Experiment-1: Recording test in analog and context sensitive mode

Aim: To record a test in analog and context sensitive modes

Theory:

By recording, we can quickly create automated test scripts. While working with an application like clicking objects with the mouse and entering keyboard input, WinRunner records our operations and generates statements in TSL, Mercury Interactive's Test Script Language. These statements appear as a script in a WinRunner test window.

Two recording modes are available in WinRunner:

1. **Context Sensitive**
2. **Analog.**

1. Context Sensitive

Context Sensitive mode records the operations in terms of the GUI objects in application. WinRunner identifies each object we click (such as a window, menu, list, or button), and the type of operation we perform (such as press, enable, move, or select).

For example, if we record a mouse click on the OK button in the Flight Reservation Login window, WinRunner records the following TSL statement in test script:

button_press ("OK");

When we run the script, WinRunner reads the command, looks for the OK

button, and presses it.

2. Analog

In Analog mode, WinRunner records the exact coordinates traveled by the mouse, as well as mouse clicks and keyboard input. For example, if you click the OK button in the Login window, WinRunner records statements that look like this:

Recorded statements	**meaning...**
move_locator_track (1);	mouse track
mtype ("<T110><kLeft>-");	left mouse button press
mtype ("<kLeft>+");	left mouse button release

When we run the test, WinRunner retraces the recorded movements using absolute screen coordinates. Recording in Analog mode should be done only when exact mouse movements are an important part of test

Procedure:

Recording in Context Sensitive Mode:

Here we will create a script that tests the process of opening an order in the Flight Reservation application. A script is created by recording in Context Sensitive mode.

Steps:

1.Open WinRunner. -Choose Programs > WinRunner > WinRunner on the Start menu.

2 Create a new test. Choose File > New. A new test window opens in WinRunner.

3 Start the Flight Reservation application and log in.

Choose Programs > WinRunner > Sample Applications > Flight 1A on the Startmenu.

In the Login window, type your name and the password as mercury, and clickOK.

4. Start recording in Context Sensitive mode. Choose Create > Record—Context Sensitive

5.**Open order #3**. In the Flight Reservation application, **choose File > Open Order**.

In the Open Order dialog box, select the Order No. check box. Type 3 in the adjacent box, and click OK.

6 Stop recording --In WinRunner, **choose Create > Stop Recording**

7 **Save the test-- Choose File > Save**. Save the test with name in a convenient location on your hard drive. Click Save to close the Save Test dialog box.

Recording in Analog Mode:

Here we will test the process of sending a fax. We will start recording in Context Sensitive mode, switch to Analog mode in order to add a signature to the fax, and then switch back to Context Sensitive mode.

Steps:

After recording in context sensitive mode above, save the test in winrunner and keep open the flight reservation window

1.Open the previously saved test file &Start recording in context sensitive mode

2. Open the Fax Order form from File option in Flight reservation window and fill in a fax number.

2 Select the Send Signature with Order check box.

3 Sign the fax again in Analog mode.

4 Stop Recording.

5 Save the test.

Running the Test

1Check that WinRunner and the main window of the Flight Reservation application are open

2Make sure that the saved test window is active in WinRunner.

3Make sure the main window of the Flight Reservation application is active.

4 Select Verify mode in the toolbar.

5. Choose Run > Run from Top

6. Click OK in the Run Test dialog box

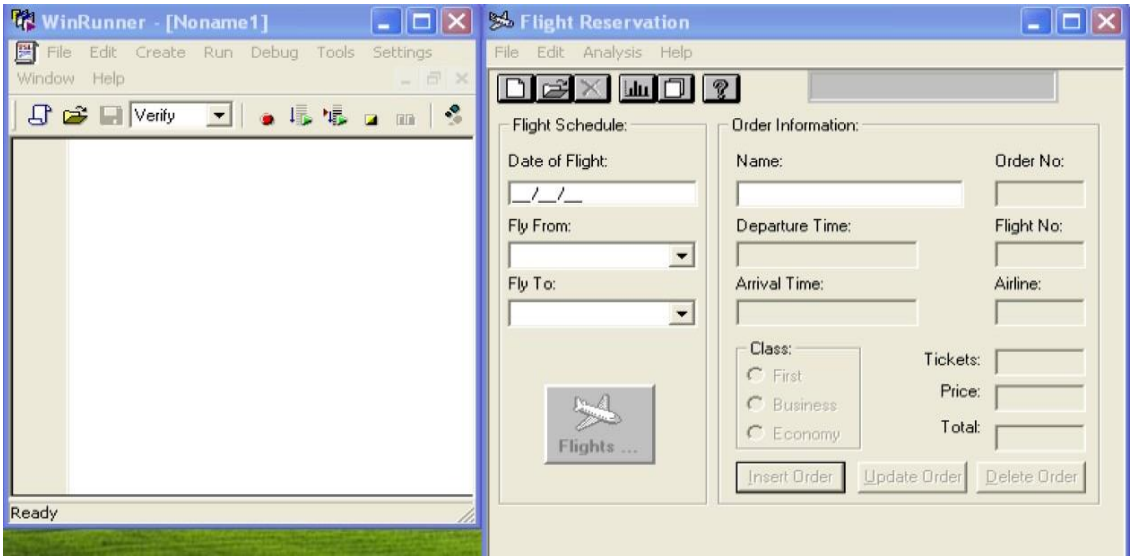
7. Review the test results.

8. Close the test results and application

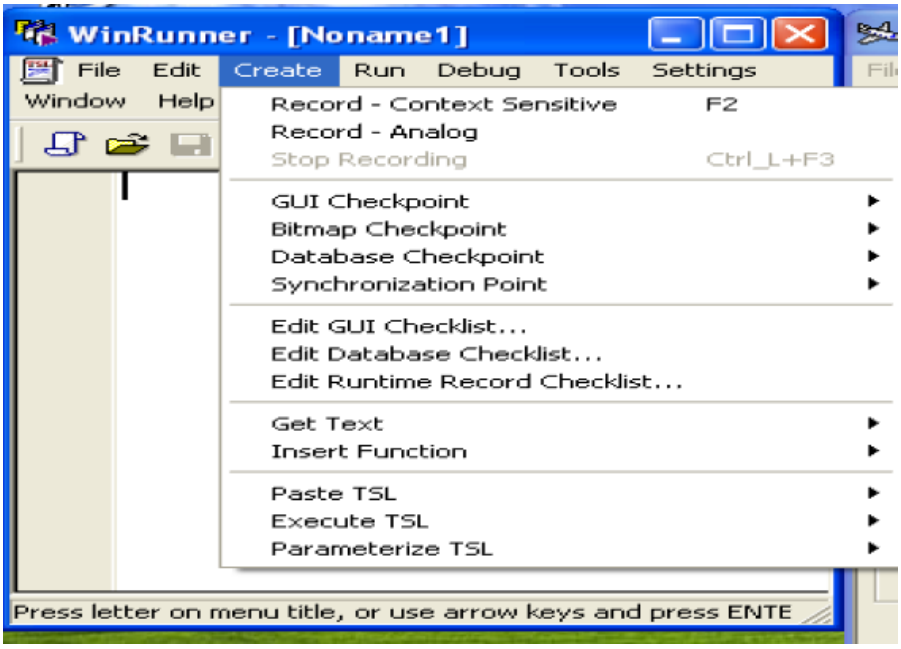
Output:

Recording in Context Sensitive Mode

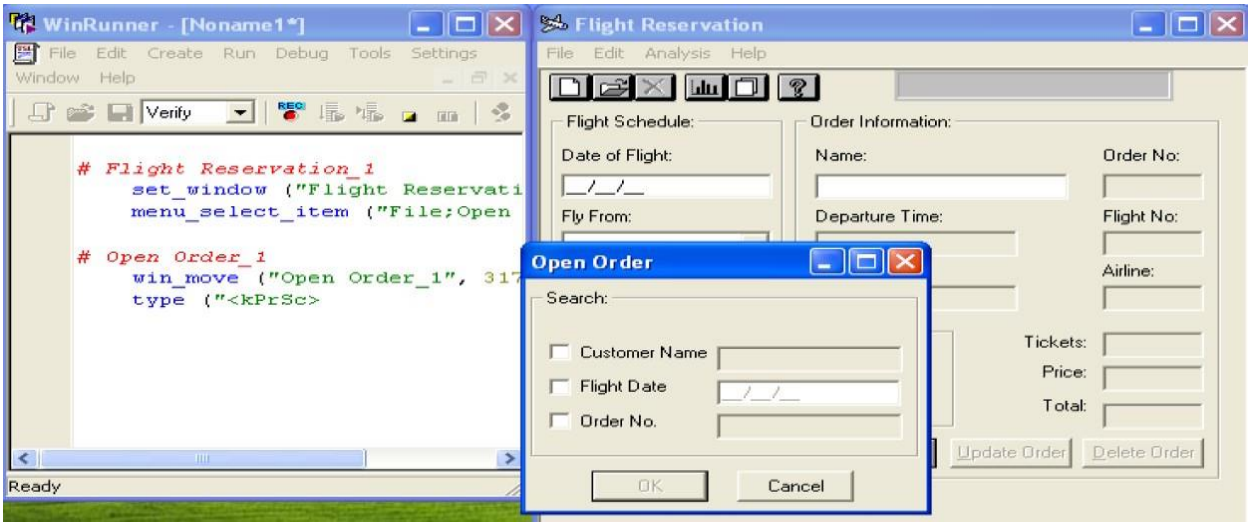
WinRunner Tool and Flight Reservation Application opened



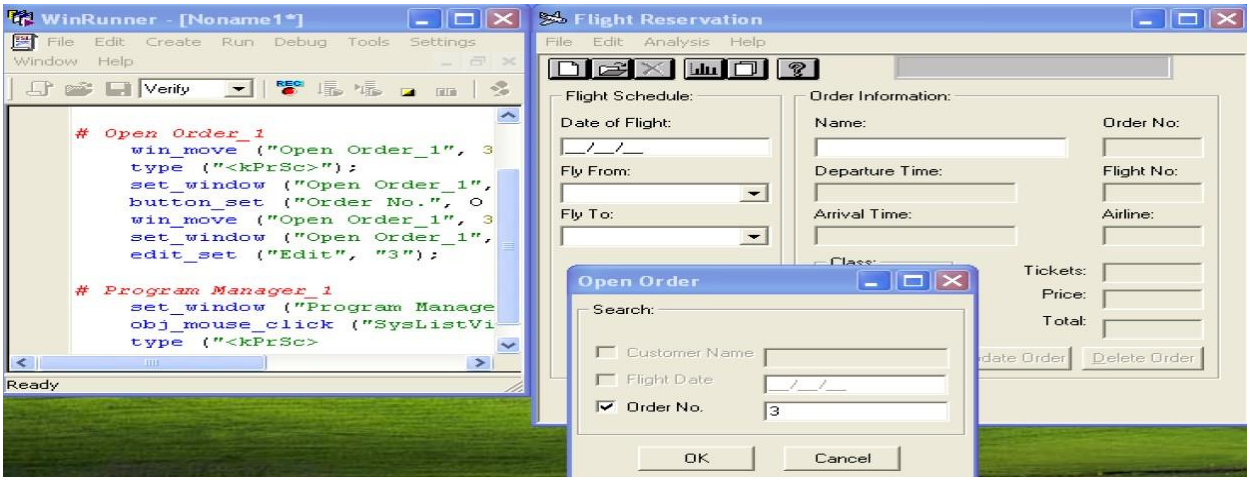
Selecting Record in Context Sensitive Mode using WinRunner tool



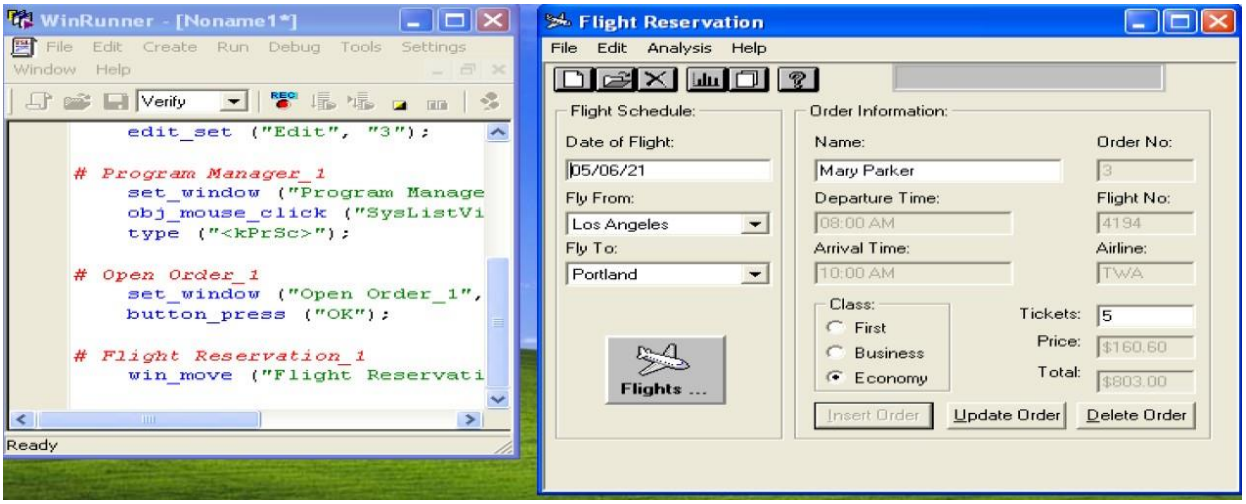
Test Script is generated in WinRunner when the Flight Reservation window is moved and open order is selected



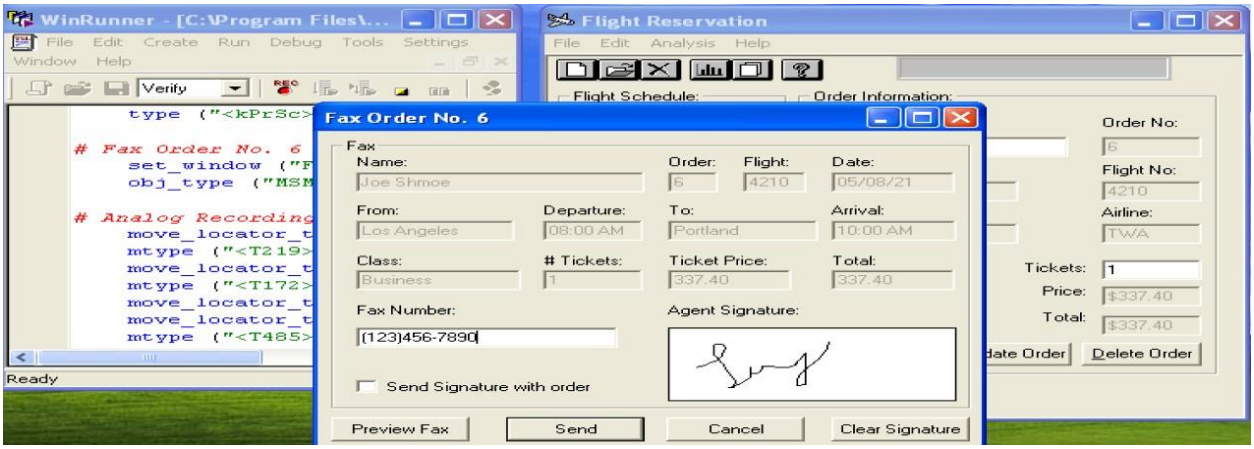
Selecting order, no 3 in open order window and clicking ok, test script is generated in winrunner



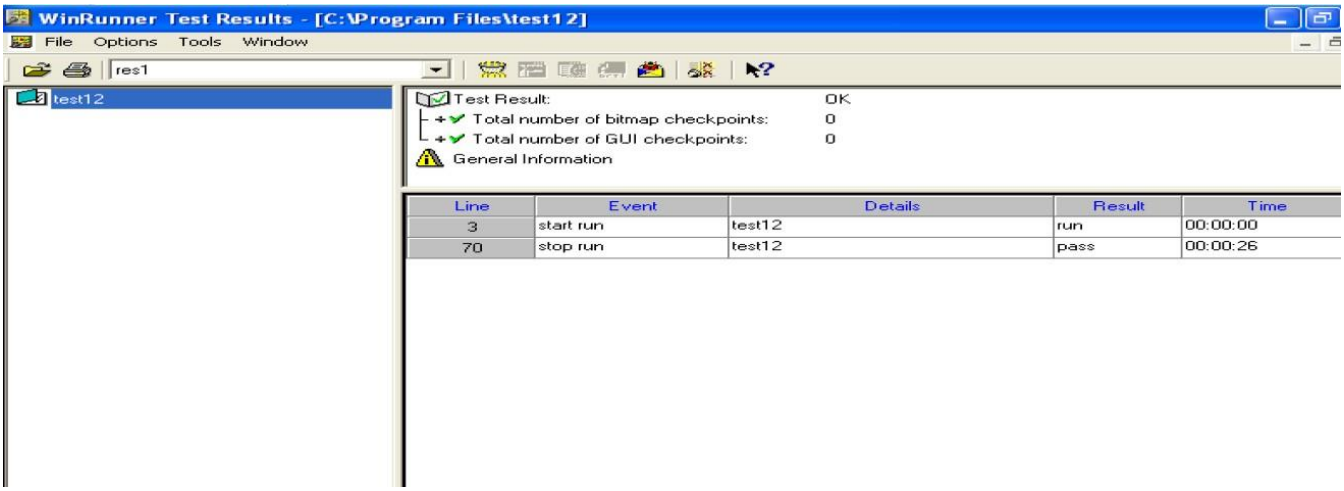
Stop the recording and saving the Test in WinRunner

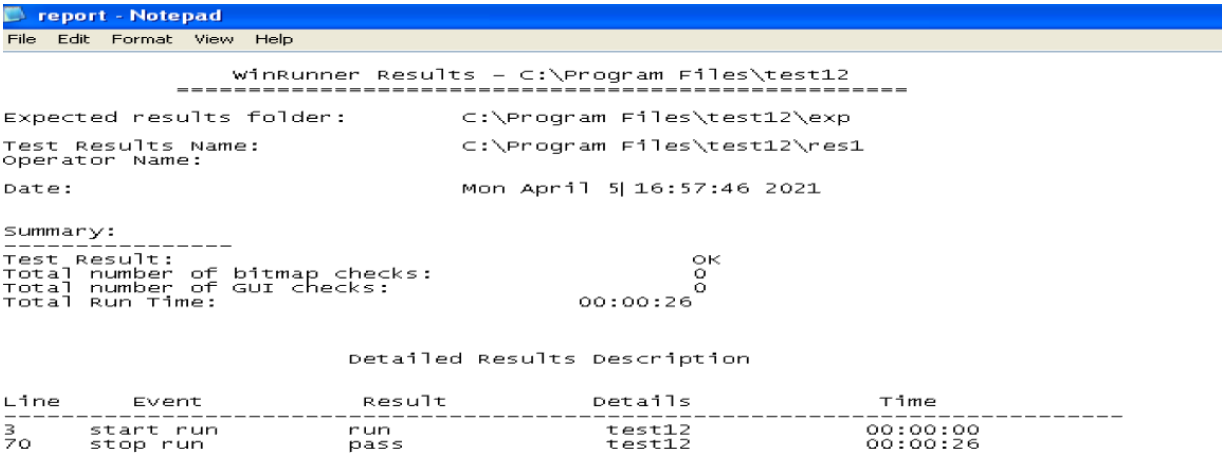


Recording in Analog Mode



Viewing Test Result





report - Notepad

File Edit Format View Help

WinRunner Results – C:\Program Files\test12

Expected results folder: C:\Program Files\test12\exp

Test Results Name: C:\Program Files\test12\res1
Operator Name:

Date: Mon April 5| 16:57:46 2021

Summary:

Test Result: OK
Total number of bitmap checks: 0
Total number of GUI checks: 0
Total Run Time: 00:00:26

Detailed Results Description

Line	Event	Result	Details	Time
3	start run	run	test12	00:00:00
70	stop run	pass	test12	00:00:26

Result: Recording in analog & context sensitive modes are done successfully

Experiment-2: GUI Checkpoint for Single Property

Aim: To create GUI checkpoint for single property.

Theory:

Behavior of Graphical User Interface objects in an application are verified by GUI checkpoints using wnrunner tool. **A GUI checkpoint** examines the behavior of an object's properties.

For example, you can check:

the content of a field, if a radio button is on or off, if a pushbutton is enabled or disabled

Procedure:

1 Start WinRunner and create a new test.

2 Start the Flight Reservation application and log in.

3 Start recording in Context Sensitive mode.

4 Open the Open Order dialog box from flight reservation application

5 Create a GUI checkpoint for the Order No. check box.

Choose Create >GUI Checkpoint>For Single Property

Use the pointer to double-click the Order No. check box. The Check GUI dialog box opens and displays the available checks.

6. Stop recording

7. Save the test.

GUI checkpoints appear as obj_check_gui and win_check_gui statements inthe test script.

For example **obj_check_gui("Order No.", "list1.ckl", "gui1", 1)**

Order No. is the object's logical name.

list1.ckl is the checklist containing the checks you selected.

gui1 is the file containing the captured GUI data.

1 is the time (in seconds) needed to perform the check.

To Run the Test

1. Make sure the above saved test and Flight Reservation application are open.

2 In WinRunner, check that Verify mode is selected in the Standard toolbar.

3.Choose Run > Run from Top,

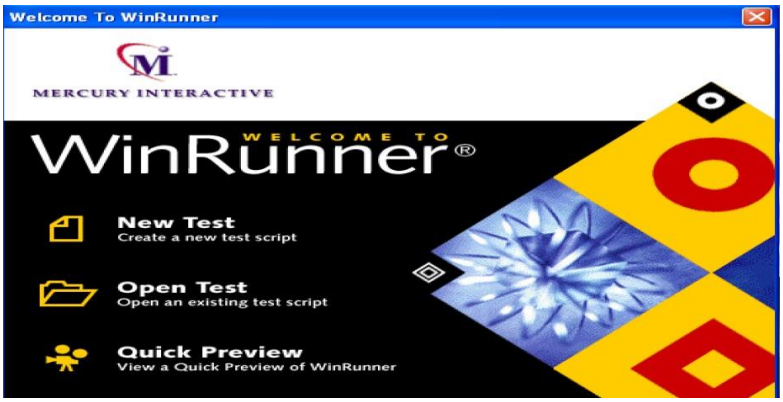
4. The Run Test dialog box opens. Accept the default test run name “res1.”

Run the test Click OK in the Run Test dialog box.

