Question-1

public class Account {

private int accountNumber;

private double balance;

private double interest;

public Account(int accountNumber, double balance) {

this.accountNumber = accountNumber;

this.balance = balance;

this.interest = 0;

}

public void calculateInterest() {

this.interest = this.balance \* 0.05;

}

public void addInterest() {

this.balance += this.interest;

}

public double getBalance() {

return this.balance;

}

}

public class SavingsAccount extends Account {

private double interestRate;

public SavingsAccount(int accountNumber, double balance, double interestRate) {

super(accountNumber, balance);

this.interestRate = interestRate;

}

@Override

public void calculateInterest() {

this.interest = this.balance \* this.interestRate;

}

@Override

public double getBalance() {

return this.balance + this.interest;

}

}

public class CurrentAccount extends Account {

private double overdraftLimit;

public CurrentAccount(int accountNumber, double balance, double overdraftLimit) {

super(accountNumber, balance);

this.overdraftLimit = overdraftLimit;

}

@Override

public void withdraw(double amount) {

if (this.balance - amount < this.overdraftLimit) {

System.out.println("Insufficient funds");

} else {

this.balance -= amount;

}

}

@Override

public double getBalance() {

return this.balance;

}

}

Question2-

class Student:

def \_\_init\_\_(self, name, grade, age):

self.name = name

self.grade = grade

self.age = age

def display(self):

print(f"Name: {self.name}")

print(f"Grade: {self.grade}")

print(f"Age: {self.age}")

class School(Student):

def \_\_init\_\_(self, name, grade, age, school\_name):

super().\_\_init\_\_(name, grade, age)

self.school\_name = school\_name

def school\_student\_display(self):

print(f"Name: {self.name}")

print(f"Grade: {self.grade}")

print(f"Age: {self.age}")

print(f"School Name: {self.school\_name}")

student = Student("John Doe", 10, 15)

student.display()

school\_student = School("Jane Doe", 11, 16, "Central High School")

school\_student.school\_student\_display()

Question-3

from selenium import webdriver

# Launch a Firefox browser

driver = webdriver.Firefox()

# Navigate to the MakeMyTrip website

driver.get("https://www.makemytrip.com/")

# Find the MakeMyTrip logo element

logo\_element = driver.find\_element\_by\_css\_selector("#makemytrip-logo")

# Check if the MakeMyTrip logo element is present

if logo\_element is not None:

print("MakeMyTrip logo is present on the page.")

else:

print("MakeMyTrip logo is not present on the page.")

# Close the browser

driver.quit()

Question-4

from selenium import webdriver

# Define a generic function to find an element by xpath

def find\_element\_by\_xpath(driver, xpath):

return driver.find\_element\_by\_xpath(xpath)

# Define a generic function to click on an element

def click\_element(driver, element):

element.click()

# Define a generic function to enter text into an element

def enter\_text\_in\_element(driver, element, text):

element.send\_keys(text)

# Launch a Chrome browser

driver = webdriver.Chrome()

# Navigate to the Makemytrip website

driver.get("https://www.makemytrip.com/")

# Find the Flights element

flights\_element = find\_element\_by\_xpath(driver, "//a[text()='Flights']")

# Click on the Flights element

click\_element(driver, flights\_element)

# Find the OneWay element

oneway\_element = find\_element\_by\_xpath(driver, "//input[@id='oneway']")

# Click on the OneWay element

click\_element(driver, oneway\_element)

# Find the FROM element

from\_element = find\_element\_by\_xpath(driver, "//input[@id='fromCity']")

# Enter text into the FROM element

enter\_text\_in\_element(driver, from\_element, "Hyderabad")

# Find the TO element

to\_element = find\_element\_by\_xpath(driver, "//input[@id='toCity']")

# Enter text into the TO element

enter\_text\_in\_element(driver, to\_element, "Bengaluru")

# Close the browser

driver.quit()

Question-5

import unittest

from selenium import webdriver

class TestNGConcepts(unittest.TestCase):

@BeforeMethod

def launch\_browser(self):

self.driver = webdriver.Chrome()

@Test

def test\_case\_1(self):

self.driver.get("https://www.makemytrip.com/")

self.assertIn("MakeMyTrip", self.driver.title)

@Test

def test\_case\_2(self):

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

self.assertIn("Google", self.driver.title)

@Test

def test\_case\_3(self):-6

self.driver.get("https://www.yahoo.com/")

self.assertIn("Yahoo", self.driver.title)

@AfterMethod

def close\_browser(self):

self.driver.quit()

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

question-6

mvn archetype:generate -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

This will create a new Maven project in the current directory.

