Google F2F Interview Qns

Programs

1. Write a program to find sum of odd numbers in an array.

**import** java.util.Scanner;

**public** **class** OddNos {

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

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

System.***out***.println("enter size of array");

**int** n=sc.nextInt();

**int**[]arr= **new** **int**[n];

System.***out***.println("enter the elements");

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

arr[i]=sc.nextInt();

}

**int** sumodd=0;

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

**if**(arr[i]%2!=0){

sumodd+=arr[i];

}

}

System.***out***.println("Sum of odd nos is : "+sumodd);

}

}

2.Write a program to do selection sort.

Algorithm

arr[] = 64 25 12 22 11

Set indexmin to index of first element

// Find the minimum element in arr[0...4]

// and place it at beginning

25 12 22 64

// **11**Find the minimum element in arr[1...4]

// and place it at beginning of arr[1...4]

11 **12** 25 22 64

// Find the minimum element in arr[2...4]

// and place it at beginning of arr[2...4]

11 12 **22** 25 64

// Find the minimum element in arr[3...4]

// and place it at beginning of arr[3...4]

11 12 22 **25** 64

3.Write a program to do bubble sort.

Algorithm

Bubble sort starts with very first two elements, comparing them to check which one is greater.

Bubble Sort

Bubble Sort

Bubble Sort

Bubble Sort

Bubble Sort

Bubble Sort

Bubble Sort

Bubble Sort

After the second iteration, it should look like this −

Bubble Sort

After the third iteration, it should look like this −

Bubble Sort

After the fourth iteration, it should look like this −

Bubble Sort

**Program to demonstrate bubble and selection sort**

**public** **class** BubbleAndSelectionSort {

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

**int**[] arr={17,34,21,67,45,98,32,90};

*bub*(arr);

*sel*(arr);

}

**public** **static** **void** bub(**int**[] arr){

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

**for**(**int** j=0;j<(arr.length-1);j++){

**if**(arr[j]>arr[j+1]){

**int** temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

System.***out***.println("BubbleSorted array");

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

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

}

}

**public** **static** **void** sel(**int**[] arr){

**int** l=arr.length;

**for**(**int** i=0;i<l-1;i++){

**int** indexmin=i;

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

**if**(arr[j]<arr[indexmin]){

//indexmin=i;

indexmin=j;

}

}

**if**(indexmin!=i){

**int** temp=arr[i];

arr[i]=arr[indexmin];

arr[indexmin]=temp;

}

}

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

System.***out***.println("SelectionSorted array");

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

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

}

}

}

TESTING

1. What are the test cases for testing a gmail application.

* Go to browser->type the url www.gmail.com-> shd go to login page->with cursor on username
* Type correct username and pwd-> shd go to gmail home page
* Type incorrect username and pwd-> shd display an error message->Invalid uname or pwd, try again.
* In Gmail homepage-> shd see the mails displayed-> unread shd be bold and read shd not be bold.
* Check the correct segregation of mails->like Primary,Social,Promotions etc
* Check whether mails are correctly displayed in spam,inbox,sent folders
* Check the compose functionality->shd open up a word pad and be able to edit it ans chk whether attachments can be included.
* Check signout functionality

2.a) Suppose Gmail is introducing a new application called snooze, give the test cases for that.

* Snooze will be like dismissing the mails for a certain period of time.
* Snooze shd be clickable
* Snooze to suggested time options- like hours,minutes,seconds
* Snoozed items shd be under Snoozed list
* Snooze shd work on mobile for network variations
* Chk that snooze is not working when u disable the snooze functionality temporarily

b) Suppose the person who is using this app moves from west coast to east coast ,give the scenarios to test that.

Check localization is working properly.

3. What are the challenges that you have faced in testing using selenium webdriver.

* Wait times are not unique in each browsers.Should identify wait times that is not too long or too short.
* Sometimes it is difficult to get hold of dynamically changing elements.
* Cross browser testing- IE is not always consistent.There are issues with IE especially IE9
* Safari browser have some issues with css selectors.
* Desktop based interactions- go for auto IT or Robot class
* Angular JS testing appln- Selenium do not support angular based locators-Use Protractor.
* Regression Tests fail in different environments due to data mismatch.

4.How much manual testing and automation you use in a project.

Manual testing- 30 % and Automation testing-70%

In a particular sprint,When you get user stories, start with manual testing and proceed with automation testing in the following sprint.

5.Suppose there is an issue with you and developer, how will you resolve it.

* one to one email communication with developer and tester.
* If it does not work, meet the developer in person and have a one to one discussion.
* If it is not resolved, contact the BA and get a clear picture of the requirements.
* Make sure, there is an email confirmation on the requirements before proceeding.

6.What is meant by automation framework flakiness. What are the reasons for that.

Automation Flakiness is the instablility in automation framework.

Though Passes test 90% , fails by 10%

Wait Time-Same wait time do not work for different browsers. Ideal wait time should be selected which is not too small or too long

Locators that can be changed frequently by developers shd not be selected. Choose good locators like id or css selectors.

Proper exception handling not done

Dependent test cases-Whenever you write,try to make test cases independent of each other.

Dynamic Data-If you assert for a hard coded data in the program, which changes dynamically. Go with AssertNotNull or assert True, rather than going for assert equals.

Test that depends on API calls or webservice calls-fails when service is down or slow.

7.What have you done in your previous project other than the normal tasks. Did you do anything other than the normal tasks assigned to you.

Attempts to introduce an API layer, in the automation FW that calls API’s directly from UI application and fetch the dynamically changing data.Actual data to be asserted is obtained through API calls.

**7.Challenges in UI Automation using Selenium WebDriver**

* Wait times are not unique in each browsers.Should identify wait times that is not too long or too short.
* Sometimes it is difficult to get hold of dynamically changing elements.
* Cross browser testing- IE is not always consistent.There are issues with IE especially IE9
* Safari browser have some issues with css selectors.
* Desktop based interactions- go for auto IT or Robot class
* Angular JS testing appln- Selenium do not support angular based locators-Use Protractor.
* Regression Tests fail in different environments due to data mismatch.

**8.Challenges in API Testing**

* Parameter Combination-Each API will take various input parameters and hence there will be huge number of combinations of input params. Combination will get increased with more number of optional parameters.
* Sequence of API calls-If there is API testing which involves sequence of API calls, all the sequences flows should be tested in the correct order. API calls need to appear in a specific order to work correctly. This creates a sequencing challenge for the testing team. For example, if a call to return a user’s profile information goes through before the profile is created, the request will return an error.
* Validating and verifying-If there are large number of parameters to be tested, each parameter should be validated. It can be a number range, maximum length restrictions, acceptance of specific string/value. The team needs to make sure all parameter data uses the correct string or numerical data type, fits within length restrictions, fits within a designated value range, and passes other validation criteria. For example, U.S. phone numbers should appear in a 10-digit format, and returning a 5-digit zip code should trigger an invalidation error.
* Schema updates- The input/output schema skeleton may vary time to time due to business requirements. The coverage should be extended for that.

**9.Challenges in Mobile Automation**

* Configuration and Fragmentation -Wide number of devices and number of browsers. Must have good knowledge to configure many devices in Android Virtual Device Manager.
* Locating elements -Need to write Xpath to locate element uniquely sometimes when you don’t have any unique id or name.
* Lack of system memory -To run the test across 15 to 20 devices.