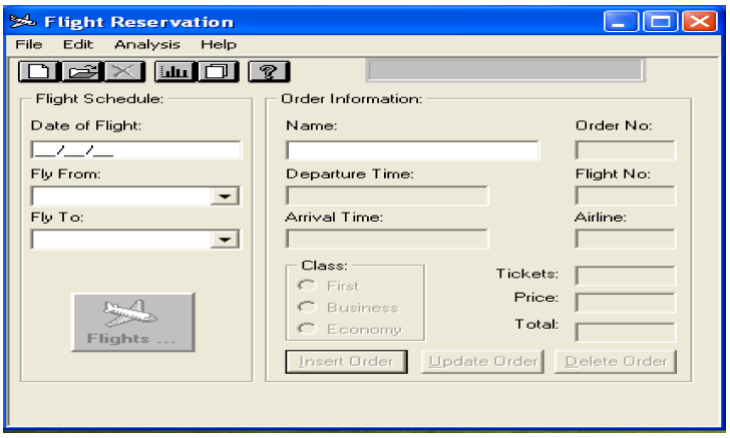
5. Review the results.

Output:

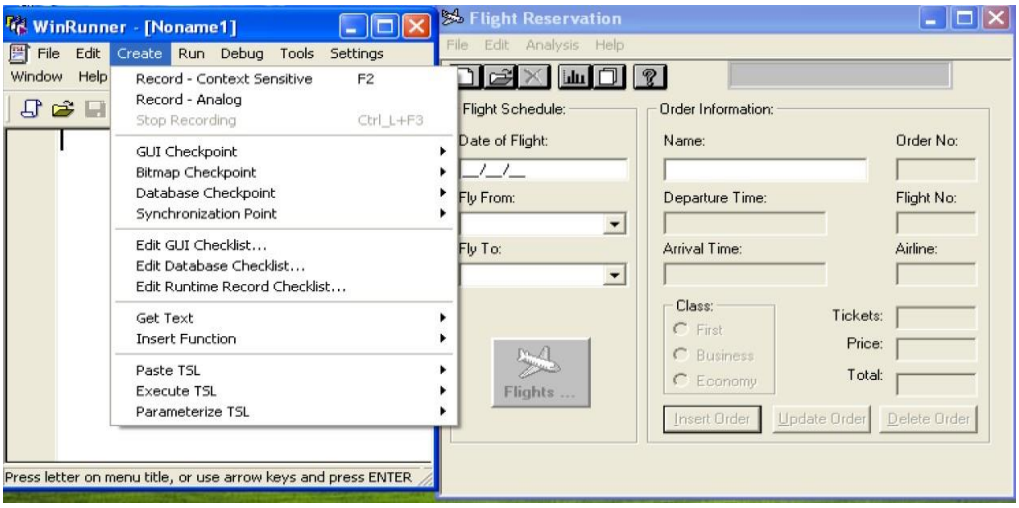
Creating New Test



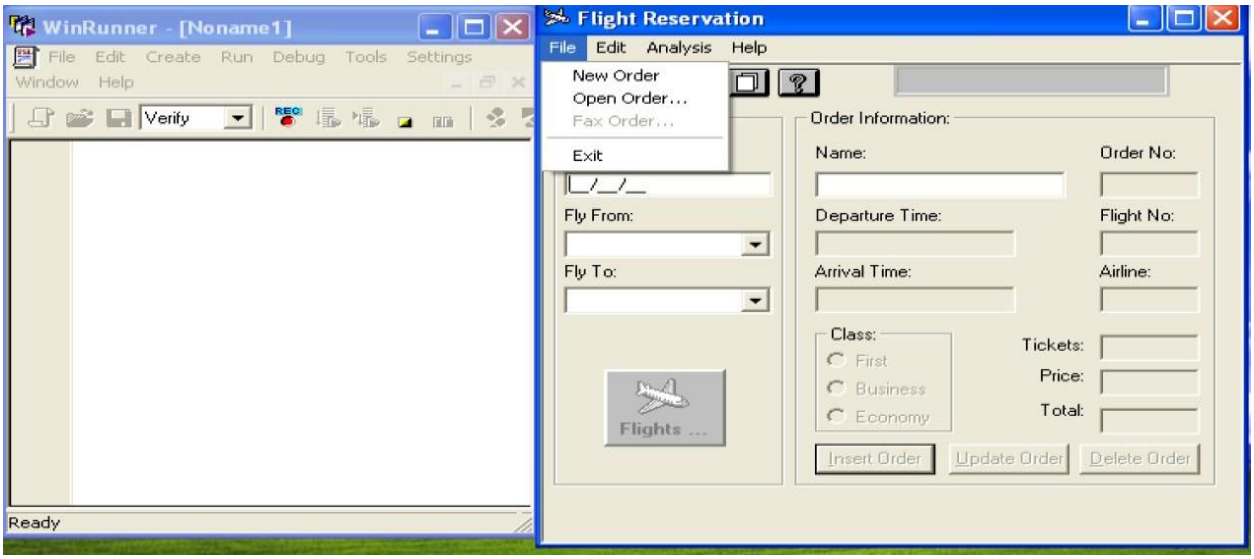
Flight Reservation application started



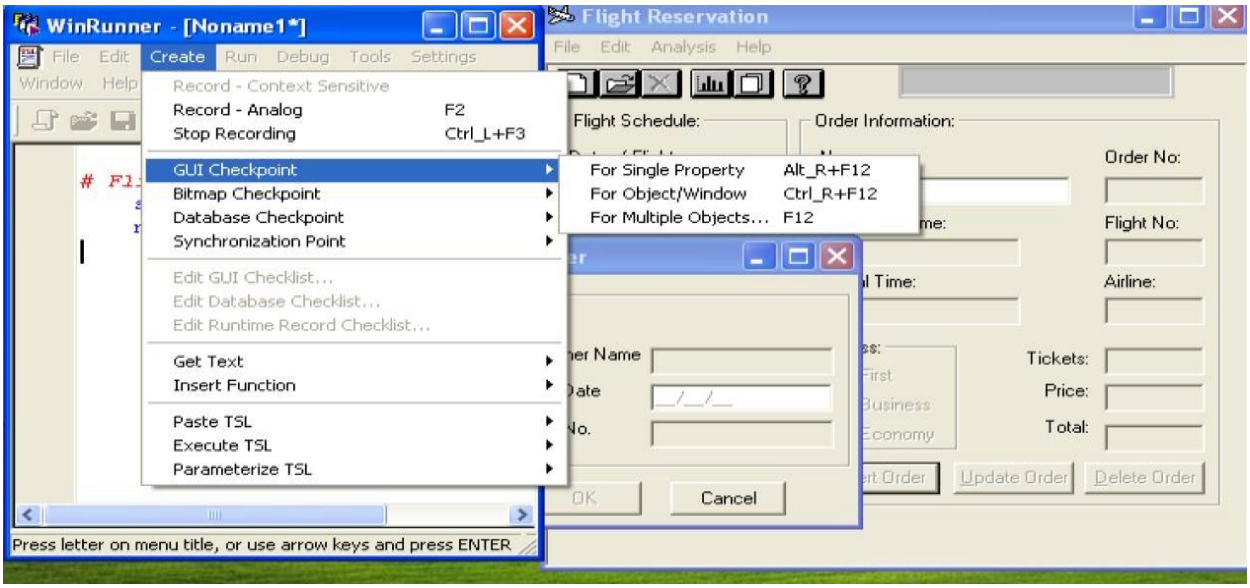
Recording in Context Sensitive Mode



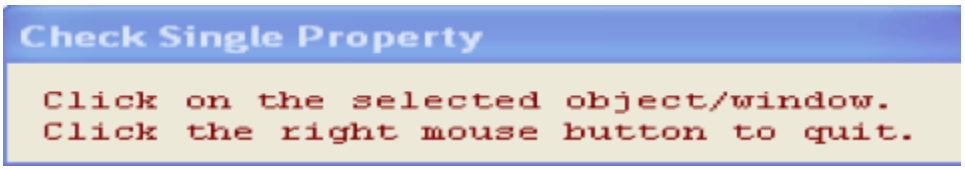
Opening an order in Flight Reservation application



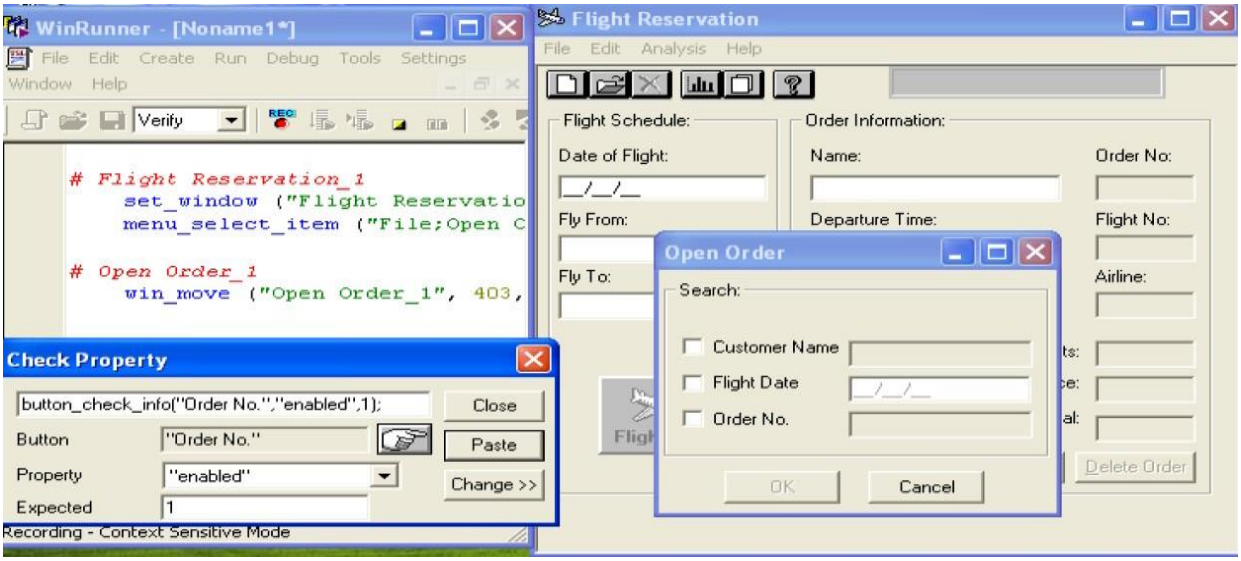
Creating an GUI Checkpoint for Single Property



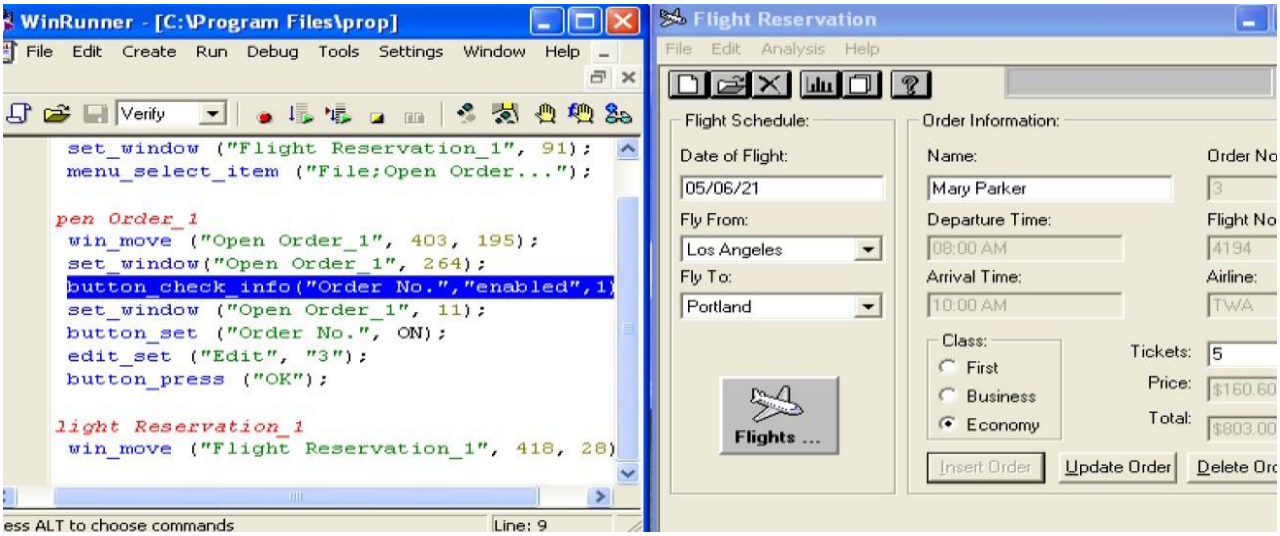
Check Property of an object /window is displayed



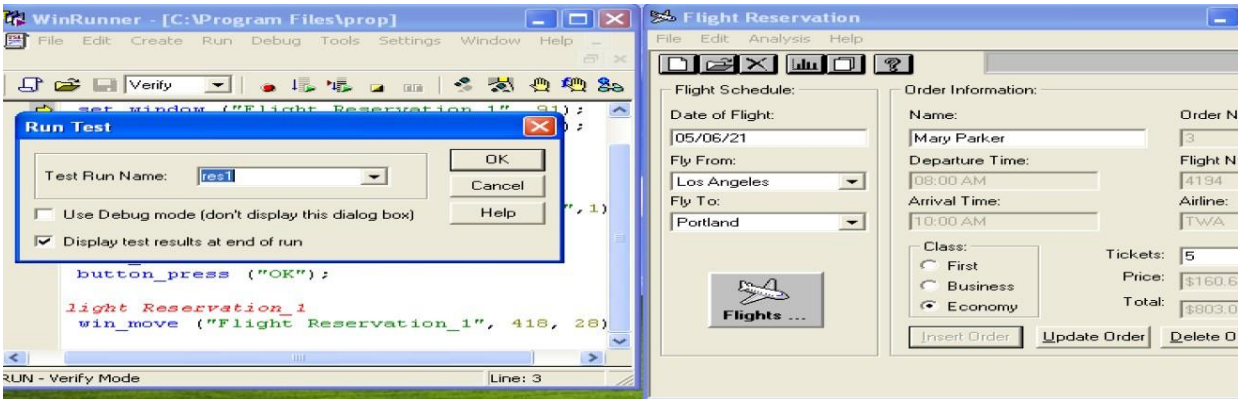
Order No check box property is examined and select paste to insert the details in test script



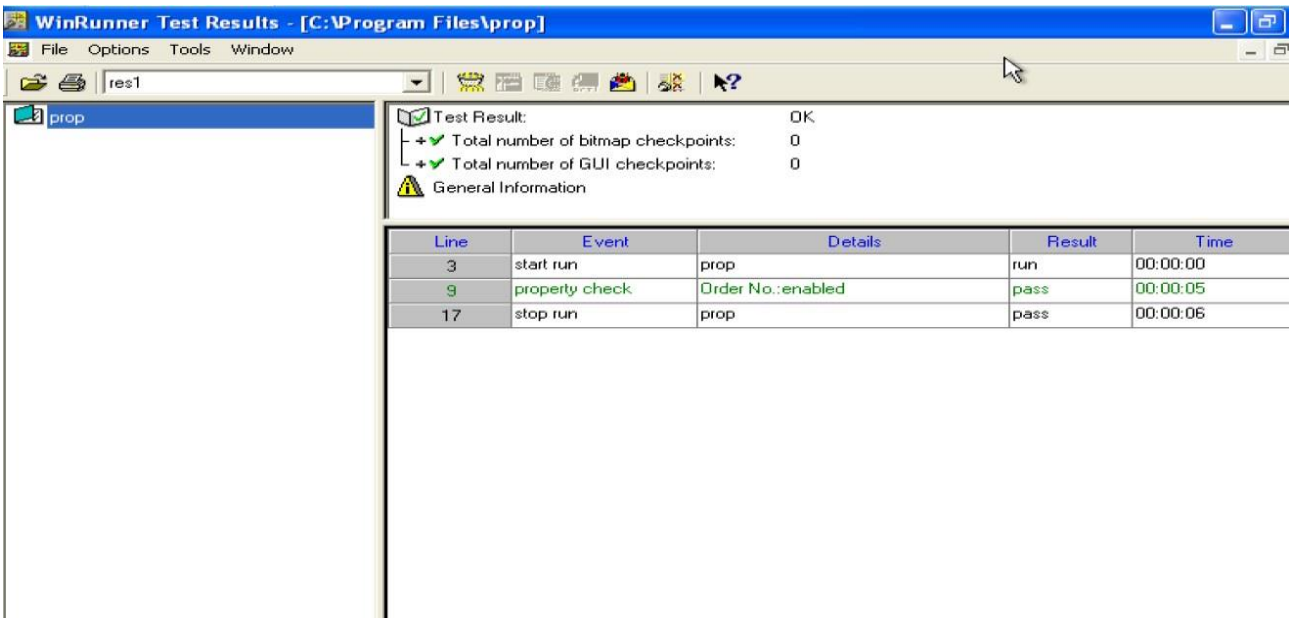
GUI checkpoint for Order No Check box property is created in test script

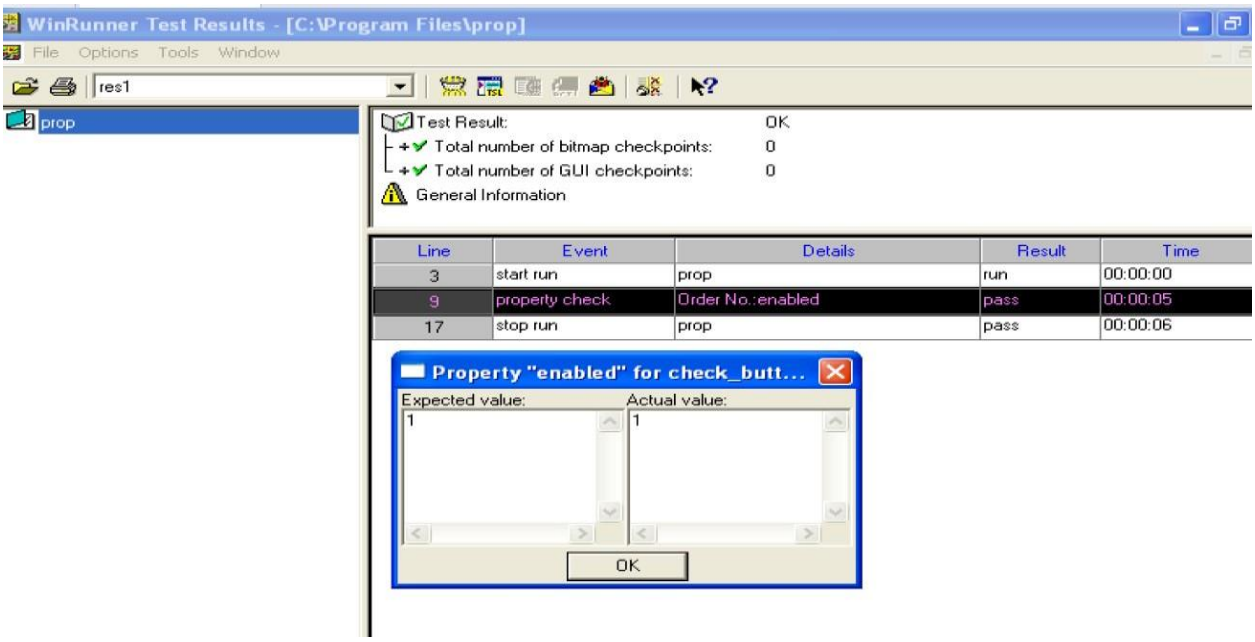


Running the Saved Test



Test Result





Result : GUI Checkpoint for Single property of an object is created and verified successfully

Experiment-3: GUI Checkpoint for Single Object/Window

Aim: To create GUI checkpoint for Single Object/Window

Theory:

Behavior of Graphical User Interface objects in an application are verified by GUI checkpoints using WinRunner tool.**A GUI checkpoint** examines the behavior of an object's properties

Here we create GUI checkpoint for Single Object

Procedure:

Steps:

1. Start WinRunner and create a new test.

2 Start the Flight Reservation application and log in.

3 Start recording in Context Sensitive mode.

4 Open the Open Order dialog box from flight reservation application

5 Create a GUI checkpoint for the Order No. check box.

Choose Create > GUI Checkpoint>For Single Object/Window

6. Use the pointer to double-click the Order No. check box.

The Check GUI dialog box opens and displays the available checks.

Accept the default check "State."

This check captures the current state (off) of the check box and stores it as expected results.

Click OK in the Check GUI dialog box to insert the checkpoint into the test script.

The checkpoint appears as an obj_check_gui statement

7. Enter "4" in the Order No. text box.

8. Click OK & Stop recording

9. Save the test.

To Run the Test

1. Make sure the above saved test and Flight Reservation application are open.

2 In WinRunner, check that Verify mode is selected in the Standard toolbar.

3.Choose Run > Run from Top,

4. The Run Test dialog box opens. Accept the default test run name "res1."

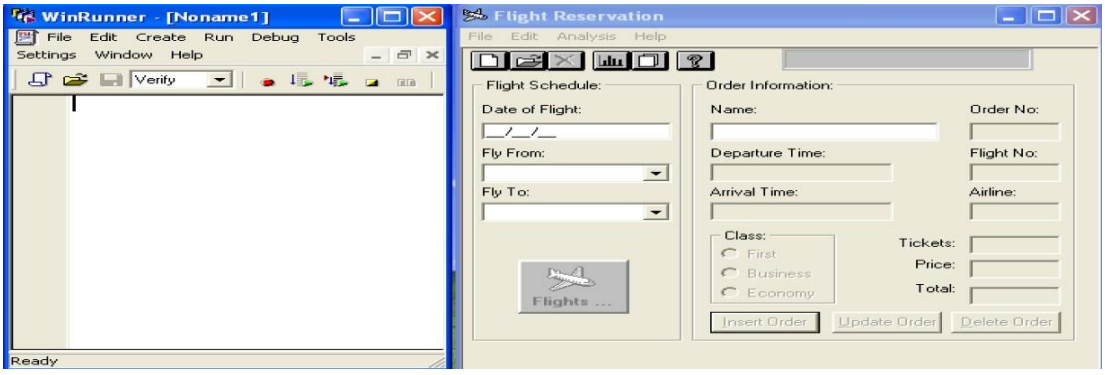
Run the test Click OK in the Run Test dialog box.

