**1. Armstrong number**

int num=153;

int temp=num;

int sum=0;

while(num>0){

int rem=num%10;

num=num/10;

sum=sum+(rem\*rem\*rem);}

if (num==temp)

Systemp.out.println(temp+"Armstrong Number");

else

System.out.println(temp+"not a Armstrong number");}

**2. Prime number 1 to N**

int n=100;

for(int i=2;i<=n;i++){

int count=0;

for(int j=2;j<i;i++){

if (i%j ==0){

count++; } }

if (count==0){

System.out.println(i);

} } }}

**3.check the given number is prime number or not**

int n=97;

boolean prime =true;

for(int i=2;i<n;i++)

{

if (n%i==0);

prime = false ;

break;

}

if(prime == true)

System.out.println(n +"the given number prime Number");

else

System.out.println(n+"the given number is not a prime number:");

}}

**4. Reverse of Given String // String pallindrom or not**

String str="reverse";

String rev=" ";

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

{

rev=rev+str.charat(i);

}

System.out.println(rev)

}

**5. Reverse of Given number // number is a pallindrom or not**

int num=121;

int temp=num;

int sum=0;

while(num>0){

int rem=num%10;

num=num/10;

sum=(sum\*10)+rem;

}

if (sum ==temp)

System.out.println(temp+" number is Pallindrom");

else

System.out.print(temp+"num is pallindrom ")

}}}

**6. Occurannce\_ofeachCharater in given String**

String str="aaabbbbcccddd";

char[] s1=str.tocharArray();

for (char ch='a';ch<='z';ch++){

int count=0;

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

{

if (ch==s1[i]){

count++; }}

if(count>0)

System.out.println(ch+"-"+count);

}}}

**7.FactroalNumber**

int n=5;

int fact=1;

for(int i=2;i<=n;i++){

fact=fact\*(i);

}

System.out.println(fact);

}}

**8.fibonacci series**

int f=0, f1=1, f2;

System.out.println(f+" ");

System.out.println(f1+" ");

for(int i=1; i<=6;i++){

f2=f+f1;

System.out.print(f2+" ");

f=f1;

f1=f2;

}}}}

**9.count Char of String**

String str=" kingking";

int count=0;

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

if (str.charAt(i) !=' ')

count++;

}

System.out.println(i); }

**10.Reverse\_sentance**

String[] s= "you shall not pass ".split(" ");

String revsent=" ";

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

revsent = revsent+s[i]+" ";

}

System.out.println(revsent);

}}

**11.reverse of element in array**

int[] arr={2,4,5,9};

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

System.out,println(arr[i]+" ");

}}}

**12.Show how many duplicate in String**

String string1="jogarao";

char[] String=s1.toCharArray();

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

int count=1;

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

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

count++;

string[j]!='0';

} }

if(count>1 && String[i]!='0'){

System.out.println(string[i])

} }

**13.even number**

int num=50;

for(int i=1;i<=num;i++){

if(i%2==0){

system.out.println(i);} }}

**14. oddNumber**

public static void main(String[] args){

int num=50;

for(int i =1;i<=num;i++){

if(i%2!=0){

System.out.println(i);

}

}

}

**15.Prime number using Array**

int[] a = { 3, 4, 5, 9, 8 };

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

int count = 0;

for (int j = 1; j <= 10; j++) {

if (a[i] % j == 0) {

count++;

}

}

if (count == 2) {

System.out.println(a[i]);

}

}

}

**16.Remove\_duplicate\_charters\_In\_String**

String s = "java";

String s1 = " ";

for (int i = 0; i < s1.length(); i++) {

char ch = s.charAt(i);

if (s1.indexOf(ch) == -1) {

s1 = s1 + ch;

}

}

System.out.println(s1);

}

}

**17.sort the Array**

public static void main(String[] args) {

int[] array =new int[]{20,10,5,3,2}

Arrays.sort(array);

for(int i=0;i<array.lenth;i++)

{

System.out.println(" "+array[i]));

}

**18. string fetch the first letter from each word "Good Noon Mam"? GNM**

String str="Good Noon Mam";

String[] str1=str.split(" ");

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

{

System.out.print(str1[i].CharAt(0);

}

**19.Remove\_Spaces\_from\_String**

String str = "shyamsinga Roy rao avuna";

String s = " ";

String space = str.replace(" ", "");

System.out.println(space);

**20. given String alternative upercase lowecases**

String str="Joga";

String str1=str.toLowercases();

char [] ch =str1.toCharArray();

for(int i=0;i<str1.length;i=i+2){

ch[i]=character.toupercase(ch[i]);

}

System.out.println(ch);

**BDD**

#Author: your.email@your.domain.com

#Keywords Summary :

#Feature: List of scenarios.

