c[i][j]=0;**

**for(int k=0;k<2;k++)**

**{**

**c[i][j]+=a[i][k]\*b[k][j];**

**}**

**System.out.print(c[i][j]+" ");**

**}**

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

**}**

**}}**

**6) Constructor**

**public class Student {**

**int id;**

**String name;**

**Student(){**

**System.out.println("this a default constructor");**

**}**

**Student(int i, String n){**

**id = i;**

**name = n;**

**}**

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

**{**

**//object creation**

**Student s = new Student();**

**System.out.println("\nDefault Constructor values: \n");**

**System.out.println("Student Id : "+s.id + "\nStudent Name : "+s.name);**

**System.out.println("\nParameterized Constructor values: \n");**

**Student student = new Student(10, "prakash");**

**System.out.println("Student Id : "+student.id + "\nStudent Name : "+student.name);**

**}**

**}**

**Automation Testing**

**1) Handling multiple windows**

**public class TestNaukri {**

**@Test**

**public void TestPopUp() throws InterruptedException{**

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

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

**driver.get("http://www.naukri.com/");**

**String parent=driver.getWindowHandle();**

**Set<String>s1=driver.getWindowHandles();**

**Iterator<String> I1= s1.iterator();**

**while(I1.hasNext())**

**{**

**String child\_window=I1.next();**

**if(!parent.equals(child\_window))**

**{**

**driver.switchTo().window(child\_window);**

**System.out.println(driver.switchTo().window(child\_window).getTitle());**

**driver.close();**

**}}**

**driver.switchTo().window(parent);**

**}**

**}**

**2) Select multiple options in dropdown**

**public class MultiDrpDwn{**

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

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

**"C:\Users\ghs6kor\Desktop\Java\geckodriver.exe");**

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

**Thread.sleep(5000);**

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

**driver.get("");**

**WebElement d= driverfindElement(By.xpath("//select[@name='selenium\_commands']"));**

**Select s=new Select(d);**

**List t =s.getOptions();**

**System.out.println("Options are: ");**

**for (WebElement i: t){**

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

**}**

**Boolean b=s.isMultiple();**

**System.out.println("Boolen value for drodown: "+ b);**

**s.selectByIndex(2);**

**Thread.sleep(1000);**

**s.selectByVisibleText("Wait Commands");**

**Thread.sleep(1000);**

**WebElement f = s.getFirstSelectedOption();**

**System.out.println("First selected option is: "+ f.getText());**

**driver.close();**

**}}**

**3) Upload files**

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

**System.setProperty("webdriver.gecko.driver", "Path of the gecko driver");WebDriver driver = new FirefoxDriver();**

**driver.get("https://smallseotools.com/plagiarism-checker/");**

**WebElement upload =driver.findElement(By.linkText(“Upload a Document:( )”);**

**upload.sendKeys("D:\\path\\File.txt");**

**driver.close();**

**}}**

**4) Frame Handling**

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

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

**driver.get("https://demoqa.com/frames");**

**WebElement iframeElement = driver.findElement(By.id("IF1"));**

**driver.switchTo().frame(iframeElement);**

**driver.quit();**

**Manual Testing**

**1) Redbus booking**

**Test Case: Book a Bus Ticket on Redbus**

**Preconditions:**

* **User has a valid Redbus account and is logged in.**
* **User has sufficient funds in their wallet or has linked a valid payment method.**
* **User knows the bus route, date of travel, and other details.**

**Test Steps:**

* **Open the Redbus website.**
* **Select the "Bus Tickets" option.**
* **Enter the source city, destination city, date of travel, and click on "Search Buses".**
* **Select the bus and seat(s) of choice.**
* **Enter the passenger details and click on "Continue Booking".**
* **Review the fare details and click on "Continue Booking".**
* **Select the preferred boarding and drop-off points and click on "Continue Booking".**
* **Review the booking details and click on "Continue Booking".**
* **Select the payment method and complete the payment process.**
* **Verify that the ticket is booked successfully and the user receives a confirmation message and ticket details.**