5. Review the results.

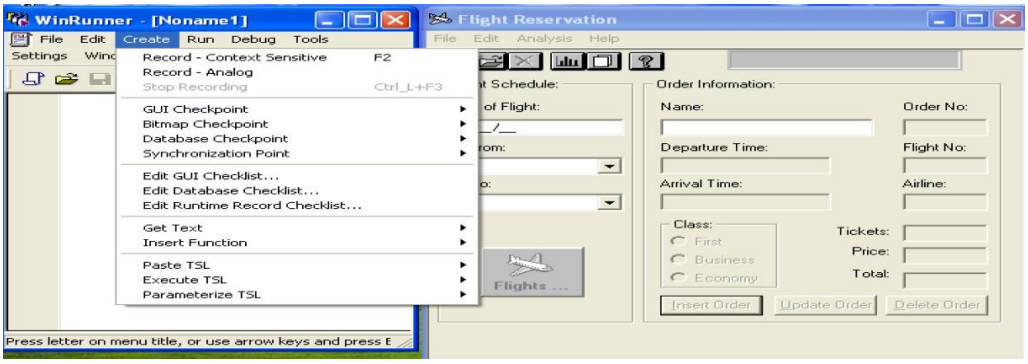
Output:

Creating GUI Checkpoint for Single Object/Window

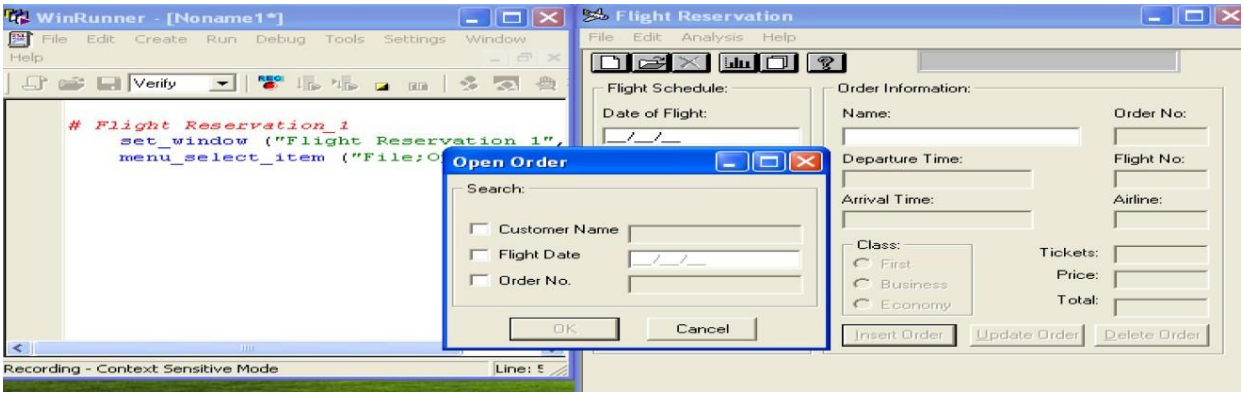
Open WinRunner tool & Application



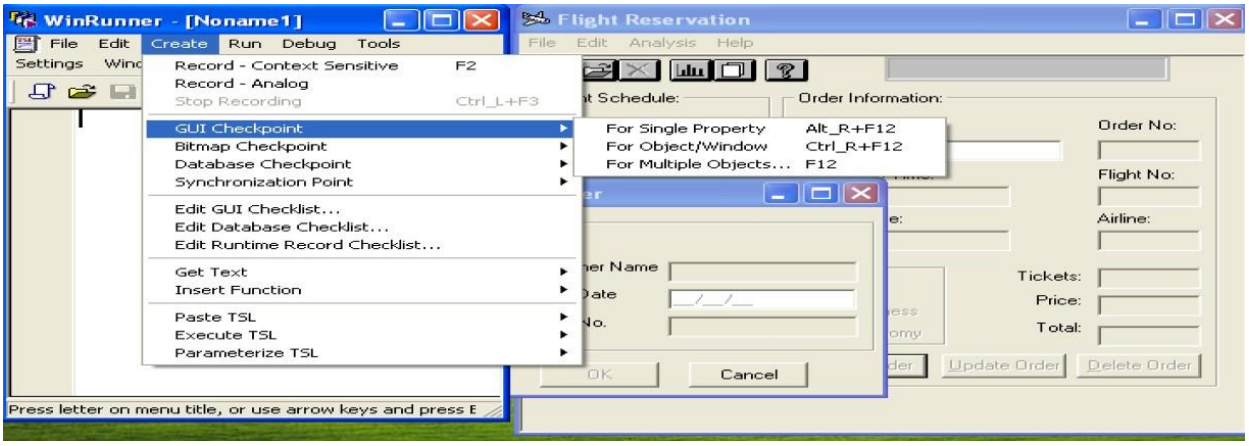
Recording in Context Sensitive Mode

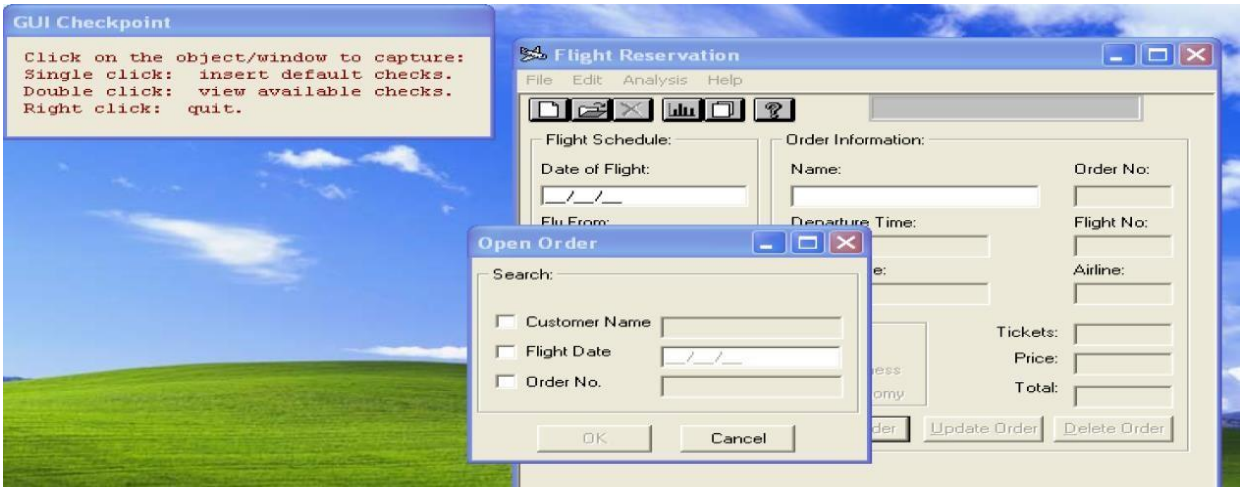


Opening Order from Flight Reservation Application

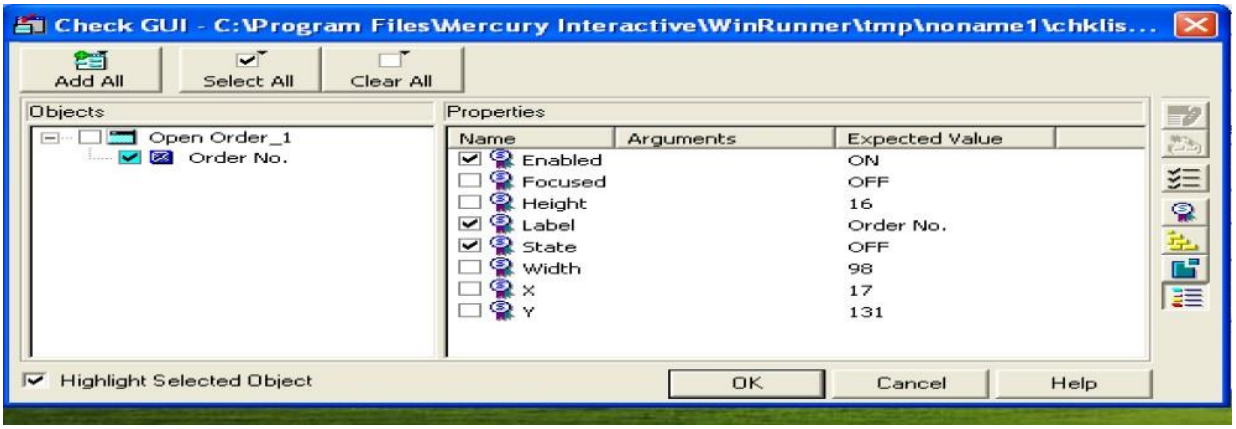


Creating a GUI Checkpoint for Object/Window for Order No checkbox object

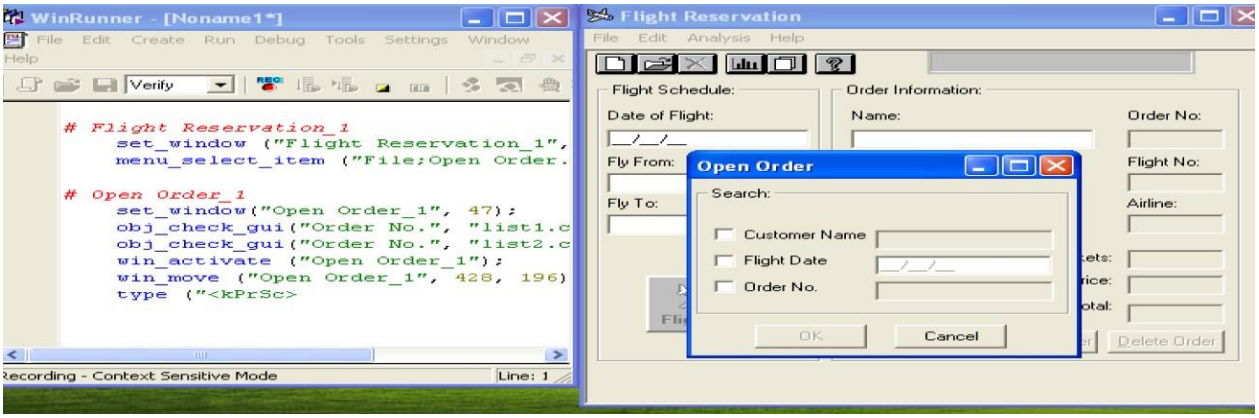


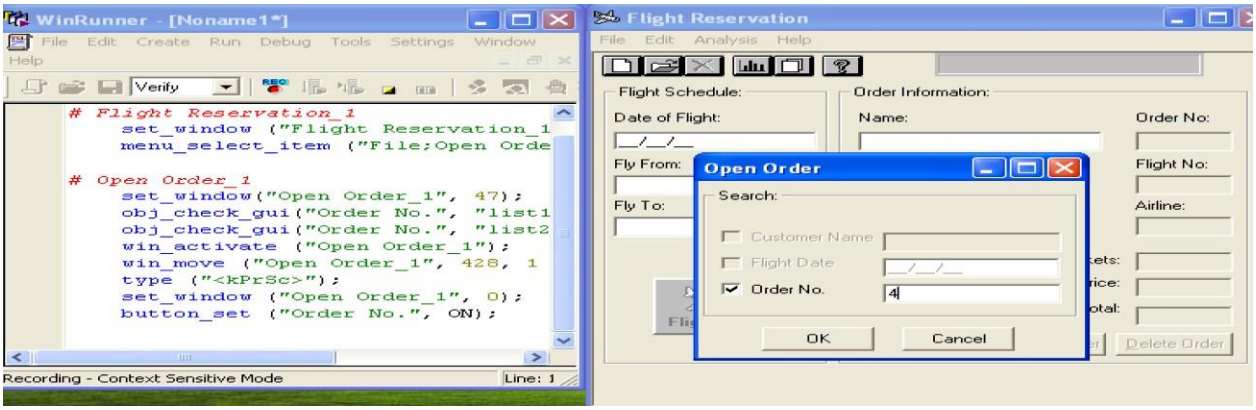


Check GUI dialog box opens on double clicking the Order No check box

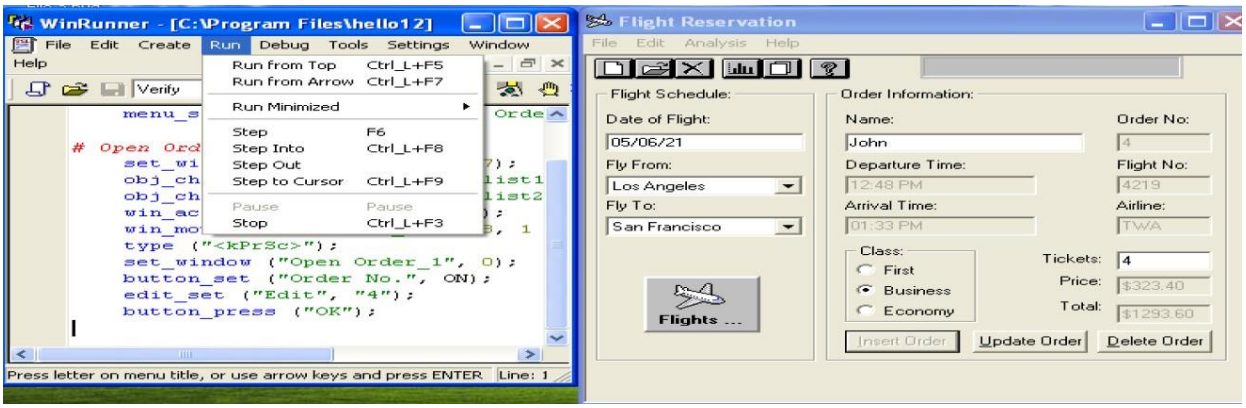


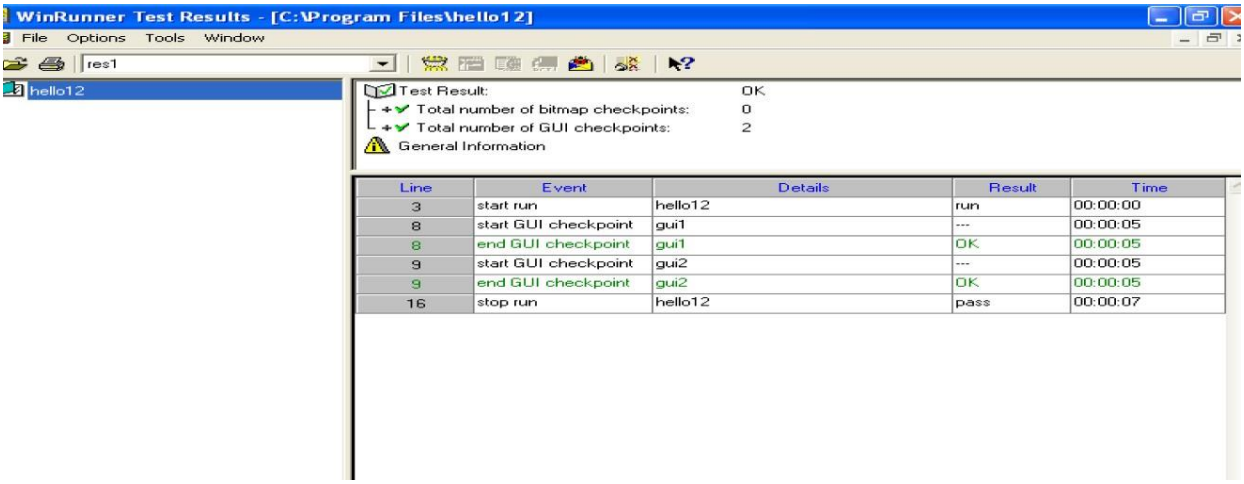
On clicking OK, GUI checkpoint is created and inserted in test script

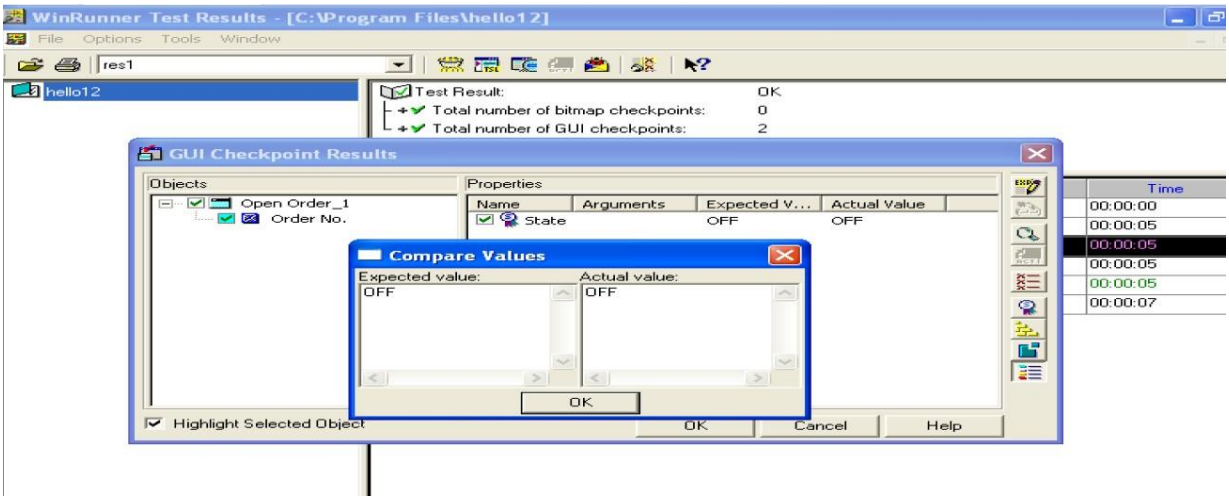


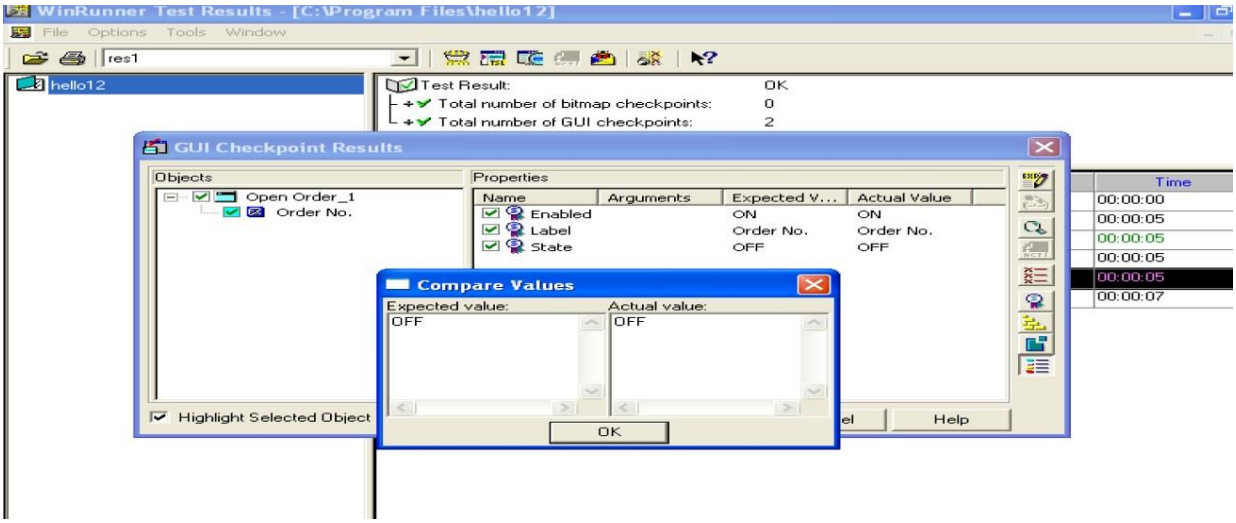


Running the Test




WinRunner Test Results - [C:\Program Files\hello12]
File Options Tools Window
res1
hello12
Test Result: OK
Total number of bitmap checkpoints: 0
Total number of GUI checkpoints: 2
General Information
Line Event Details Result Time
3 start run hello12 run 00:00:00
8 start GUI checkpoint gui1 --- 00:00:05
8 end GUI checkpoint gui1 OK 00:00:05
9 start GUI checkpoint gui2 --- 00:00:05
9 end GUI checkpoint gui2 OK 00:00:05
16 stop run hello12 pass 00:00:07


WinRunner Test Results - [C:\Program Files\hello12]
File Options Tools Window
res1
hello12
Test Result: OK
Total number of bitmap checkpoints: 0
Total number of GUI checkpoints: 2
GUI Checkpoint Results
Objects
Open Order_1
Order No.
Properties
Name Arguments Expected V... Actual Value
State OFF OFF
Compare Values
Expected value:
OFF
Actual value:
OFF
OK
Highlight Selected Object
OK
Cancel
Help
Time
00:00:00
00:00:05
00:00:05
00:00:05
00:00:05
00:00:07


WinRunner Test Results - [C:\Program Files\hello12]
File Options Tools Window
res1
hello12
Test Result: OK
Total number of bitmap checkpoints: 0
Total number of GUI checkpoints: 2
GUI Checkpoint Results
Objects
Open Order_1
Order No.
Properties
Name Arguments Expected V... Actual Value
Enabled ON ON
Label Order No. Order No.
State OFF OFF
Highlight Selected Object
Help
Compare Values
Expected value:
OFF
Actual value:
OFF
OK
Time
00:00:00
00:00:05
00:00:05
00:00:05
00:00:05
00:00:07

Experiment-4: GUI Checkpoint for Multiple Objects

Aim: To create GUI checkpoint for Single Object/Window

Theory:

Behaviour of Graphical User Interface objects in an application are verified by GUI checkpoints using WinRunner tool. **A GUI checkpoint** examines the behavior of an object's properties

Here we create GUI checkpoint for Multiple Objects

Procedure:

Steps:

1. Start WinRunner and create a new test.

2 Start the Flight Reservation application and log in.

3 Start recording in Context Sensitive mode.

4 Open the Open Order dialog box from flight reservation application

5 Create a GUI checkpoint for the Order No. check box.

Choose Create > GUI Checkpoint>For Multiple Objects

6. Click on Add button to add multiple objects for checkpoint

Click on the Open Order Window to add multiple objects.