#Scenario: Business rule through list of steps with arguments.

#Given: Some precondition step

#When: Some key actions

#Then: To observe outcomes or validation

#And,But: To enumerate more Given,When,Then steps

#Scenario Outline: List of steps for data-driven as an Examples and <placeholder>

#Examples: Container for s table

#Background: List of steps run before each of the scenarios

#""" (Doc Strings)

#| (Data Tables)

#@ (Tags/Labels):To group Scenarios

#<> (placeholder)

#""

## (Comments)

#Sample Feature Definition Template

@tag

Feature: Title of your feature

I want to use this template for my feature file

@tag1

Scenario: Title of your scenario

Given I want to write a step with precondition

And some other precondition

When I complete action

And some other action

And yet another action

Then I validate the outcomes

And check more outcomes

@tag2

Scenario Outline: Title of your scenario outline

Given I want to write a step with <name>

When I check for the <value> in step

Then I verify the <status> in step

Examples:

| name | value | status |

| name1 | 5 | success |

| name2 | 7 | Fail |

**TestRunner**

**@CucumberOptions(**

features= {".\\src\\test\\java\\sample.feature"},

tags=" @smoketest or @regressiontest",

glue= {"stepDefination\_sample","Hooksdom"}, //package add

dryRun = **false**,// complation

monochrome = **true**,// use to remove unwanted symbols

plugin = {"pretty"}// READ by simple

tags = "~@smoketest or @regressiontest"

tags= "not @smoketest"

)

**public** **class** TestRunnerSample **extends** AbstractTestNGCucumberTests{

}

**HOOKS**

**public** **class** hooks {

@Before(order = 0)

**public** **void** connectDB() {

System.***out***.println("Connect to data base ");

}

@After(order = 0)

**public** **void** disconnectDB() {

System.***out***.println("DisConnect to data base ");

}

**StepDefinations**

**public** **class** Login {

WebDriver driver;

@Given("open browser")

**public** **void** open\_browser() {

WebDriverManager.*chromedriver*().setup();

driver = **new** ChromeDriver();

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

driver.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(10));

}

@Given("enter Url")

**public** **void** enter\_url() {

driver.get("http://localhost:8888/");

}

@When("enter username")

**public** **void** enter\_username() {

driver.findElement(By.*name*("user\_name")).sendKeys("admin");

}

@When("entrt password")

**public** **void** entrt\_password() {

driver.findElement(By.*name*("user\_password")).sendKeys("admin");

**@tages**

Feature: Flipkart search

Background:

Given open Browser

And Enter Url

When Wait for Page

@smoketest

Scenario: search for mobile

And search for mobile

And Close Browser

@regressiontest

Scenario: search for TVMI

And search for Mi TV "MI TV"

And Close Browser

@integration

Scenario: search for TVMI

And search for Mi TV "MI TV"

And Close Browser

**BrokenLINKS ans IMGES**

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

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

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

WebDriverManager.*chromedriver*().setup();

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

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

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

*driver*.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(10));

List<WebElement> links = *driver*.findElements(By.*tagName*("a")); // **for links "a" and for Images "img"**

System.***out***.println("total links are " + " " + links.size());

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

WebElement ele = links.get(i);

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

// for links "href" and for Images " src"

*verifyLinkActive*(url);

}

}

**private** **static** **void** verifyLinkActive(String linkurl) {

{

**try** {

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

HttpsURLConnection urlcon =(HttpsURLConnection)url.openConnection();

urlcon.setConnectTimeout(3000);

urlcon.connect();

**if** (urlcon.getResponseCode() == 200) {

System.***out***.println(linkurl + "-" + urlcon.getResponseMessage());

} **else** **if** (urlcon.getResponseCode()==HttpsURLConnection.***HTTP\_NOT\_FOUND***) {

System.***out***.println(linkurl + "-" + urlcon.getResponseMessage() + " -- link is broken");

}

} **catch** (Exception e) {

}

}

}

}

**Listners:**

**public** **class** ListnersImplementation1 **implements** ITestListener {

ExtentHtmlReporter reporter;

ExtentReports reports;