**Expected Results:**

* **User is able to navigate to the booking page and enter all required details.**
* **User is able to select the bus, seats, and passenger details without any errors.**
* **User is able to review the fare and booking details and proceed to payment.**
* **User is able to select the payment method and complete the payment process.**
* **User receives a confirmation message and ticket details after the booking is successful.**

**Post-conditions:**

* **The user's Redbus account is debited with the ticket fare.**
* **The user can view the booked ticket details in their Redbus account.**
* **The user can cancel the ticket before the journey date, if required.**

**2) Negative test case for IRCTC website**

**Invalid login credentials:**

**Attempt to login to the IRCTC website using an invalid username and password combination. Verify that the user is not able to login and an appropriate error message is displayed.**

**Invalid train number:**

**Attempt to book a train ticket using an invalid train number. Verify that the user is not able to proceed with the booking and an appropriate error message is displayed.**

**Insufficient funds:**

**Attempt to book a train ticket without having sufficient funds in the wallet or with an invalid payment method linked. Verify that the user is not able to complete the payment process and an appropriate error message is displayed.**

**Invalid journey date:**

**Attempt to book a train ticket using an invalid journey date. Verify that the user is not able to proceed with the booking and an appropriate error message is displayed.**

**3) Test cases for amazon mobile view**

**Navigation:**

**Verify that the user is able to navigate to different sections of the Amazon website using the mobile view. Check that the navigation menu is accessible and functional.**

**Search:**

**Verify that the search functionality works correctly in the mobile view. Test searching for products with different keywords and filters.**

**Product details:**

**Verify that the user is able to view the details of a product, including the product name, image, description, price, and reviews.**

**Add to cart:**

**Verify that the user is able to add a product to the cart from the mobile view. Check that the cart icon is updated with the correct number of items.**

**Checkout:**

**Verify that the user is able to proceed to checkout from the cart page. Check that the checkout process is mobile-friendly and easy to use.**

**Payment:**

**Verify that the user is able to complete the payment process using the mobile view. Test with different payment methods and check that the transaction is successful.**

**Account management:**

**Verify that the user is able to manage their account (e.g. login, view orders, edit profile) using the mobile view.**

**Wishlist:**

**Verify that the user is able to add products to their wishlist from the mobile view. Check that the wishlist is accessible and easy to use.**

**Ratings and reviews:**

**Verify that the user is able to view and submit ratings and reviews for products using the mobile view. Check that the rating and review forms are functional.**

**Performance:**

**Verify that the Amazon mobile view is fast and responsive, with no significant performance issues. Test with slow internet speeds and different mobile devices to ensure compatibility.**

**4) Test case for IRCTC website**

**Test Case: Book a Train Ticket**

**Objective: To verify that a user can successfully book a train ticket using the IRCTC website.**

**Preconditions:**

* **User has a valid IRCTC account and is logged in.**
* **User has sufficient funds in their wallet or has linked a valid payment method.**
* **User knows the train number, journey date, and other details.**

**Test Steps:**

* **Open the IRCTC website.**
* **Click on the "Trains" tab.**
* **Enter the source station, destination station, journey date, and click on "Search Trains".**
* **Select the train and class of travel.**
* **Enter the passenger details and click on "Continue Booking".**
* **Review the fare details and click on "Continue Booking".**
* **Select the preferred berth type and click on "Continue Booking".**
* **Review the booking details and click on "Continue Booking".**
* **Select the payment method and complete the payment process.**
* **Verify that the ticket is booked successfully and the user receives a confirmation message and ticket details.**

**Expected Results:**

* **User is able to navigate to the booking page and enter all required details.**
* **User is able to select the train, class of travel, and passenger details without any errors.**
* **User is able to review the fare and booking details and proceed to payment.**
* **User is able to select the payment method and complete the payment process.**
* **User receives a confirmation message and ticket details after the booking is successful.**

**Post-conditions:**

* **The user's IRCTC account is debited with the ticket fare.**
* **The user can view the booked ticket details in their IRCTC account.**
* **The user can cancel the ticket before the journey date, if required.**