Select the Objects and properties in GUI Checkpoint window and click OK

7. Enter "4" in the Order No. text box.

8. Click OK & Stop recording

9. Save the test.

To Run the Test

1. Make sure the above saved test and Flight Reservation application are open.

2 In WinRunner, check that Verify mode is selected in the Standard toolbar.

3.Choose Run > Run from Top,

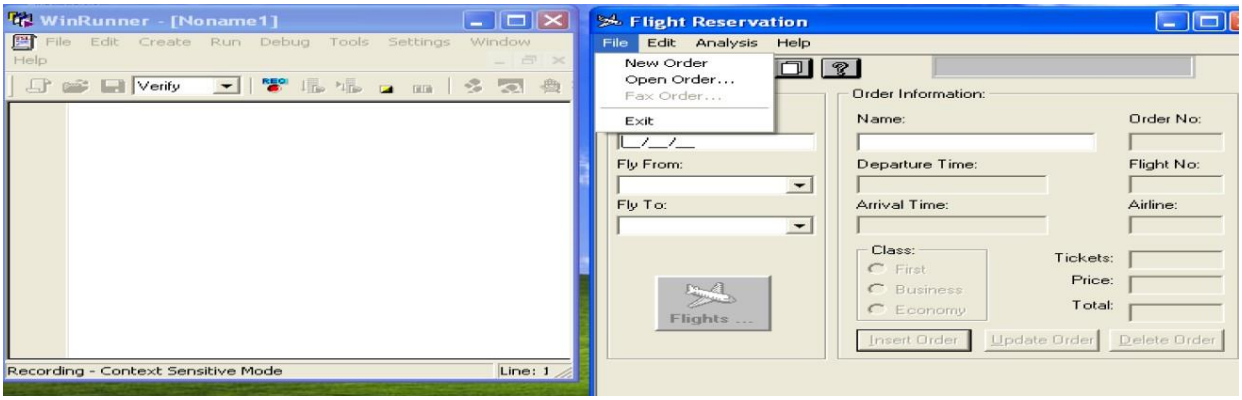
4. The Run Test dialog box opens. Accept the default test run name "res1."

Run the test Click OK in the Run Test dialog box.

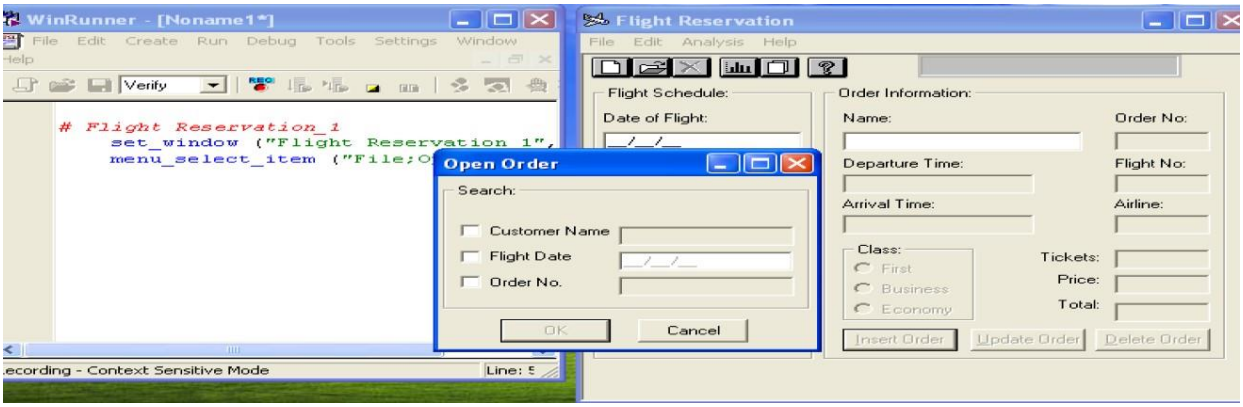
5. Review the results.

Output:

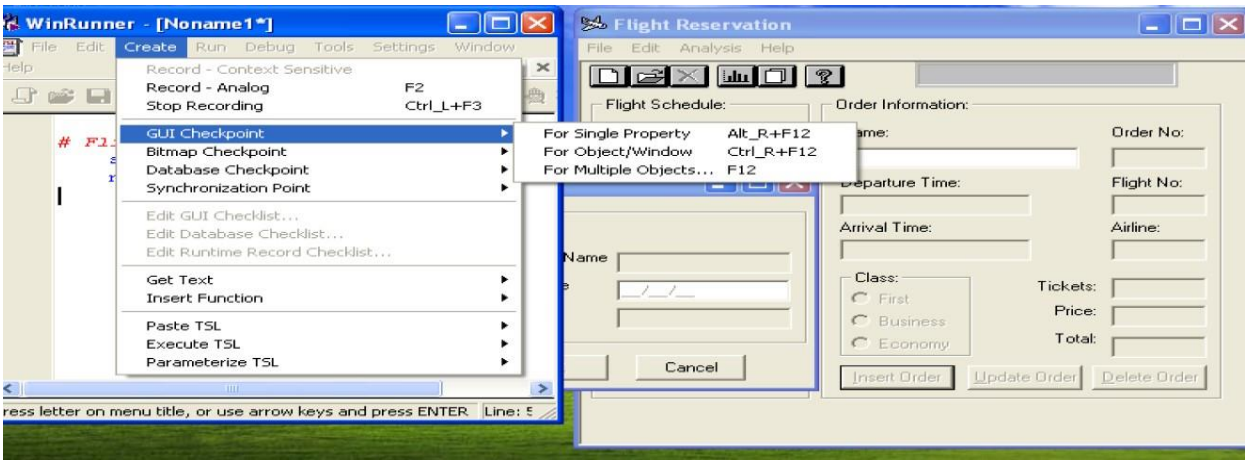
Opening Order Window



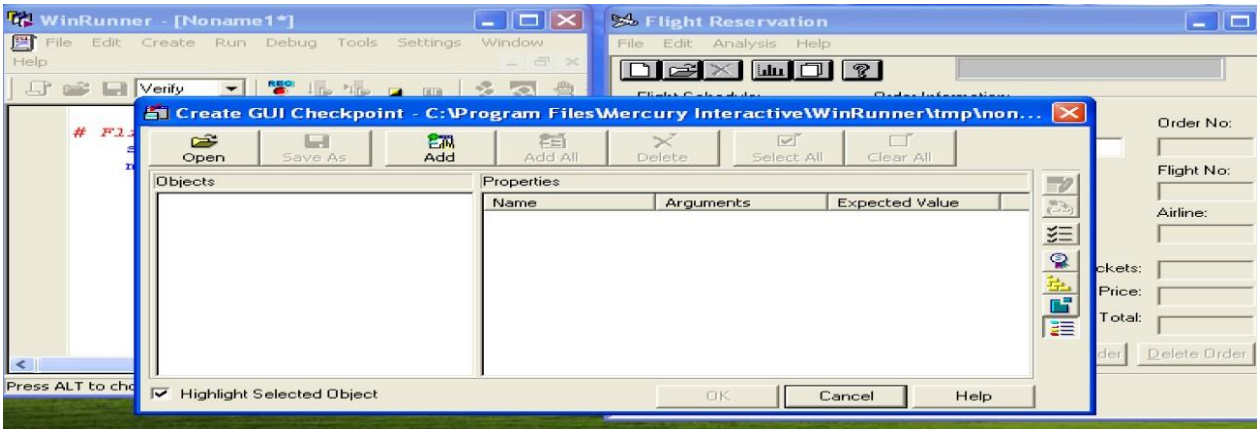
Recording the Test



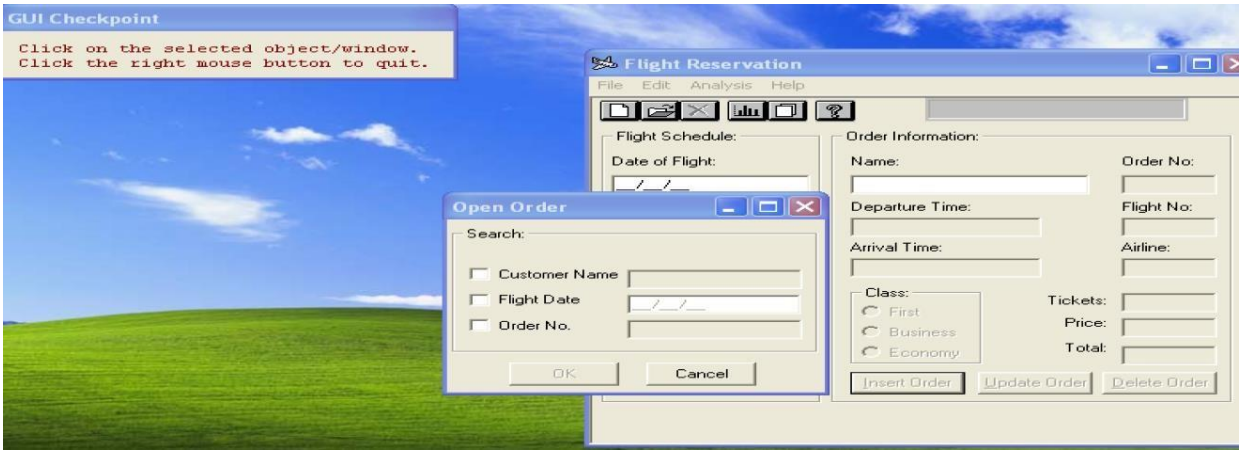
Creating GUI Checkpoints for Multiple Objects



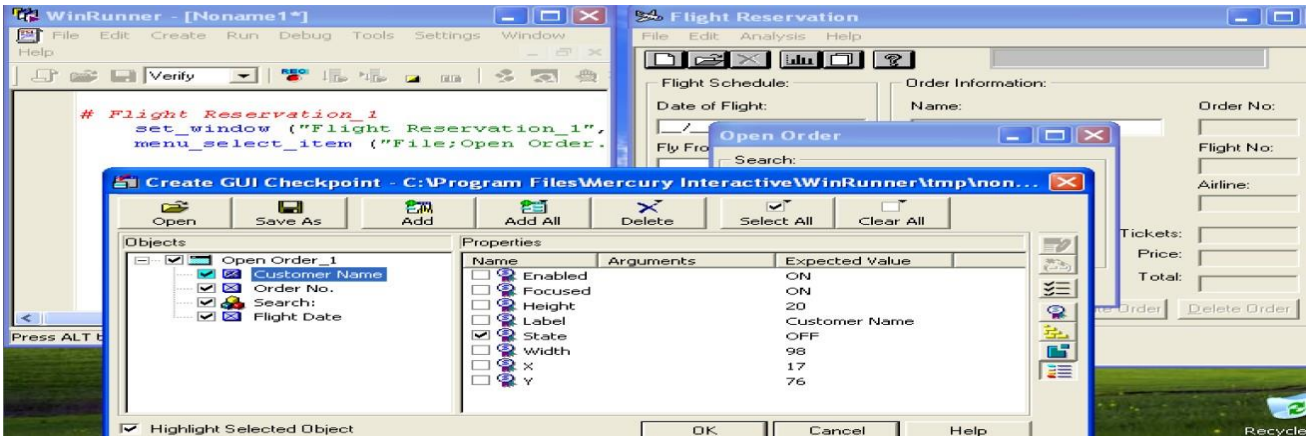
Click on Add button to add multiple objects for checkpoint



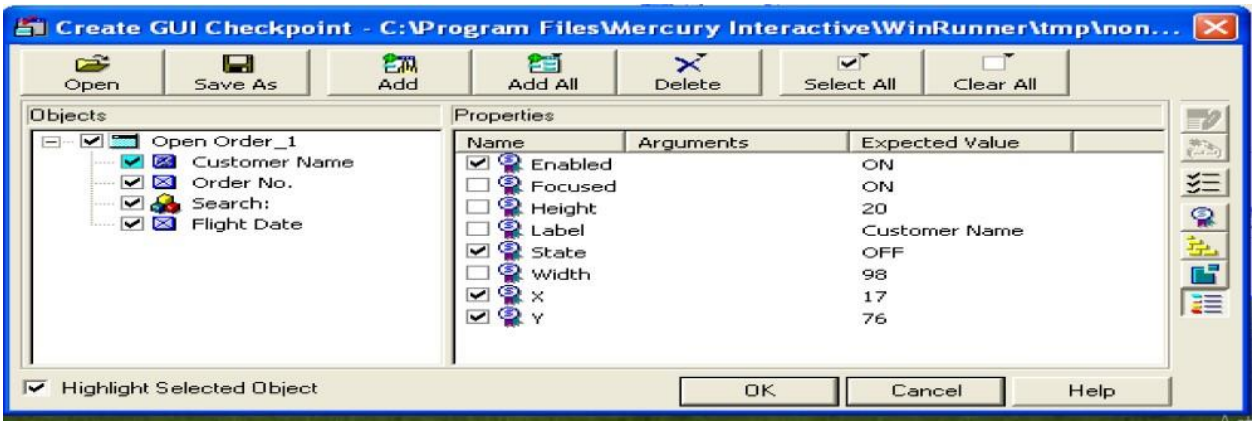
Select Window- Open order



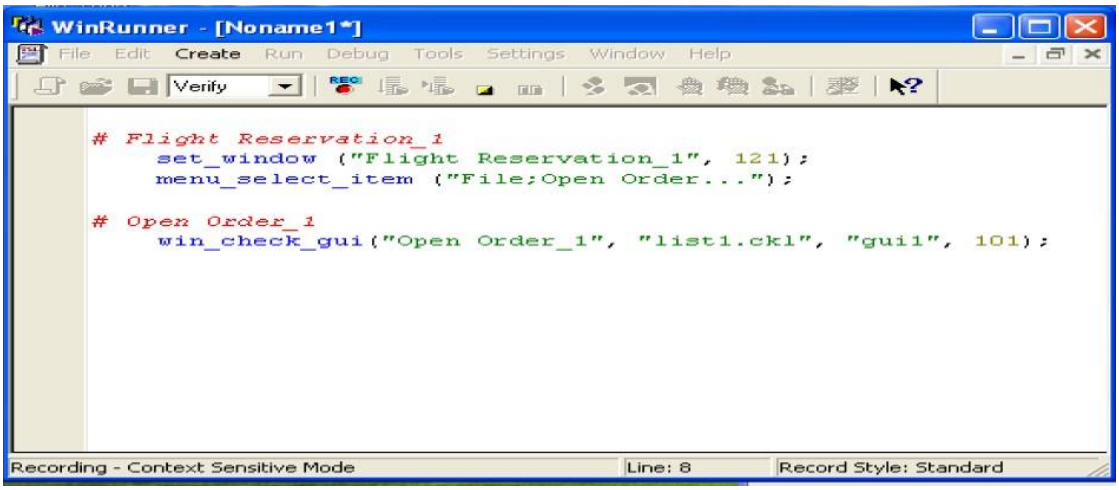
Select Objects & Properties for Checkpoint



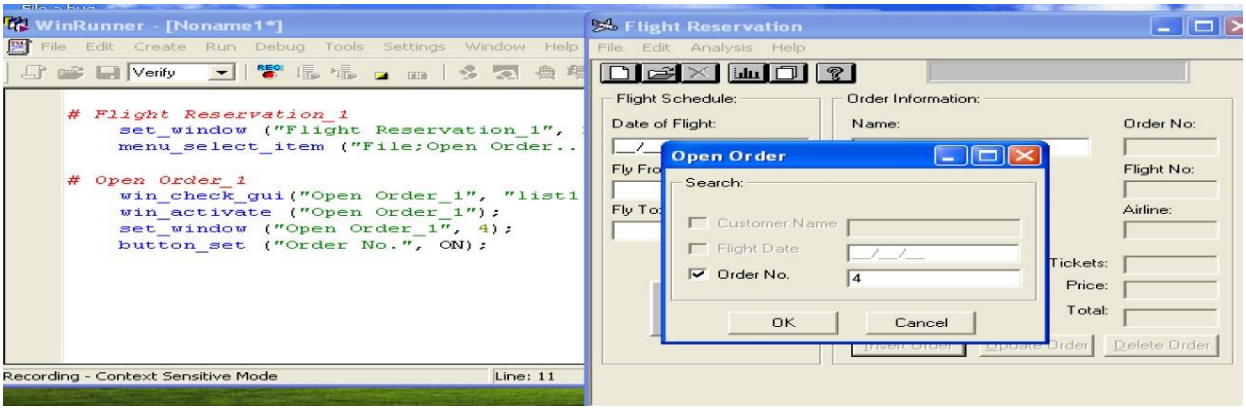
Click Ok



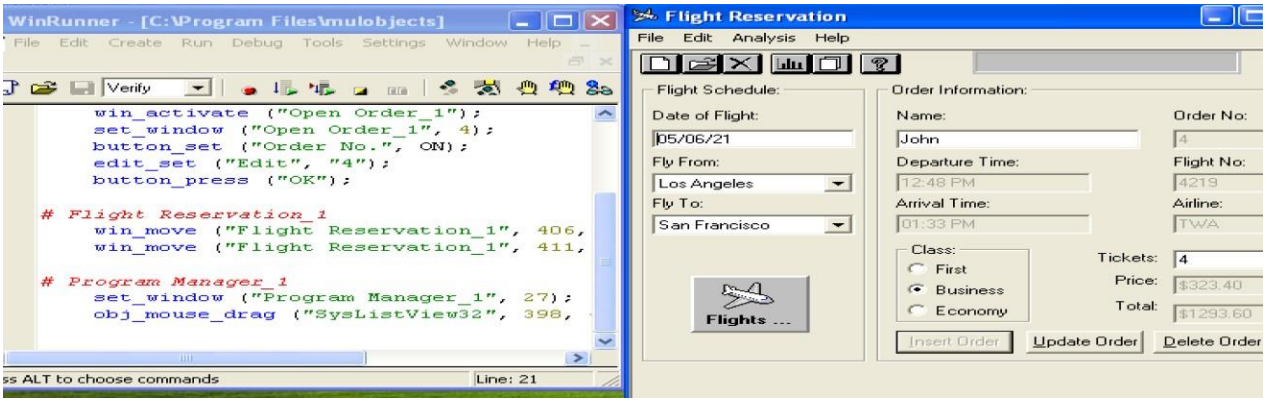
Window check point is created



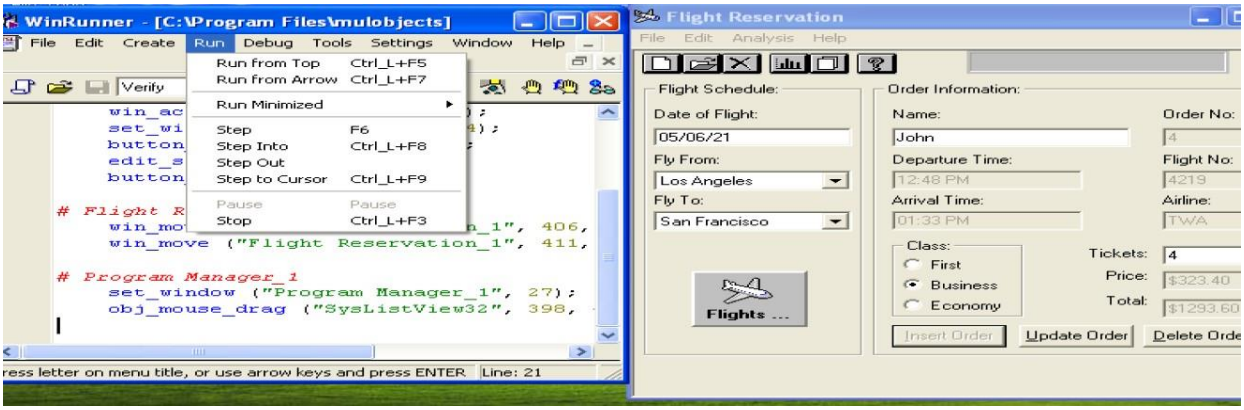
Opening Order 4



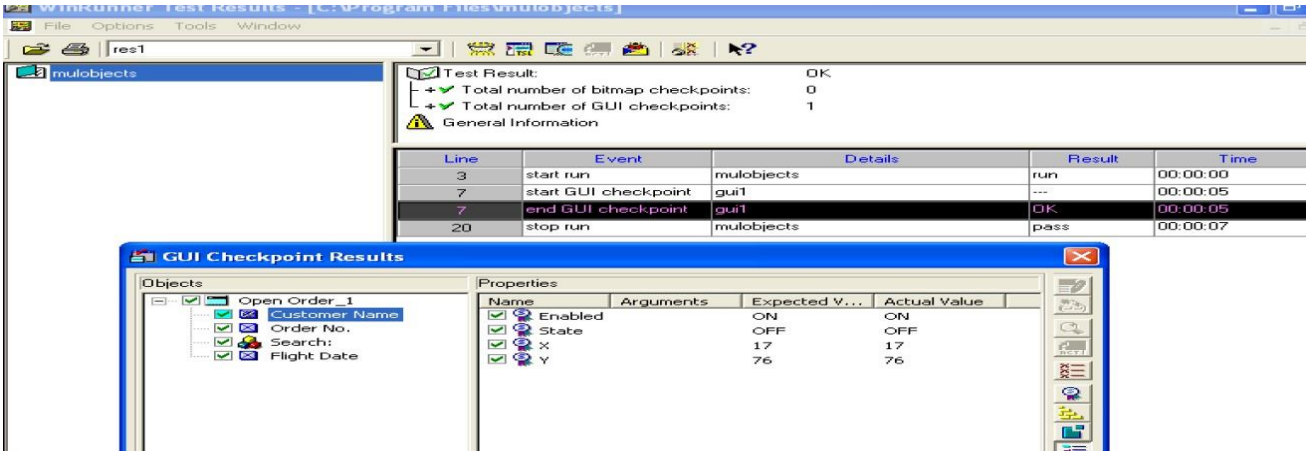
Stop recording and Saving test



Running the Test



Test Result



Result: GUI Checkpoint for Multiple Objects have been created successfully

Experiment-5:

a) Bitmap Checkpoint for Object/Window

Aim: To create Bitmap checkpoint for Object/Window

Theory:

If an application contains bitmap areas, such as drawings or graphs, we can check these areas using a bitmap checkpoint. A bitmap checkpoint compares captured bitmap images pixel by pixel.

Procedure:

Here we will test the Agent Signature box in the Fax Order dialog box. We will use a bitmap checkpoint to check that we can sign name in the box. Then we will use another bitmap checkpoint to check that the box clears when we click the Clear Signature button.