ExtentTest test;

**public** **void** onTestStart(ITestResult result) {

test = reports.createTest(result.getMethod().getMethodName()+""+"Test PAssed");

}

**public** **void** onTestSuccess(ITestResult result) {

System.***out***.println(result.getMethod().getMethodName());

}

**public** **void** onTestFailure(ITestResult result) {

test.log(Status.***FAIL***,result.getMethod().getMethodName());

test.log(Status.***FAIL***,result.getThrowable());

**try** {

String path = ExtensrRepo.*Screenshort*(result.getMethod().getMethodName());

test.addScreenCaptureFromPath(path);

} **catch** (IOException e)

{

e.printStackTrace();

}

}

**public** **void** onTestSkipped(ITestResult result) {

test.log(Status.***SKIP***, result.getMethod().getMethodName() + " Test got Skipped");

}

**public** **void** onTestFailedButWithinSuccessPercentage(ITestResult result) {

}

**public** **void** onTestFailedWithTimeout(ITestResult result) {

}

**public** **void** onStart(ITestContext context) {

reporter = **new** ExtentHtmlReporter("./ExtentReports/SDET4.html");

reporter.config().setDocumentTitle("VTigerCRM");

reporter.config().setTheme(Theme.***STANDARD***);

reports = **new** ExtentReports();

reports.attachReporter(reporter);

reports.setSystemInfo("Name of application", "VTigercRM");

reports.setSystemInfo("Build.No", "4.2.8");

}

**public** **void** onFinish(ITestContext context) {

reports.flush();

}

}

**JAVASCRIPTEXCUTER USING SEND DATE**

JavascriptExecutor jse = ((JavascriptExecutor)driver);

WebElement email = driver.findElement(By.*name*("q")); jse.executeScript("arguments[0].value='joga';", email);

**LOG4J**

**public** **class** ApplicationUsingLOG4j {

Logger log = Logger.*getLogger*(ApplicationUsingLOG4j.**class**);

**public** WebDriver driver;

@BeforeSuite

**public** **void** database() {

log.info("connect dataBase");

}

@BeforeClass

**public** **void** openbrowser1() {

log.info("Open Browser");

WebDriverManager.*chromedriver*().setup();

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

**POM CLASS**

**public** **class** ContactCreatPage { // rule 1

WebDriver driver;

WebDriverUtil allactionsuse = **new** WebDriverUtil(driver);

String fisrtName = javafaker.*objForJavautil*().getFistName();

String lastName = javafaker.*objForJavautil*().getLastName();

@FindBy(xpath = ("//a[text()='Contacts']")) //rule no2 java ann

**private** WebElement createcontantmodule; // rule no 3 private

**public** WebElement getCreatecontantmodule() { // rule no 3 getters

**return** createcontantmodule;

}

**public** ContactCreatPage(WebDriver driver) {// rule no 4 constructor

**this**.driver = driver;

PageFactory.*initElements*(driver, **this**);

}

**public** **void** createcontantmodule() { // rule no 5 business login

createcontantmodule.click();

}

**WebDriver Utile**

**public** **class** WebDriverUtil {

WebDriver driver;

**public** WebDriverUtil(WebDriver driver) {

**this**.driver = driver;

}

**public** **void** maxwindow() {

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

}

**public** **void** pageLoadTime() {

driver.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(10));

}

// DropDown actions

**public** **void** selectValuefromDropdownvalue(WebElement element, String value) {

Select select = **new** Select(element);

select.selectByValue(value);

}

**public** **void** selectValuefromDropdownindex(WebElement element, **int** index) {

Select select = **new** Select(element);

select.selectByIndex(index);

}

**public** **void** movetoDropdownvisibletext(WebElement element, String text) {

Select select = **new** Select(element);

select.selectByVisibleText(text);

}

// Mouse Over Actions

**public** **void** movetoelement(WebElement element) {

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

act.moveToElement(element).build().perform();

}

**public** **void** wedriverwait(WebElement ele) {

WebDriverWait wait = **new** WebDriverWait(driver, Duration.*ofSeconds*(120));

wait.until(ExpectedConditions.*elementToBeClickable*(ele));

}

**public** **void** movetoElementRightClick(WebElement element) {

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

act.moveToElement(element).contextClick().build().perform();

}

**public** **void** movetoElementClick(WebElement element) {

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

act.moveToElement(element).contextClick().build().perform();