To implement the programs you mentioned in the Maven project, you can create a new Java class for each program and add the necessary code to the class. You can also add the necessary JAR dependencies to the project's pom.xml file.

To run the programs using Maven commands, you can use the following command:

mvn exec:java -Dexec.mainClass=<your\_main\_class>

For example, to run the first program, you would use the following command:

mvn exec:java -Dexec.mainClass=com.example.TestNGConcepts

This will compile the program and run it using the TestNG runner.

Question-7

<https://github.com/Neeraja1118/neeraja-sdet-assessment-repo>

Question-8

Collection: Universities API

Request 1: Get Universities in United States

Method: GET

URL: http://universities.hipolabs.com/search?country=United+States

Assertion:positive test case

Status Code: 200

Request 2: Get Universities in Invalid Country

Method: GET

URL: http://universities.hipolabs.com/search?country=Invalid Country

Assertion:negative test case

Status Code: 404

Question-9

Create a Thread Group

1. Open JMeter.
2. Click the Test Plan node in the tree.
3. Right-click and select Add > Threads (Users) > Thread Group.
4. Enter a name for your thread group and click Add.

Send a request to the Makemytrip website

1. Right-click on the Thread Group node and select Add > Sampler > HTTP Request.
2. Enter the following URL in the Server Name or IP field:

www.makemytrip.com

1. Enter the following path in the Path field:

/

1. Select the GET method.

Add assertions

1. Right-click on the HTTP Request node and select Add > Assertion > Response Assertion.
2. Select the Response Code assertion type.
3. Enter the expected response code in the Expected field. This will typically be 200 for a successful request.
4. Click the Add button.

Use listeners

1. Click the Listeners tab.
2. Right-click and select Add > Listener > View Results Tree. This will display the request and response data for each thread.
3. Right-click again and select Add > Listener > Assertion Results. This will display a table of the assertion results for each thread.

Run the test

Click the green Run button in the toolbar.

View the assertion results

Click the Assertion Results tab. The results will be displayed in a table. If all of the assertions passed, then the test was successful.

Question-10

###### **Import libs**

import pytest from selenium import webdriver

###### **Fixture to launch the browser**

@pytest.fixture(scope="module") def driver(): driver = webdriver.Chrome() driver.implicitly\_wait(10) yield driver driver.quit()

###### **Test case to verify the W3Schools logo is present on the page**

def test\_w3schools\_logo(driver): driver.get("https://www.w3schools.com/")

# Find the W3Schools logo element

logo\_element = driver.find\_element\_by\_css\_selector("#w3schools-logo")

# Verify that the logo element is present

assert logo\_element is not None

# Get the logo element's text

logo\_text = logo\_element.text

# Verify that the logo text is "W3Schools"

assert logo\_text == "W3Schools"

###### **Run the test case**

if **name** == "**main**": pytest.main()

This program uses the following pytest fixture:

Python

@pytest.fixture(scope="module")

def driver():

driver = webdriver.Chrome()

driver.implicitly\_wait(10)

yield driver

driver.quit()

Use code with caution. [Learn more](https://bard.google.com/faq#coding)

content\_copy

This fixture launches a Chrome browser and makes it available to all of the test cases in the module. The scope="module" parameter ensures that the browser is only launched once for all of the test cases in the module.

The following test case verifies that the W3Schools logo is present on the page:

Python

def test\_w3schools\_logo(driver):

driver.get("https://www.w3schools.com/")

# Find the W3Schools logo element

logo\_element = driver.find\_element\_by\_css\_selector("#w3schools-logo")

# Verify that the logo element is present

assert logo\_element is not None

# Get the logo element's text

logo\_text = logo\_element.text

# Verify that the logo text is "W3Schools"

assert logo\_text == "W3Schools"

Use code with caution. [Learn more](https://bard.google.com/faq#coding)

content\_copy

To run the test case, simply execute the following command:

pytest test\_w3schools\_logo.py

If the test case passes, then the W3Schools logo is present on the page.