To create Bitmap checkpoint
Choose Create > Bitmap Checkpoint>For Object/Window or for Screen Area

Steps:

1 Start WinRunner and open a new test.

2 Start the Flight Reservation application and log in.

3 Start recording in Context Sensitive mode.

4 Open order #6.

5 Open the Fax Order dialog box.

6 Enter a 10-digit fax number in the Fax Number box.

7 Move the Fax Order dialog box.

8 Switch to Analog mode.

9 Sign your name in the Agent Signature box.

10 Switch back to Context Sensitive mode.

11 Insert a bitmap checkpoint that checks your signature.

12 Click the Clear Signature button.

13 Insert another bitmap checkpoint that checks the Agent Signature box.

14 Click the Cancel button on the Fax Order dialog box.

15 Stop recording.

16 Save the test.

Run the Test on new version of Application

1. Make sure the above saved test is open, open flight_1B application & login

2 In WinRunner, check that Verify mode is selected in the Standard toolbar.

3.Choose Run > Run from Top,

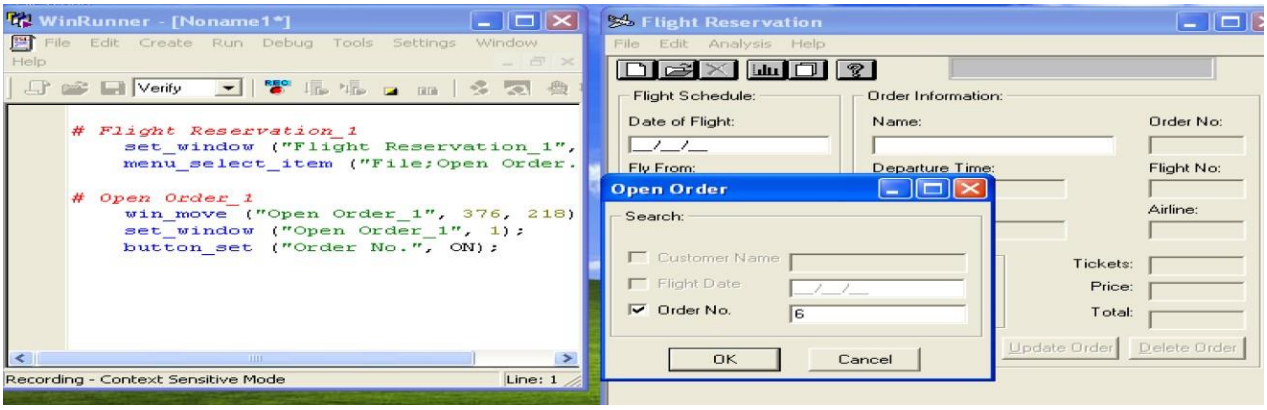
4. The Run Test dialog box opens. Accept the default test run name “res1.”

Run the test Click OK in the Run Test dialog box.

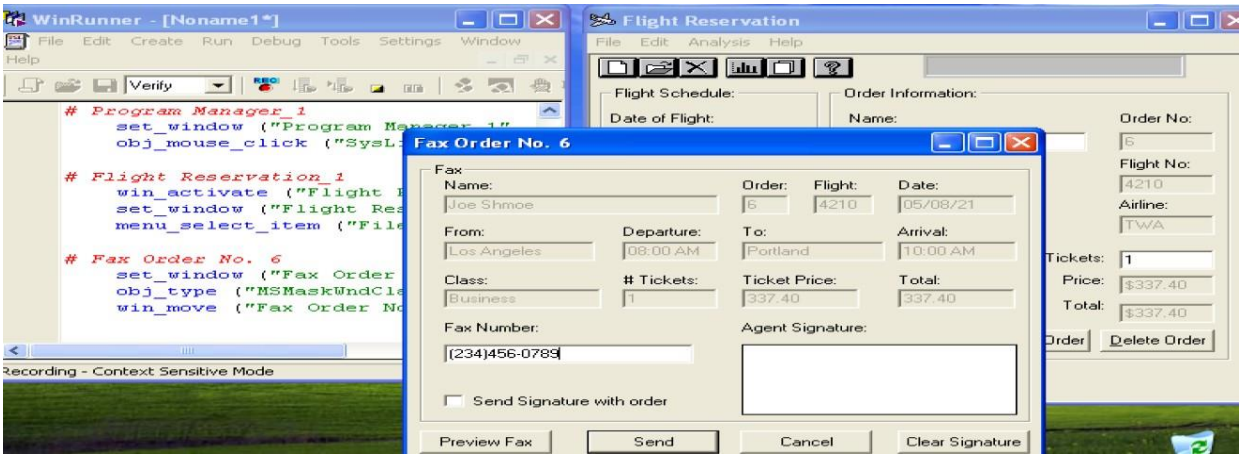
5. Review the results.

Output:

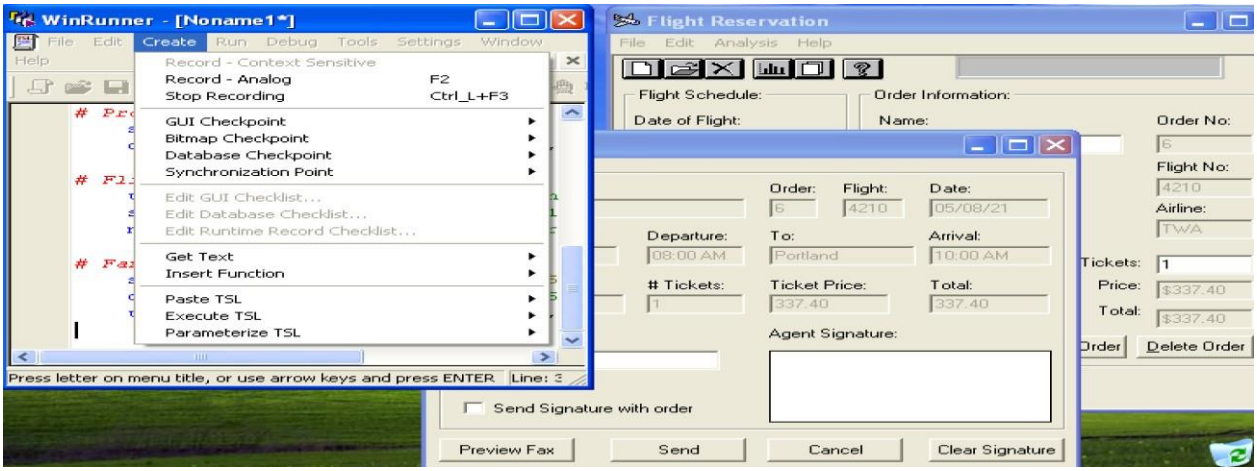
Opening order 6 in Flight Reservation window



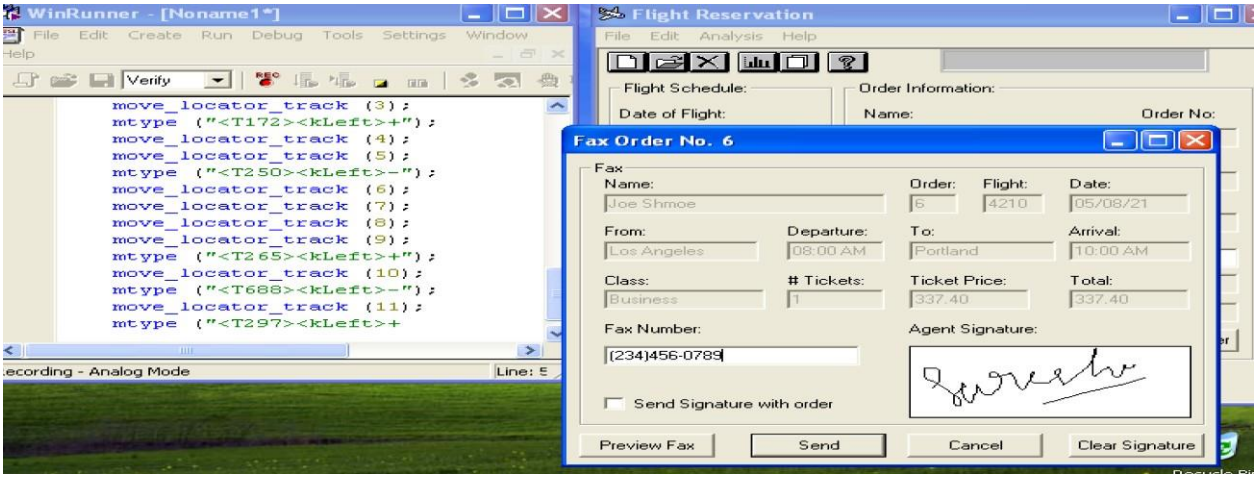
Open Fax Order Dialog box, enter fax number, move the fax order dialog box



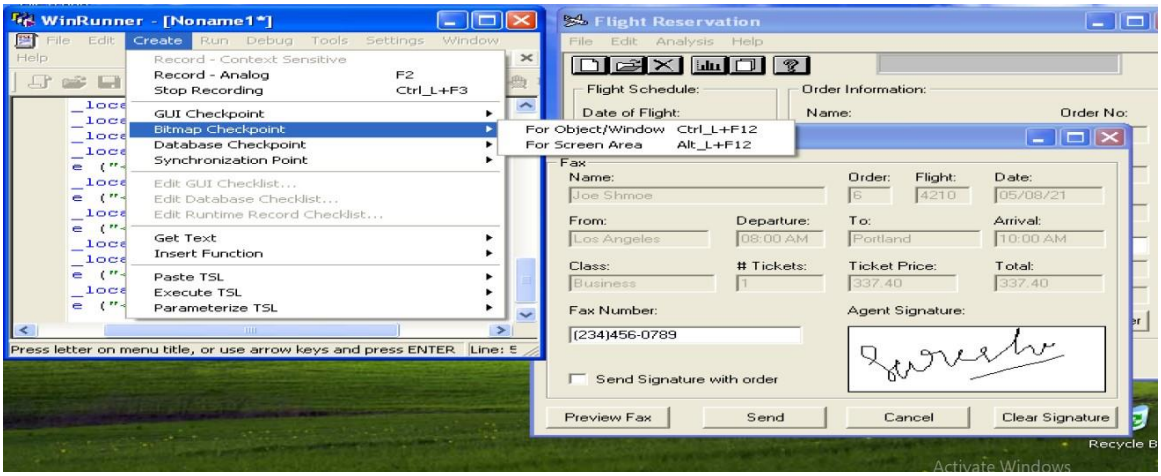
Switching to Analog Mode



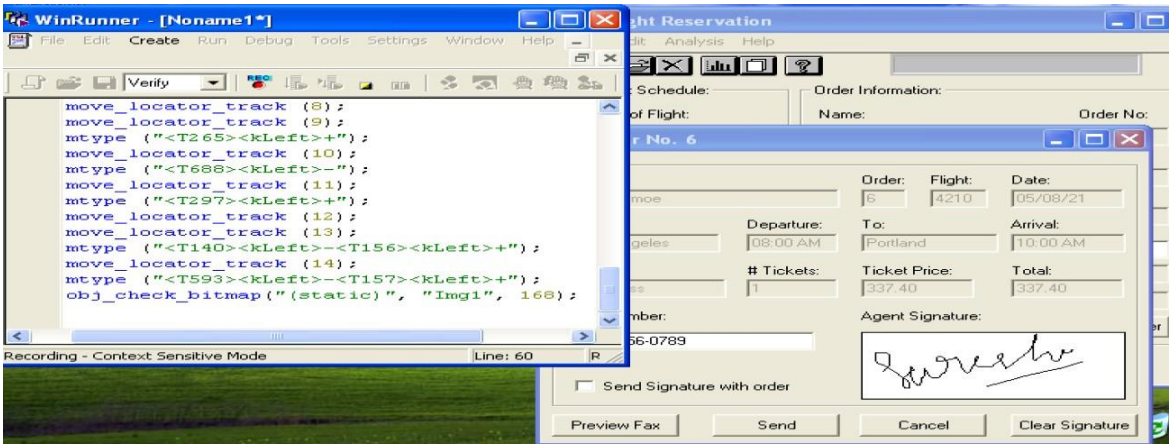
Sign in the Agent Signature Box



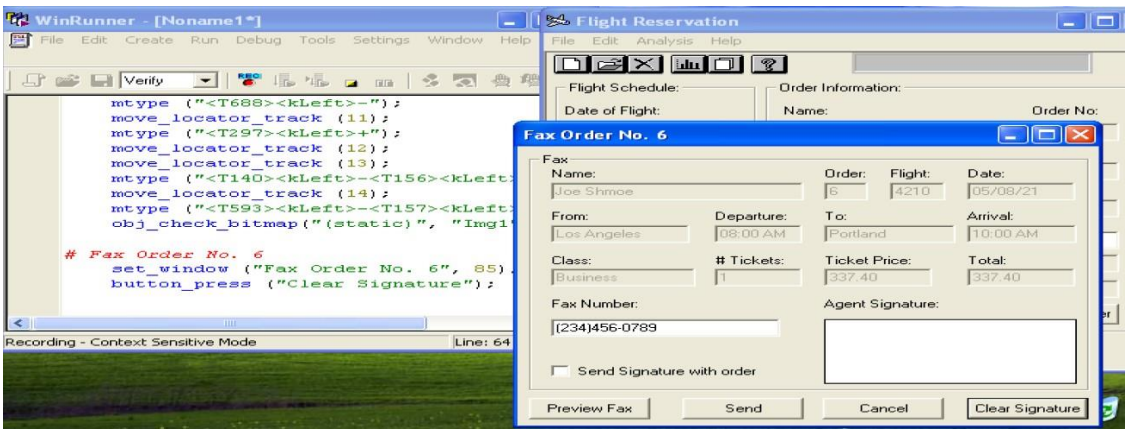
Switching back to Context Sensitive Mode



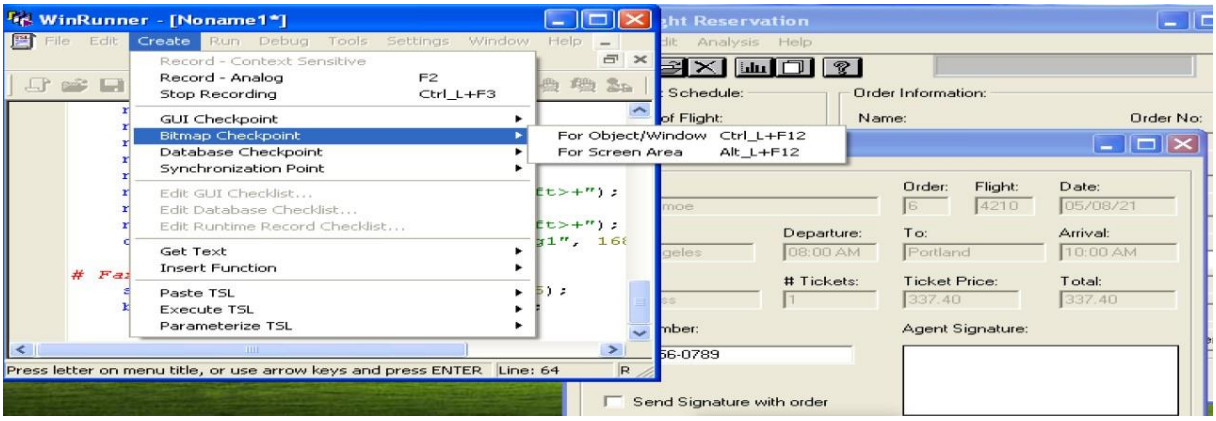
Insert a Bitmap checkpoint that checks the signature



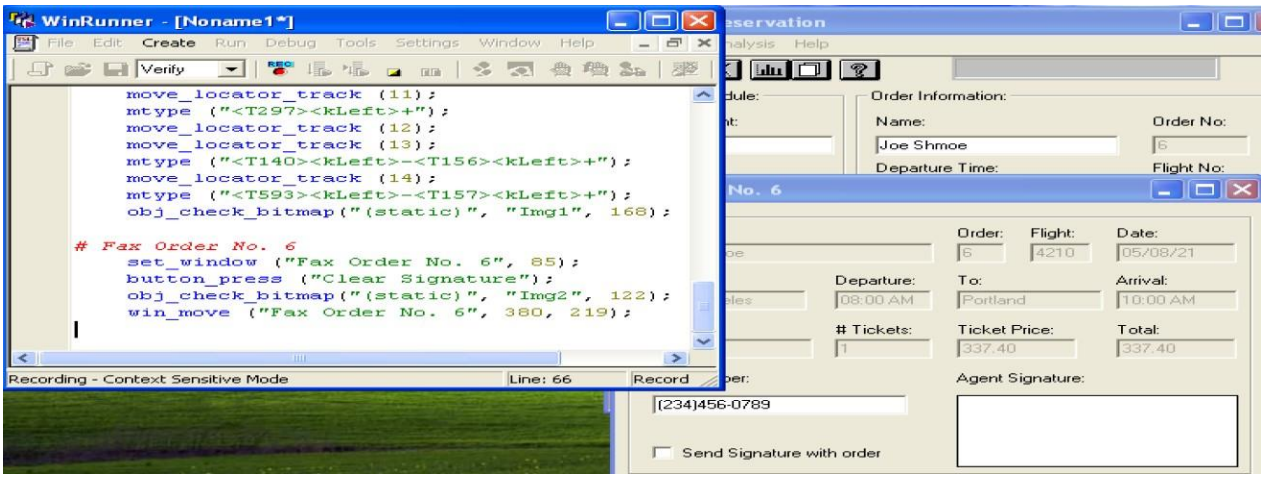
Clear the signature by clicking clear signature button

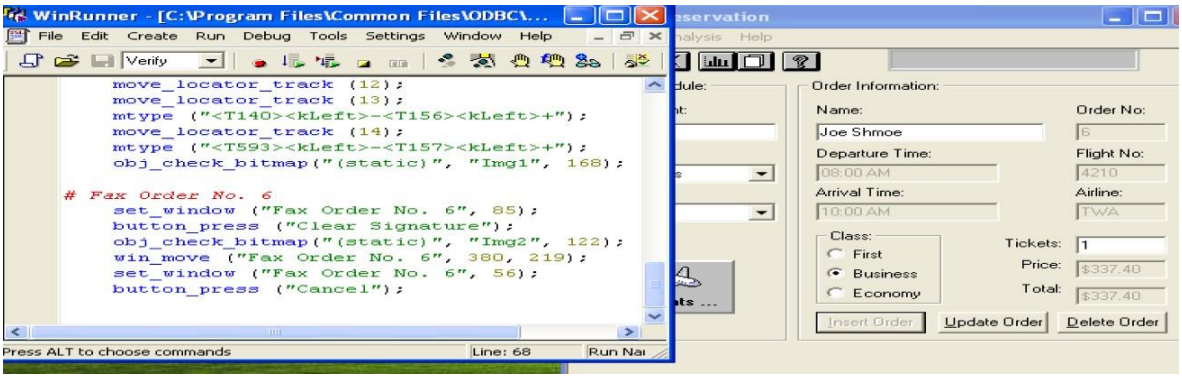


Insert Another Bitmap checkpoint that checks Agent Signature box



Bitmap checkpoints are created for 2 signatures as images

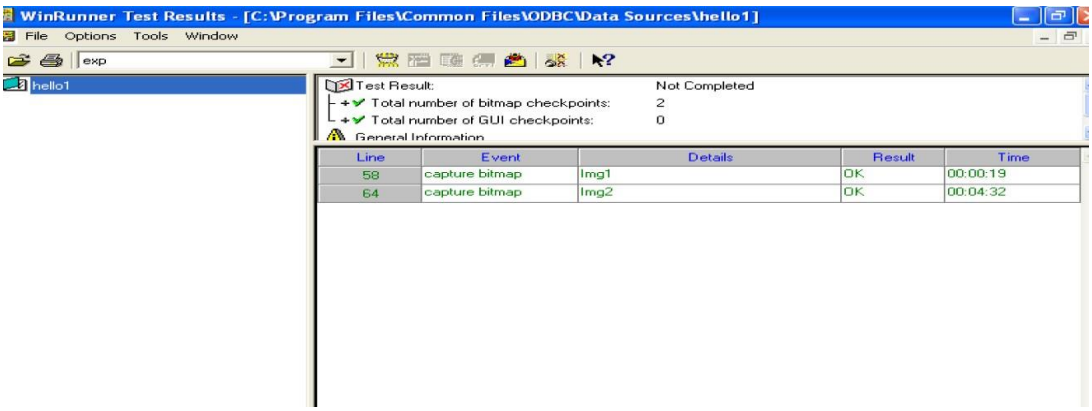




Viewing Expected Results

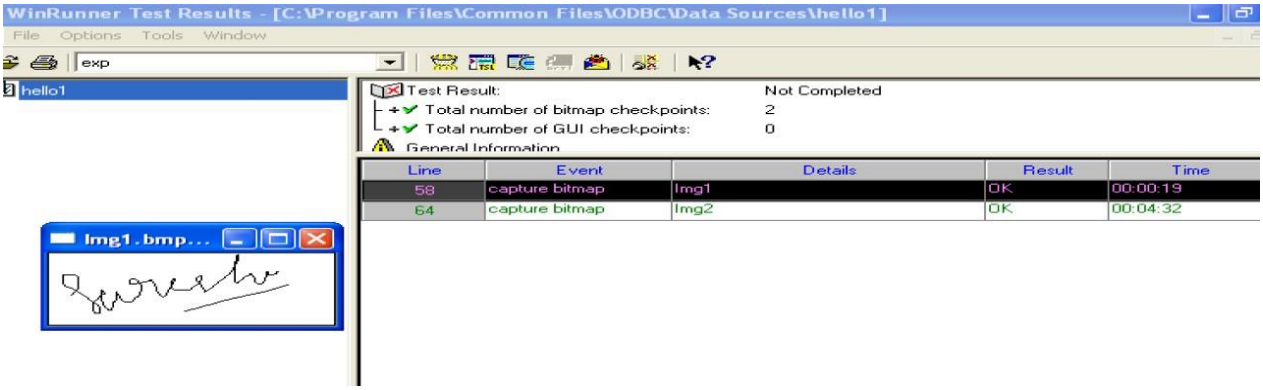
Open the WinRunner Test Results window.

Choose Tools > Test Results or click the Test Results button

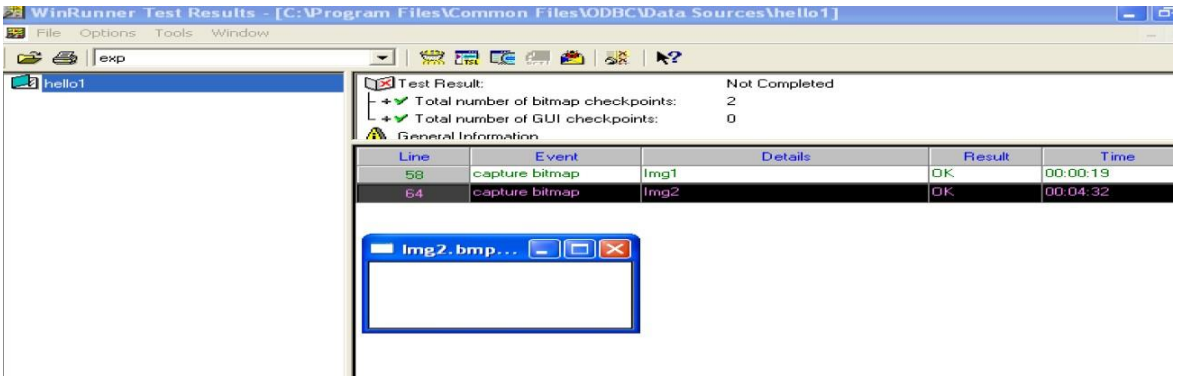


View the captured bitmaps.