}

// AlartPopUps

**public** **void** acceptAlerts() {

driver.switchTo().alert().accept();

}

**public** **void** dissmisAlert() {

driver.switchTo().alert().dismiss();

}

**public** **void** sendkeydu(WebElement element, String keystosend) {

element.sendKeys(keystosend);

}

**public** **void** scrollbyposition(**int** x, **int** y) {

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("window.scrollBy,(" + x + "," + y + ")");

}

**private** **void** javascriptexecutersend(WebElement element) {

JavascriptExecutor js1 = (JavascriptExecutor) driver;

js1.executeScript("arguments[0], value'raja'", element);

}

// window handles

**public** **void** switchtowindows(String title) {

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

**for** (String id : allId) {

String value = driver.switchTo().window(id).getTitle();

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

**if** (value.equalsIgnoreCase(title)) {

**break**;

}

}

**Paraller execution:**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name=*"Suite"* thread-count=*"5"* parallel=*"tests"*>

<test name=*"TestRunner1"*>

<parameter name=*"BROWSER"* value=*"Chrome"*></parameter>

<classes>

<class name=*"vtiger\_Testcase.Tc001\_CreateOrganizationTest"* />

</classes>

</test> <!-- Test -->

<test name=*"TestRunner2"*>

<parameter name=*"BROWSER"* value=*"Chrome"*></parameter>

<classes>

<class name=*"vtiger\_Testcase.Tc002\_CreateOrganizationTest"* />

</classes>

</test> <!-- Test -->

<test name=*"TestRunner3"*>

<parameter name=*"BROWSER"* value=*"Chrome"*></parameter>

<classes>

<class name=*"vtiger\_Testcase.Tc003\_CreateContactTest"* />

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**Group Excutions:**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<groups>

<run>

<include name=*"smoketest"* />

</run>

</groups>

<test thread-count=*"5"* name=*"Test"*>

<parameter name=*"BROWSER"* value=*"Chrome"*/>

<classes>

<class name=*"vtiger\_Testcase.Tc001\_CreateOrganizationTest"* />

<class name=*"vtiger\_Testcase.Tc002\_CreateOrganizationTest"* />

<class name=*"vtiger\_Testcase.Tc003\_CreateContactTest"* />

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**Xpath:**

**SVG TAG**

driver.findElement(By.*xpath*("(//\*[local-name()='svg' and @class='icon icon-noon ' ])[last()]")).click();

**normalized Space**

driver.findElement(By.*xpath*("//div[normalize-space()='Rankings']")).click();

**sigletone\_concept**

**public** **class** sigletone\_concept {

**private** sigletone\_concept()

{

}

**public** **static** sigletone\_concept sssmethod()

{

sigletone\_concept stn=**new** sigletone\_concept();

**return** stn;

}

}

**Exceptions:**

**ElementNotVisibleException**: In spite of the element being present in the DOM, it is not visible (can not be interactive). For example, elements defined in HTML with *type =”hidden”*

**ElementNotSelectableException**: An element is disabled (can not be clicked/selected) in spite of being present in the DOM

**NoSuchElementException**: Webdriver is not able to determine the elements during runtime, i.e., the *FindBy* method cannot find a particular component

**NoSuchFrameException**: Webdriver attempts to switch to an invalid frame, which is unavailable

**NoAlertPresentException**: Webdriver is trying to switch to an invalid alert, which is unavailable

**NoSuchWindowException**: Webdriver is trying to switch to an invalid window, which is unavailable

**StaleElementReferenceException**: The referenced element is no longer present on the DOM page (a reference to a component is now Stale). For example, the item belongs to a different frame than the current one or the user has navigated away to another page

**SessionNotFoundException**: Webdriver is acting immediately after ‘quitting’ the browser

**TimeoutException**: The command did not complete in the specified time. For example, the element didn’t display at the specified time. This is especially encountered when working with waits

**WebDriverException**: Webdriver is acting immediately after ‘closing’ the browser

Now, as we are aware of the common Exceptions that one can face in Selenium WebDriver, the next step is to understand “***How to handle those Exceptions?***”

**Implicitly Wait**

**driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(10));**

**Explicity wait**

**WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(10));**

**wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector(".classlocator")));**

**Wait<WebDriver> fluentWait = new FluentWait<WebDriver>(driver)**

**.withTimeout(Duration.ofSeconds(30))**

**.pollingEvery(Duration.ofSeconds(5))**

**.ignoring(NoSuchElementException.class);**