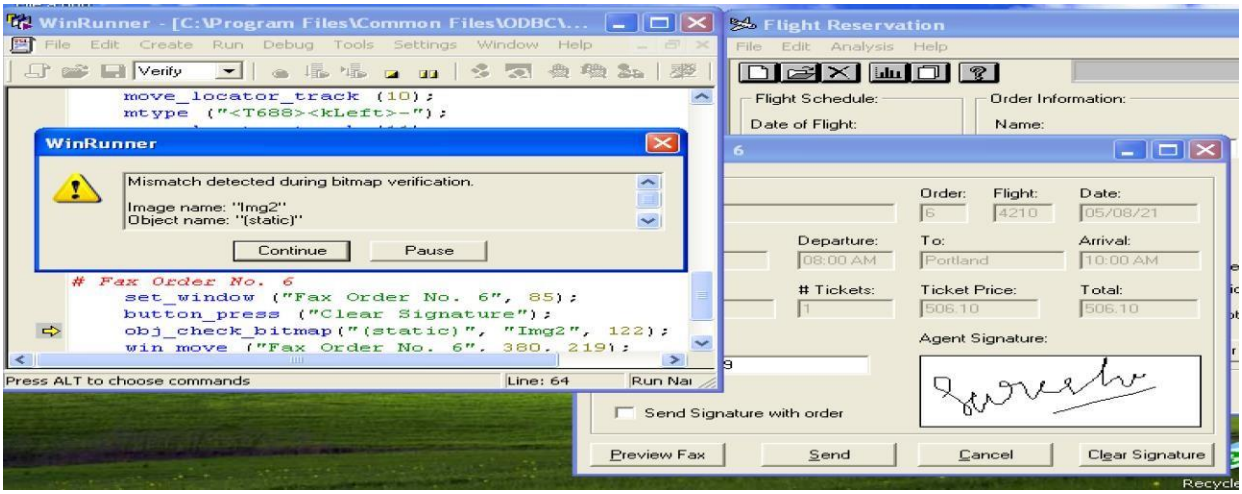
In the test log section, double-click the first “capture bitmap” event, or select it and click the Display button.



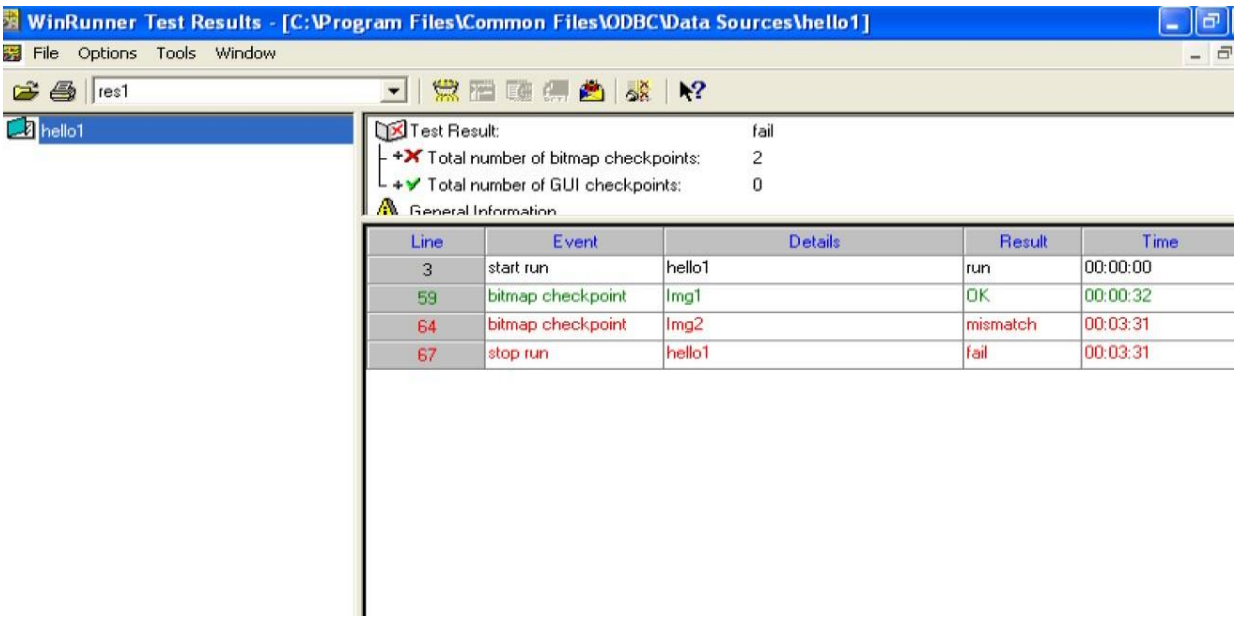
Next, double-click the second “capture bitmap”event, or select it and click the Display button



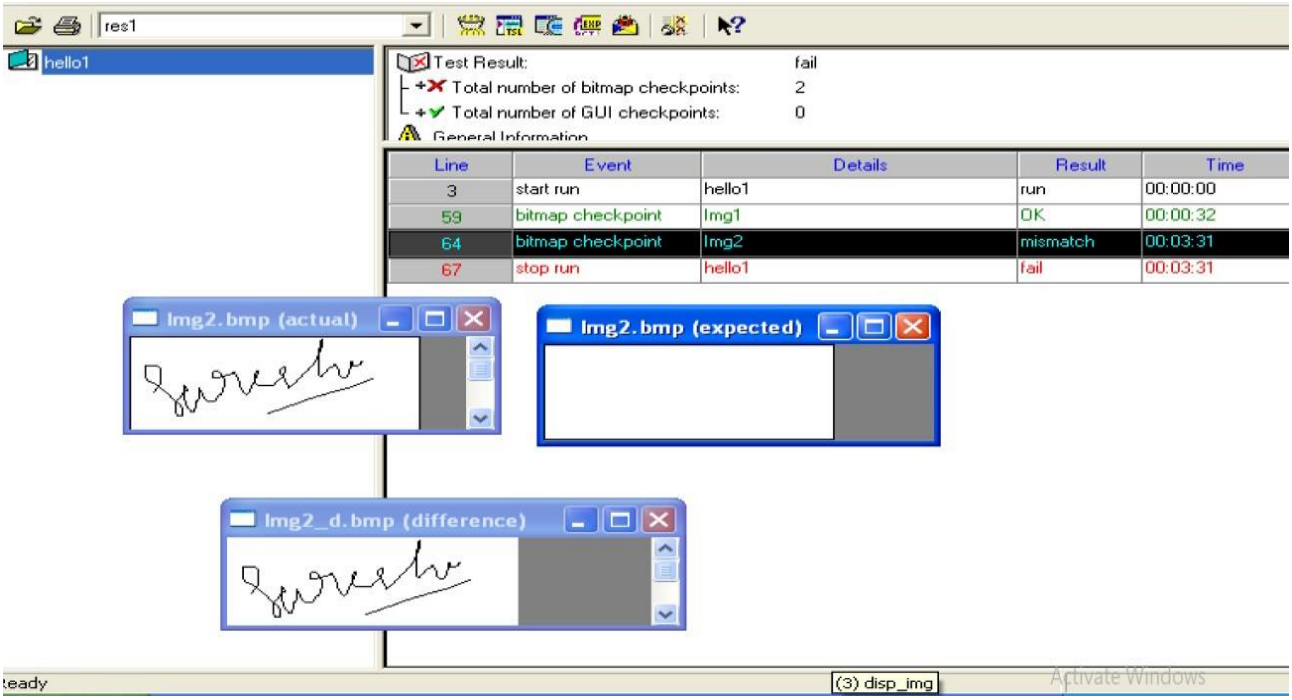
Running Test on New Version Flight 1b application



The test failed because the Agent Signature field did not clear when WinRunner clicked the Clear Signature button during the run of the test



Test Results-Expected, Actual and Difference results



b) Bitmap Checkpoint for Screen Area

Aim: To create Bitmap checkpoint for Screen Area

Steps:

1 Start WinRunner and open a new test.

2 Start the Flight Reservation application and log in.

3 Start recording in Context Sensitive mode.

4 Open order #6.

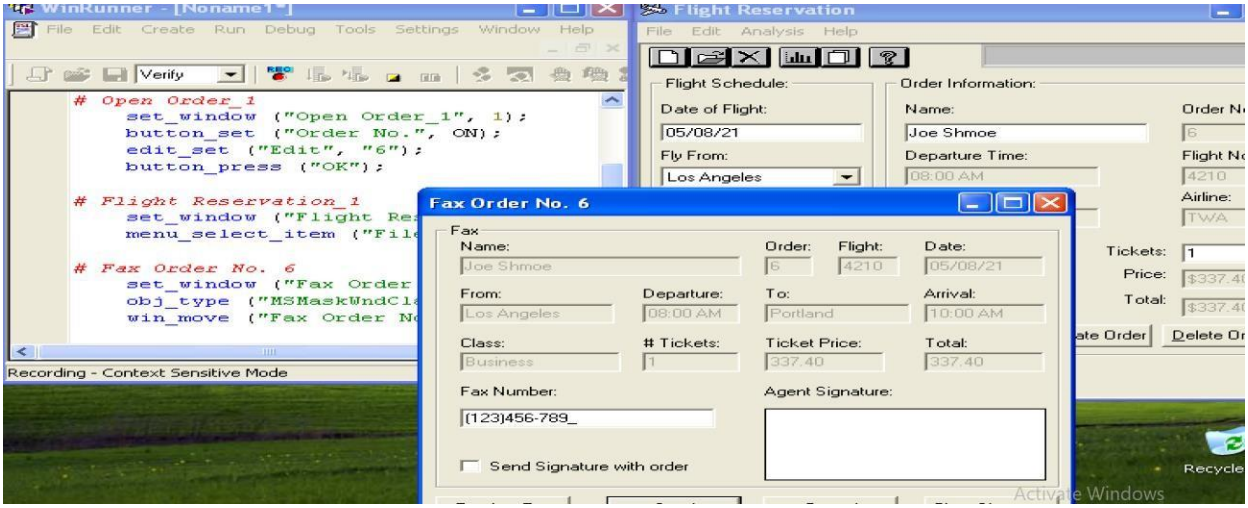
5 Open the Fax Order dialog box.

6 Insert a bitmap checkpoint for screen area, drag the mouse to select the screen

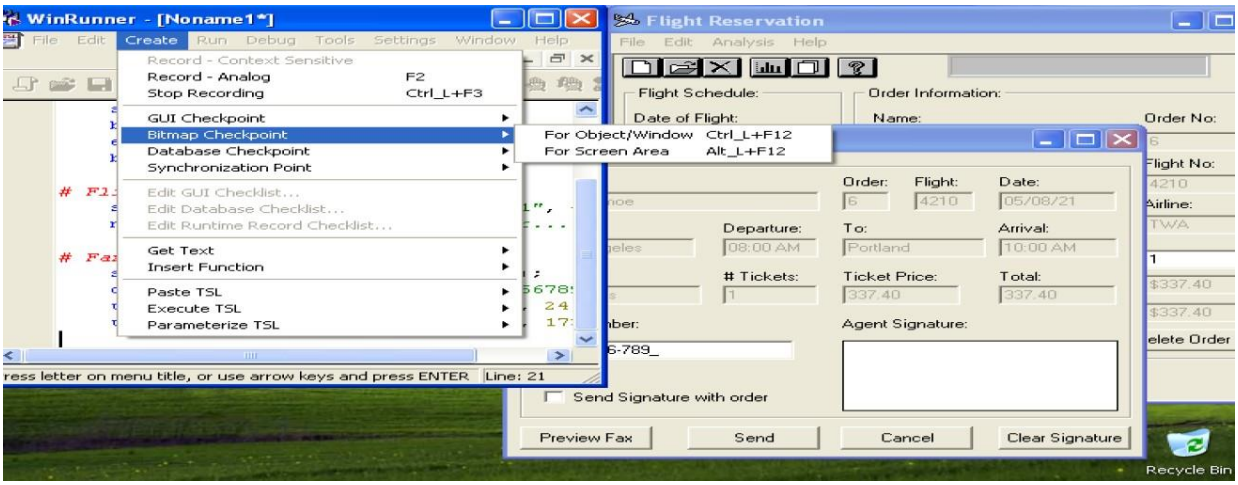
7. Stop recording and save the test

Output:

Opening Fax order window

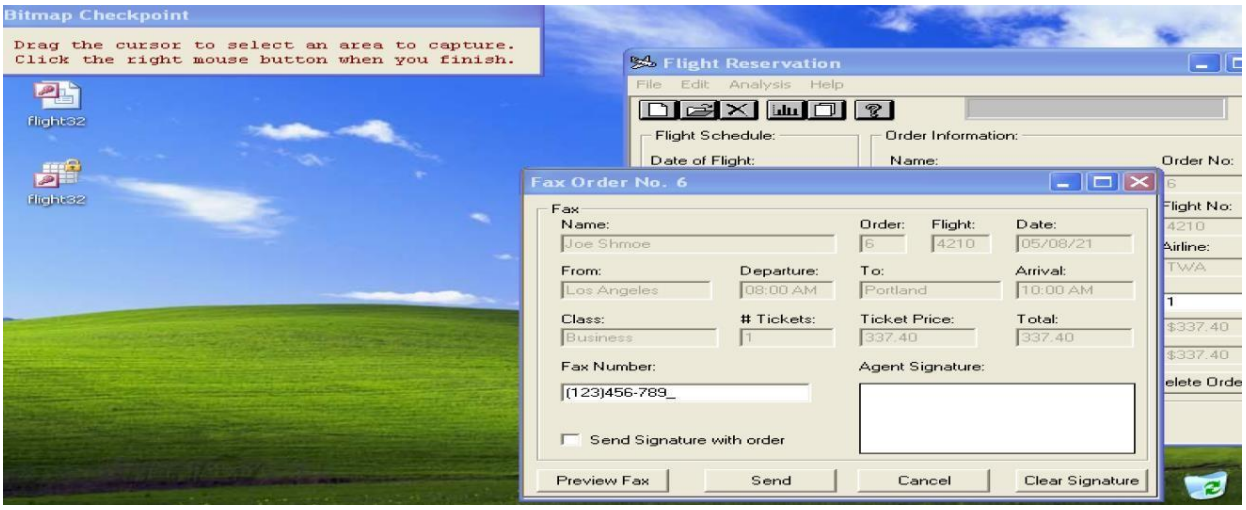


Creating Bitmap Checkpoint for Fax order Window

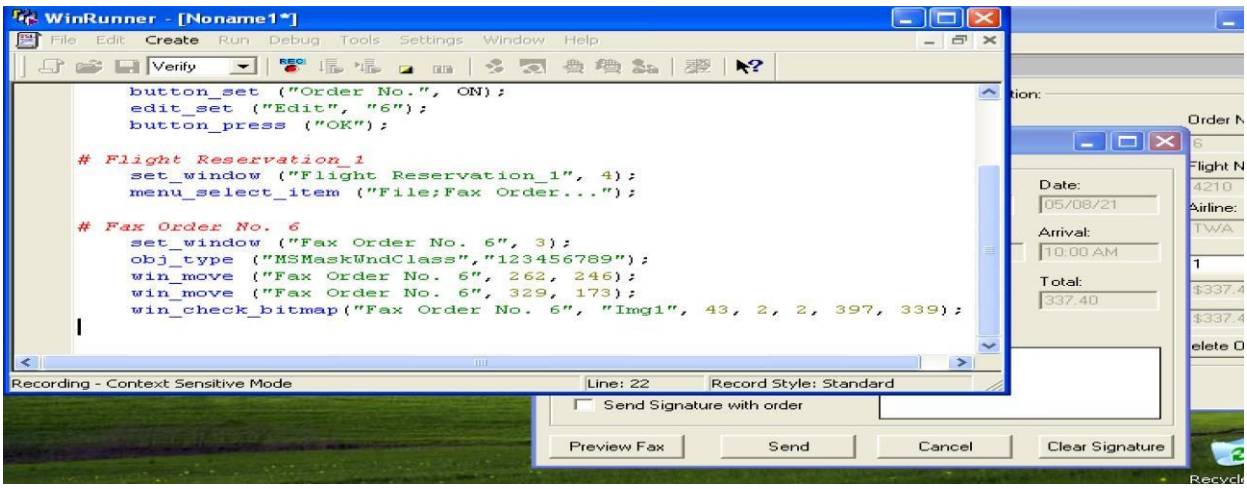


Bitmap Checkpoint window opens and drag the mouse to select Fax Order Window,

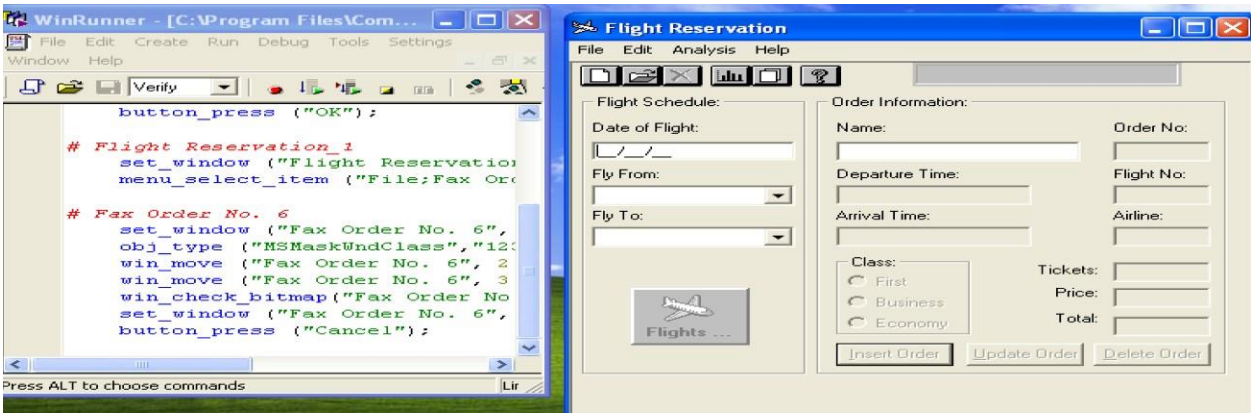
And right click the mouse to insert bit map checkpoint

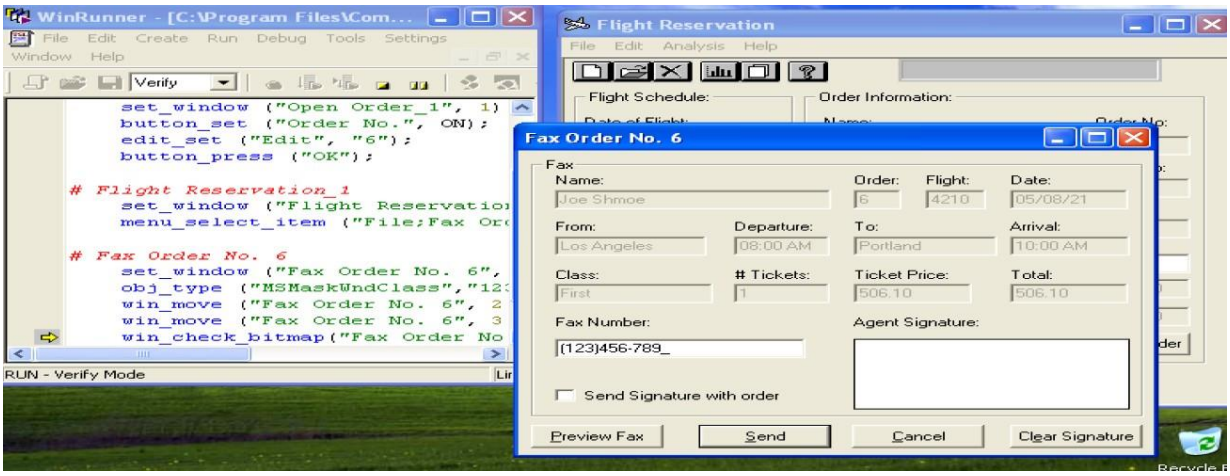


Win_check_bitmap() function is created in test script

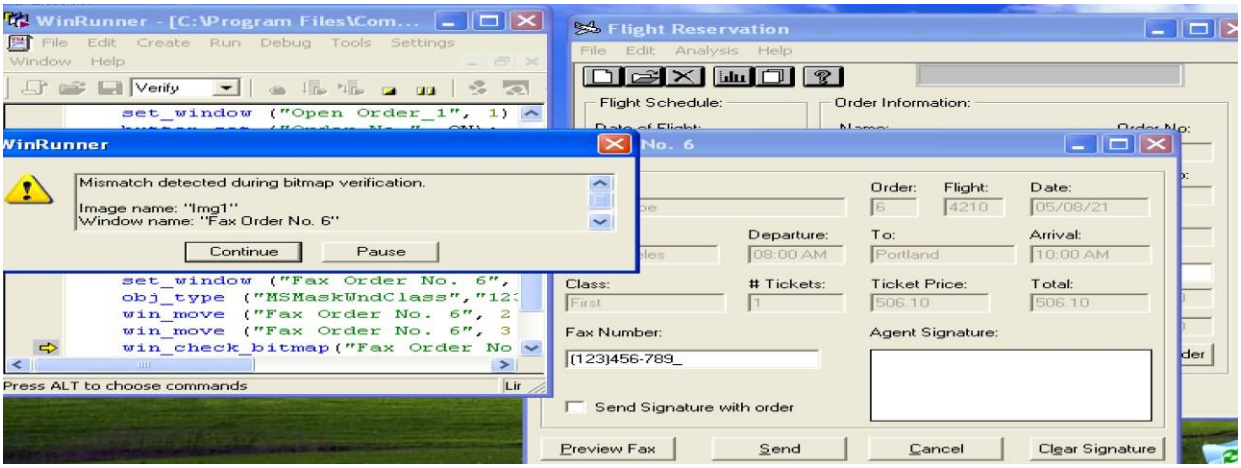


Running the test on flight 1 b application

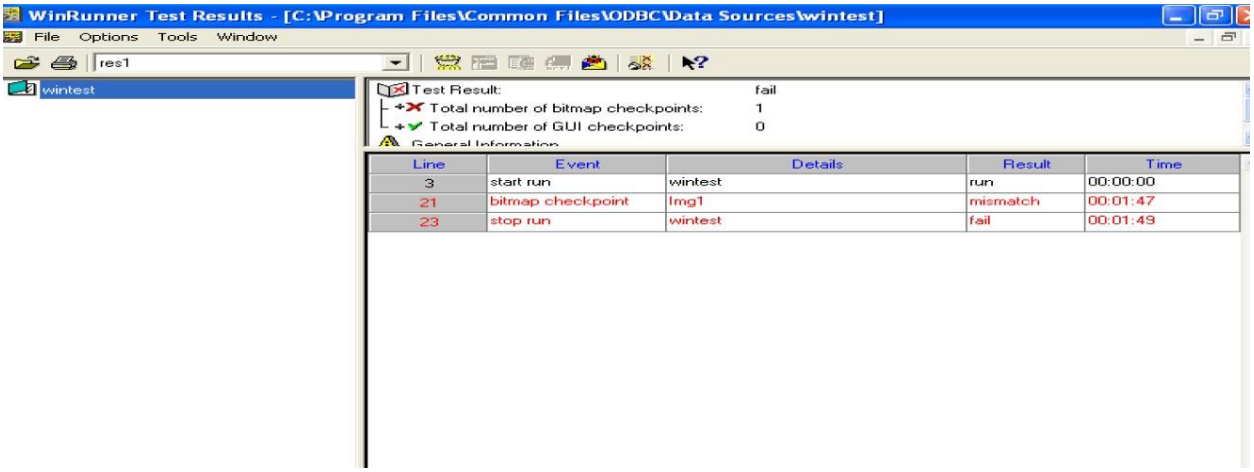




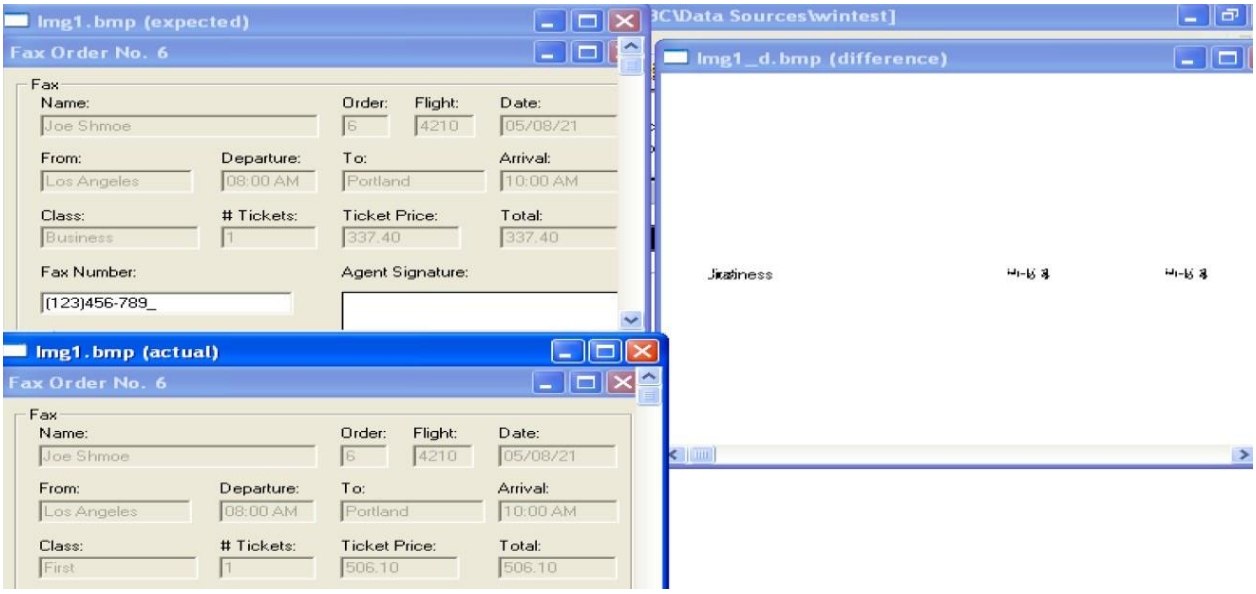
Running the test



Test Results



Test Results- Expected, Actual, Difference results



Result: Bit map checkpoints for Object and Screen Area are successfully created.

Experiment-6: Database Checkpoint for default check

Aim: To create database checkpoint for default check

Theory:

Database checkpoints can be used in the test scripts to check databases of application and the defects can be traced. In this process, define a query on the database and then create a database checkpoint that checks the properties of the results of the query. When checking these properties, we can check the contents of the results or how many rows or columns the results contain.

In the default check, the query has to be manually defined and the properties we want to test cannot be specified.

Procedure:

To create database checkpoint for default check

Choose Create> Database Checkpoint > Default Check Steps:

Steps

1 Start WinRunner and open a new test.

2 Start recording in Context Sensitive mode.

3 Choose Create> Database Checkpoint > Default Check

4 If Microsoft Query is installed and you are creating a new query, an instruction screen opens for creating a query. Click OK to close the instruction screen.

If Microsoft Query is not installed, the Database Checkpoint wizard opens to a screen where you can define the ODBC query manually.

5. Define a query, copy a query, or specify an SQL statement

6. WinRunner takes several seconds to capture the database query and restore the WinRunner window.

7. Stop recording in context sensitive mode

8. Save the test and run the test to view test results.

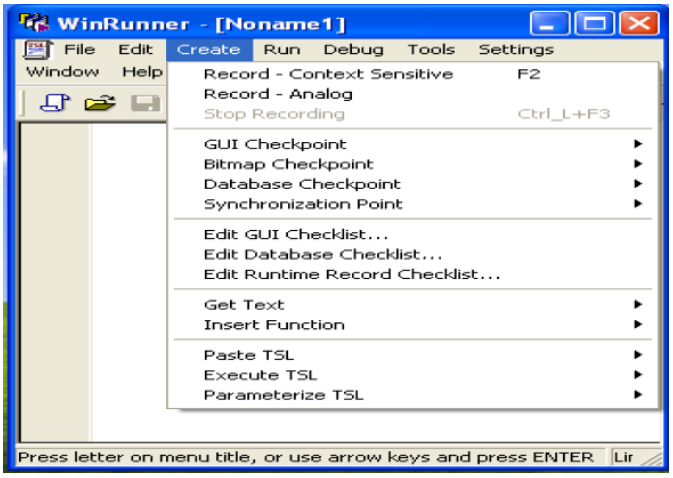
WinRunner captures the data specified by the query and stores it in the test's exp folder.

WinRunner creates the msqr*.sql query file and stores it and the database checklist in the test's chklist folder.

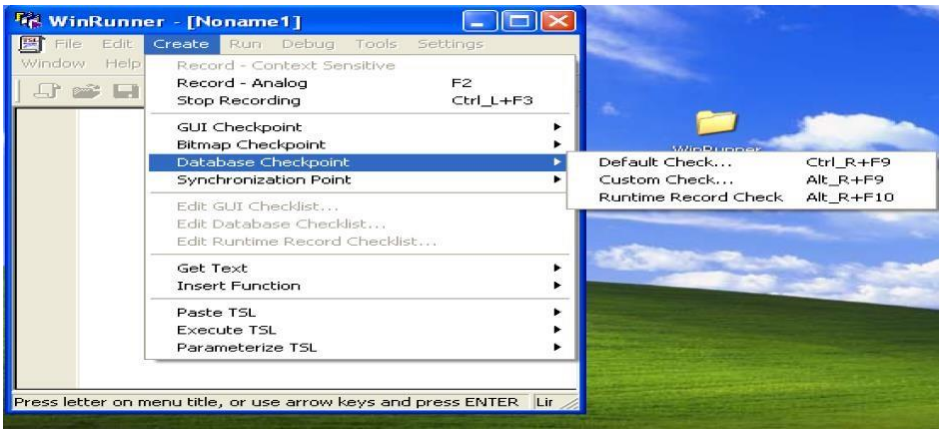
A database checkpoint is inserted in the test script as a db_check statement.

Output:

Recording test in Context Sensitive mode



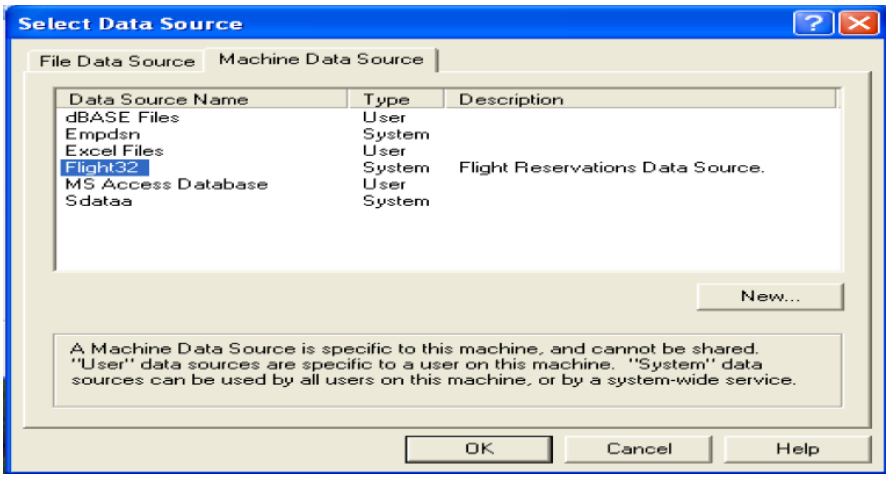
Creating Database checkpoint for Default check



Specifying SQL statement

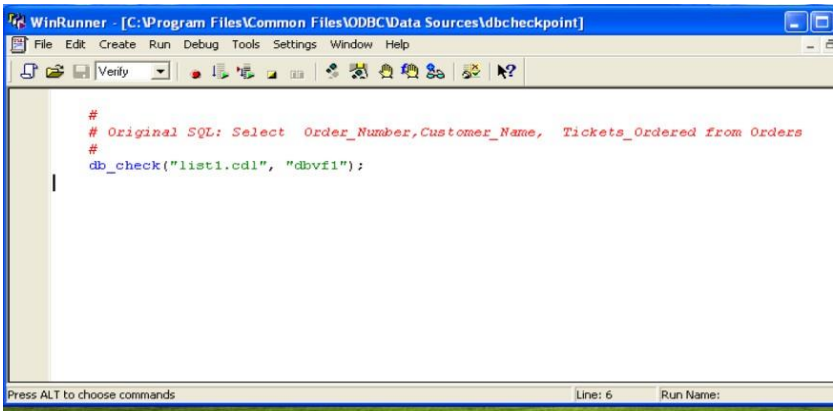
Creating connection string to Database (MS-ACESS using DSN)

Selecting Flight32 application dsn

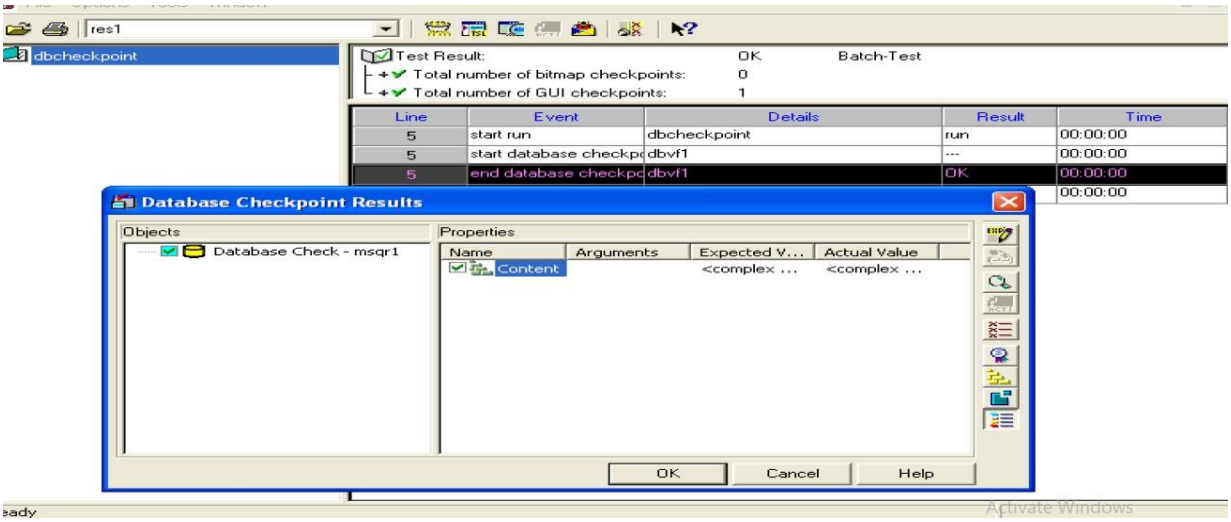


Querying database

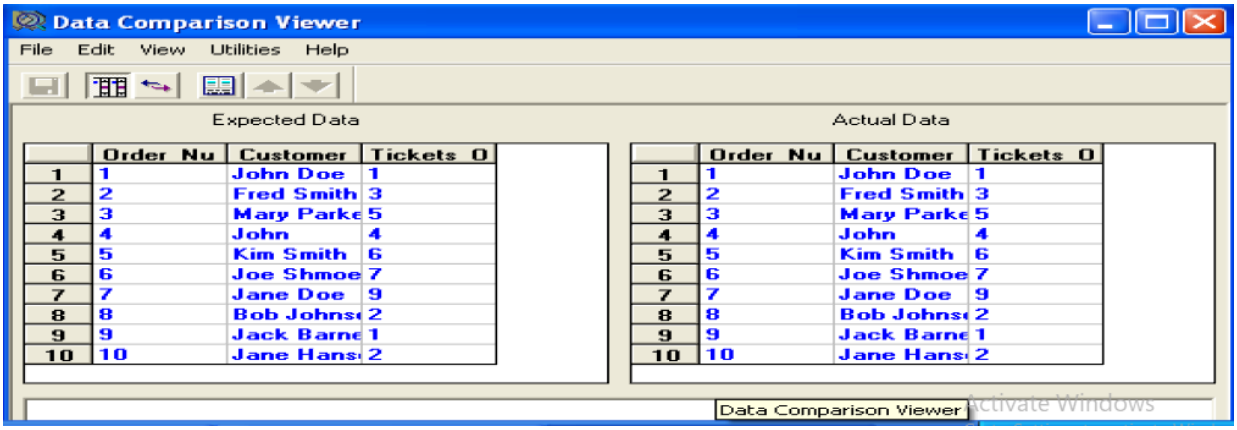
Database checkpoint is create in the test script



After Running the Test



Comparing the values stored in database and data displayed



Experiment-7: Database Checkpoint for custom check

Aim: To create database checkpoint for custom check

Theory:

In custom check, we can create a database checkpoint where we can specify which properties to check on a result set. Different operations can be done on a database by the custom check like:

Check the contents of the part or the entire result set.

Edit the expected results of the contents of the result set

Count the rows in the result set

Count the columns in the result set.

Procedure:

To create database checkpoint for default check

Choose Create> Database Checkpoint >Custom check

Steps

1 Start WinRunner and open a new test.

2 Start recording in Context Sensitive mode.

3 Choose Create> Database Checkpoint >Custom Check

4 If Microsoft Query is installed and you are creating a new query, an instruction screen opens for creating a query. Click OK to close the instruction screen.

If Microsoft Query is not installed, the Database Checkpoint wizard opens to a screen where you can define the ODBC query manually.

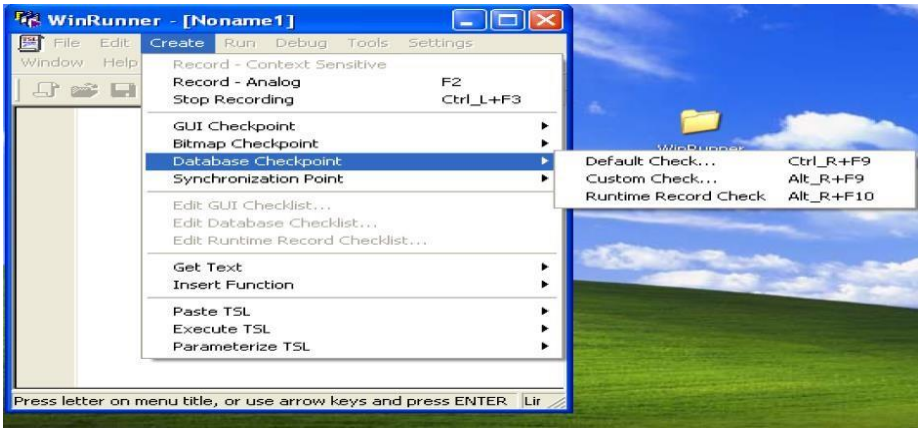
5. Define a query, copy a query, or specify an SQL statement

6. WinRunner takes several seconds to capture the database query and restore the WinRunner window.

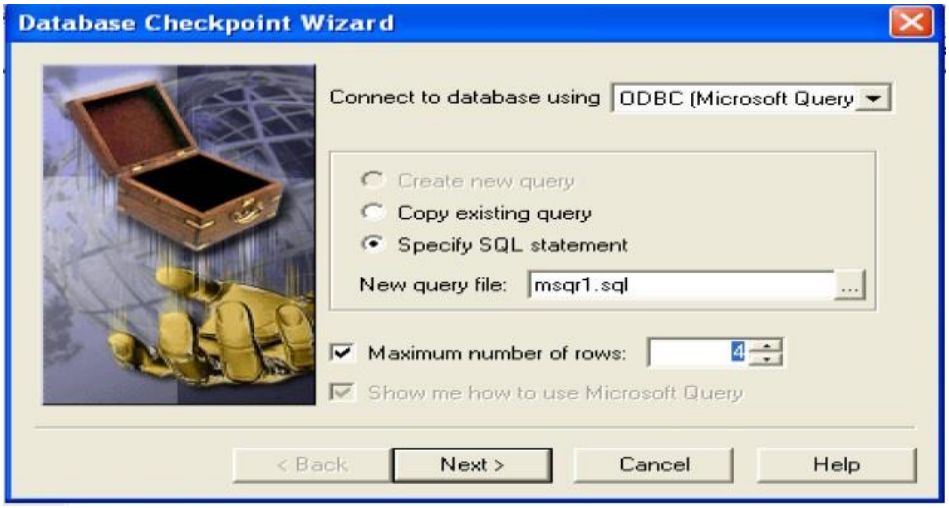
7. Stop recording in context sensitive mode

8. Save the test and run the test to view test results.

Output:
Creating Database checkpoint for Custom check

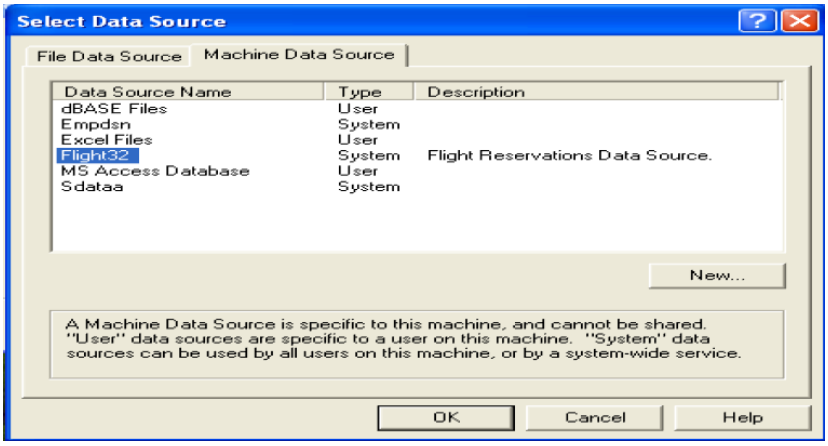


Specifying SQL statement& selecting no of rows to be verified



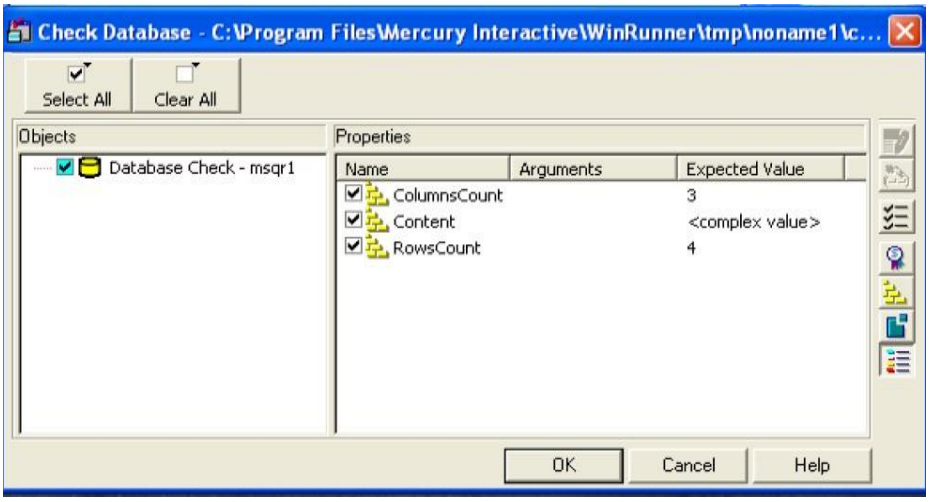
Creating connection string to Database (MS-ACESS using DSN)

Selecting Flight32 application dsn

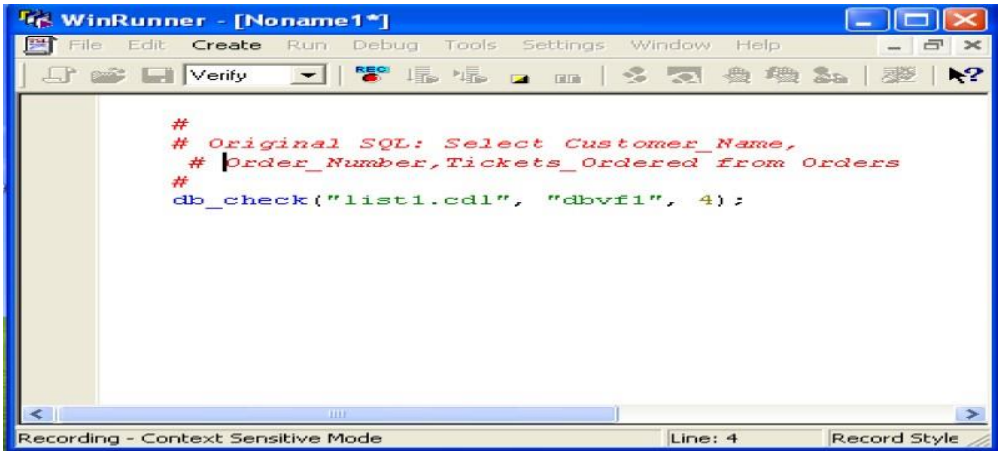


Querying database

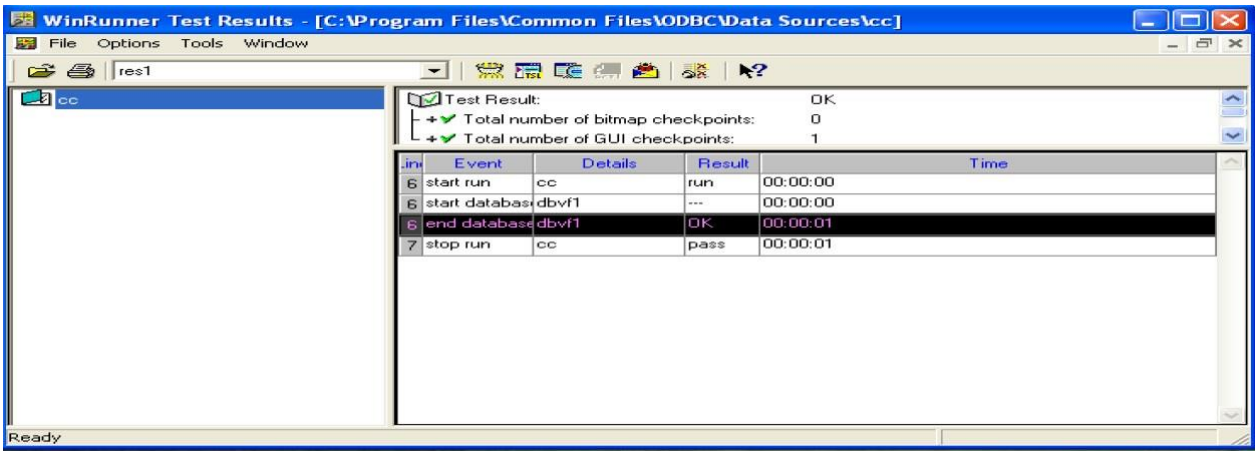
Selecting the properties to be checked in the database result set



Database checkpoint for custom check is created in test script

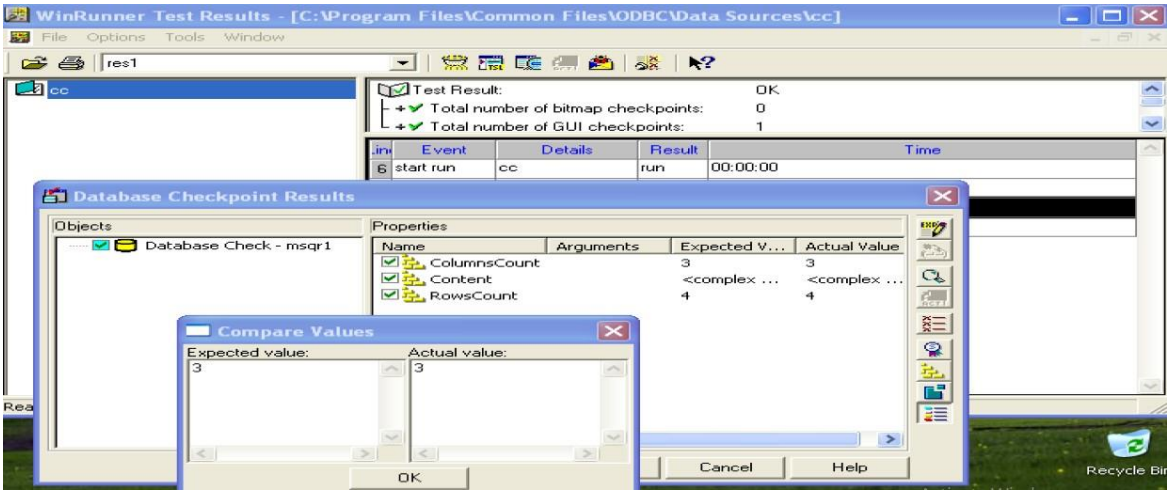


After Running the Test

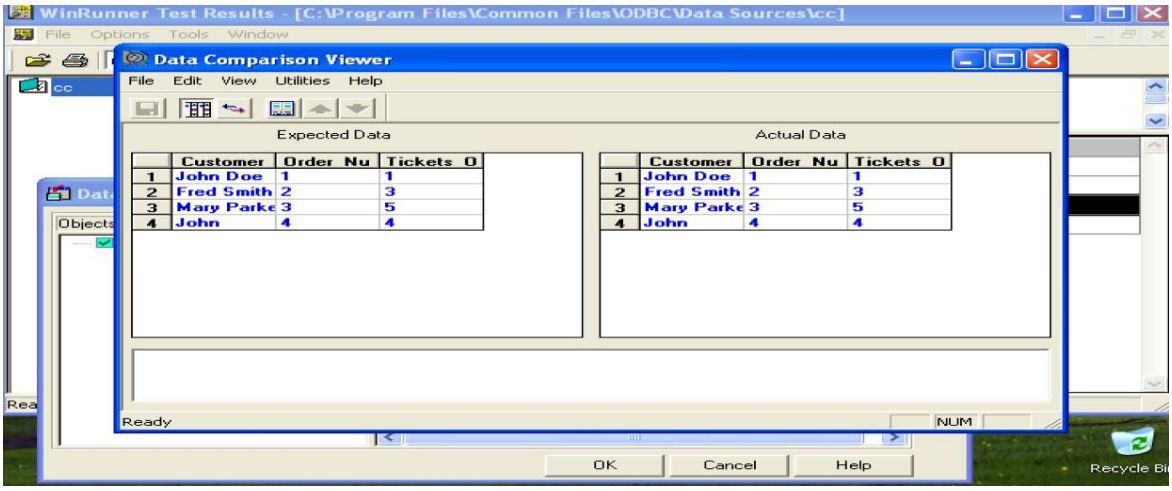


Selected properties of database verified results

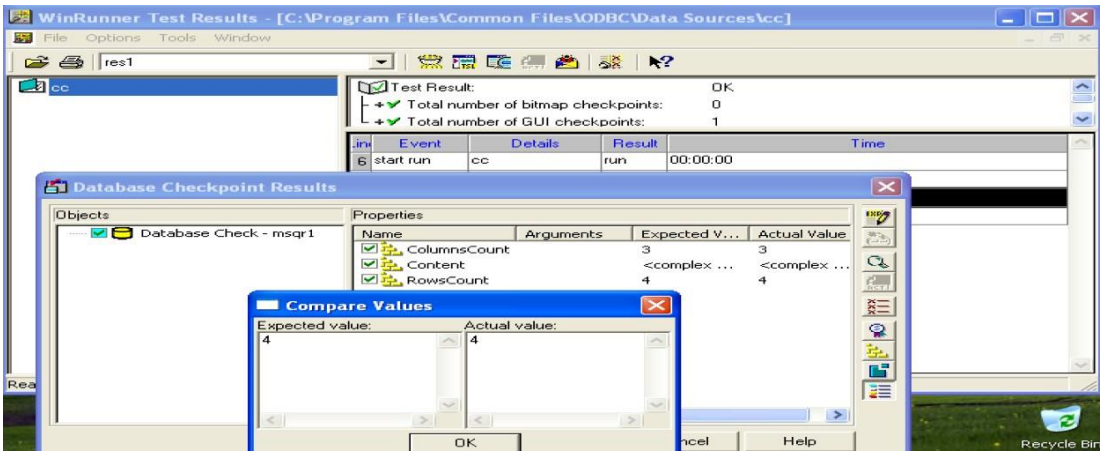
Column count property values are checked



Content of Database is checked



Row count property values are checked



Experiment-8: Database Checkpoint for runtime record check

Aim: To create database checkpoint for runtime record check

Theory:

A runtime database record checkpoint is created to compare information displayed in the application during a test run with the current value(s) in the corresponding record(s) in a database.

Runtime database record checkpoints are added by running the Runtime Record Checkpoint wizard.

When finished, the wizard inserts the appropriate db_record_check statement into test script.

The Runtime Record Checkpoint wizard guides through the steps of defining the query, identifying the application controls that contain the information corresponding to the records in the query, and defining the success criteria for the checkpoint

Procedure

To create database checkpoint for runtime check

Choose Create> Database Checkpoint >runtime record check

Steps

1Start WinRunner, Create new test

2. Open flight reservation application and login.

3 In WinRunner, Start recording in Context Sensitive mode.

4. In flight reservation application, select open order from File menu and type order no as 6,

click ok

5. In Winrunner Choose **Create> Database Checkpoint >runtime record check**

6.Select specify SQL statement form Runtime record checkpoint wizard and click next.

7. Create specify the connection string and specify SQL query and click next.

8. Map the database column field names with the field values in the application window using the hand pointer tool

9. Select the option one or more matching records.

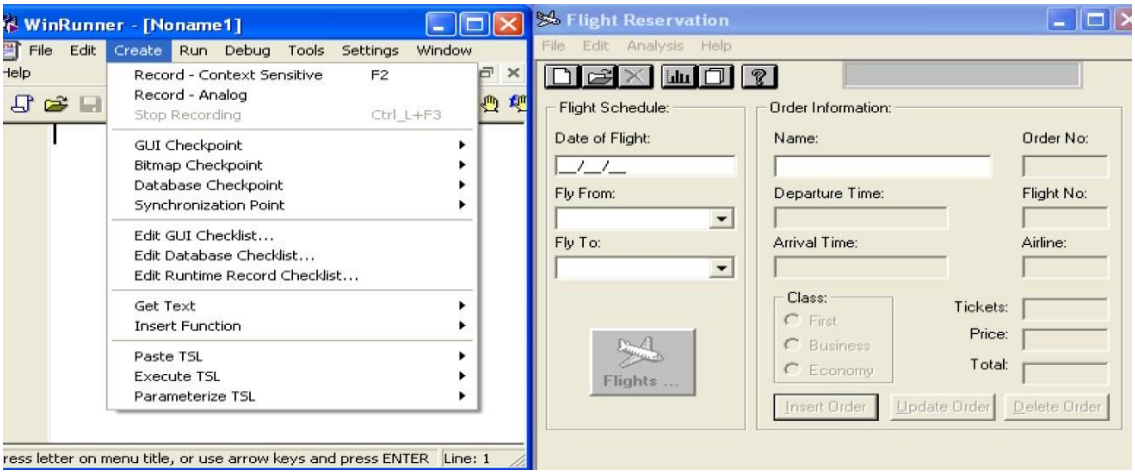
9. Click finish to create the database runtime record checkpoint.

10. Stop recording and save the test.

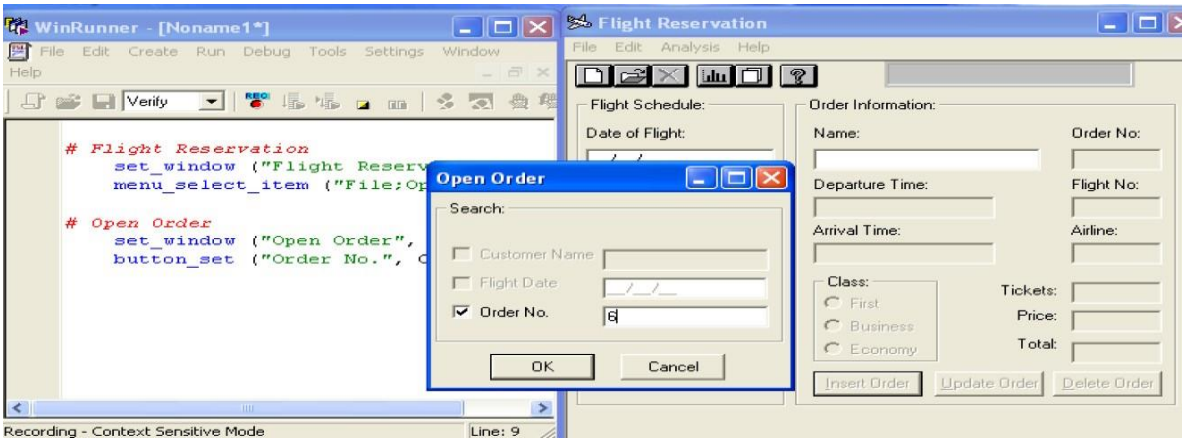
11. Run the test and view the results.

Output:

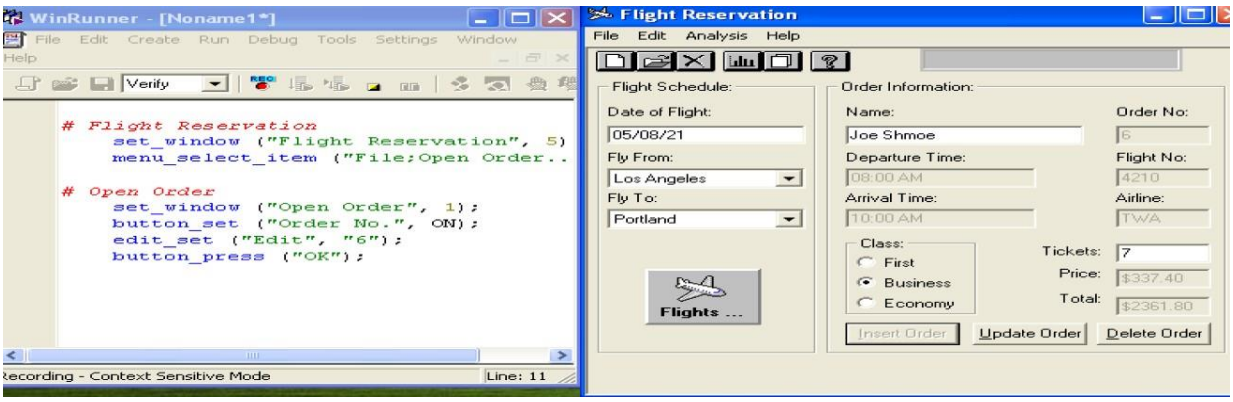
Recording the test in Context Sensitive mode



Opening order from Flight Reservation application

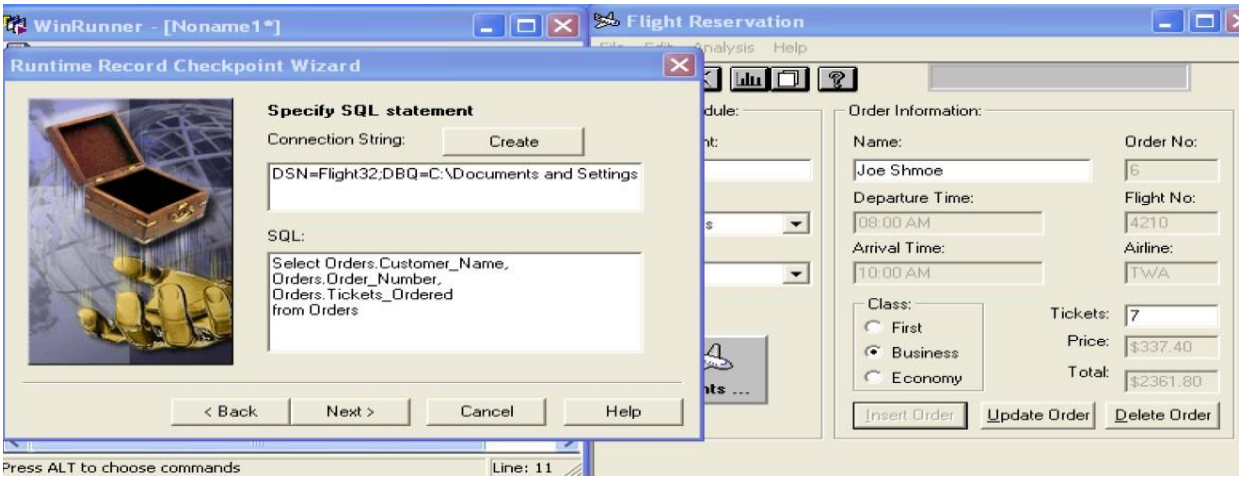


Order 6 is opened

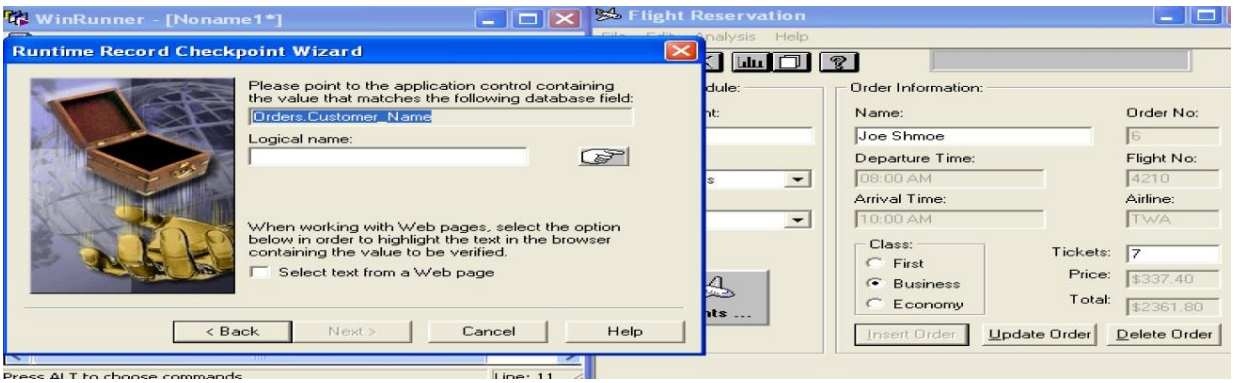


Database checkpoint for Runtime record is started

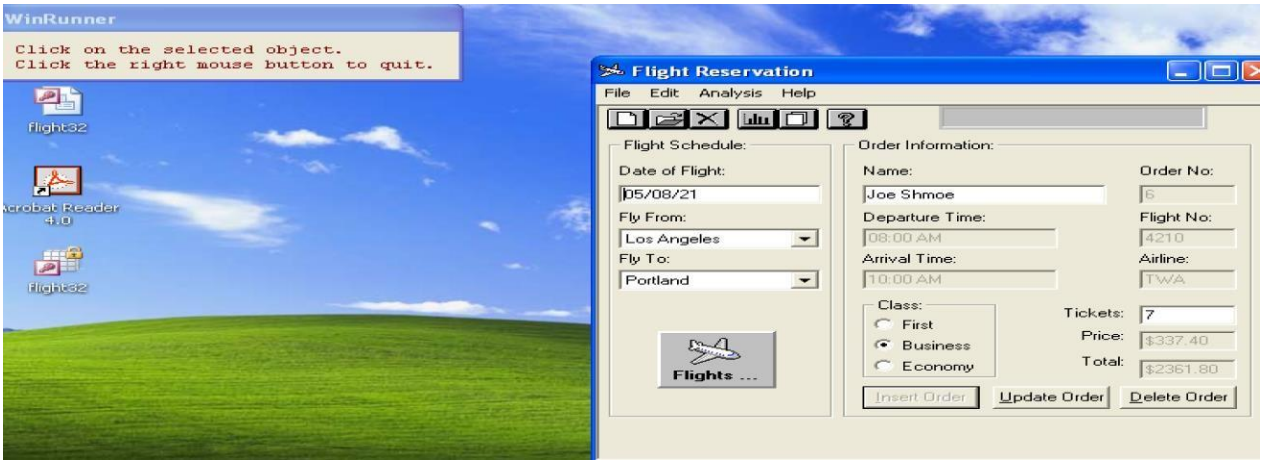
Specify Connection string to database and specify SQL statement

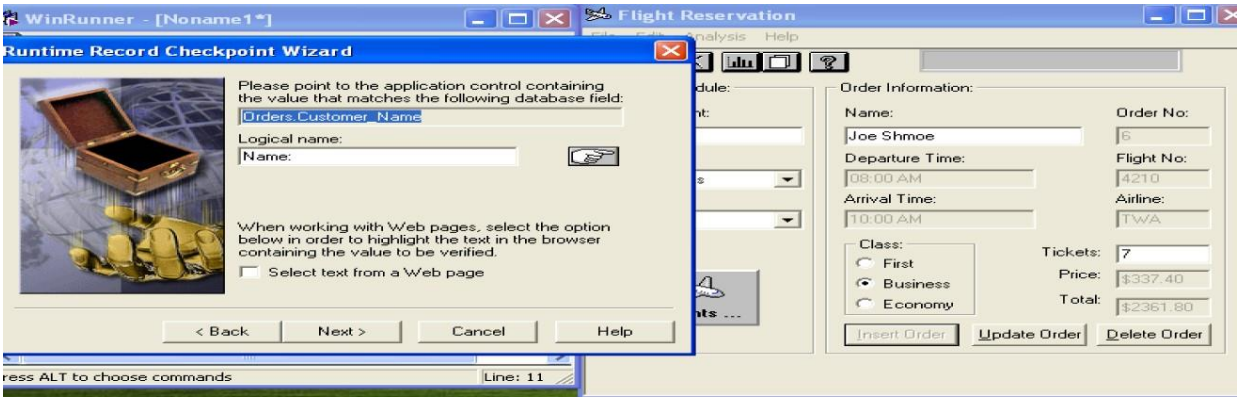


Mapping the Database table fields with the values of those fields displaying in application window by selecting Hand symbol

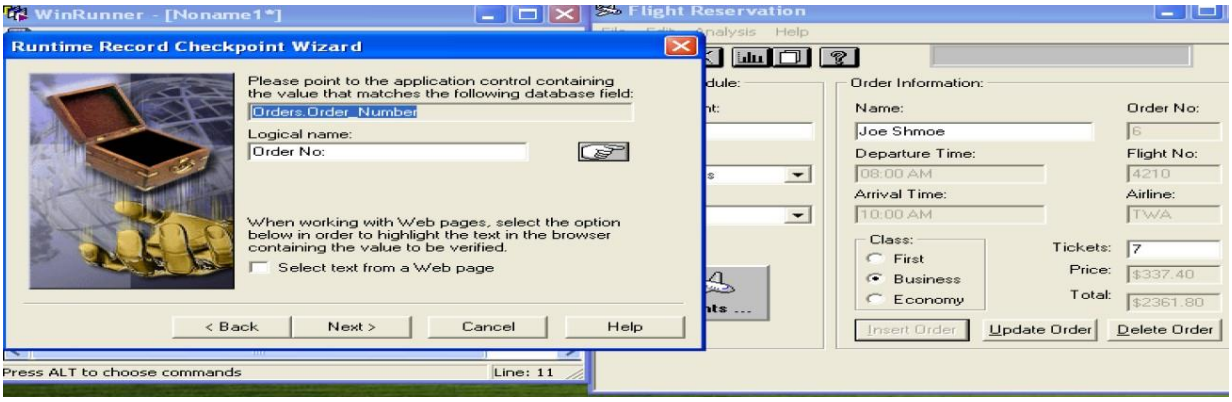


Hand tool is selected and pointed to Name text field value and double clicked on text field of Name

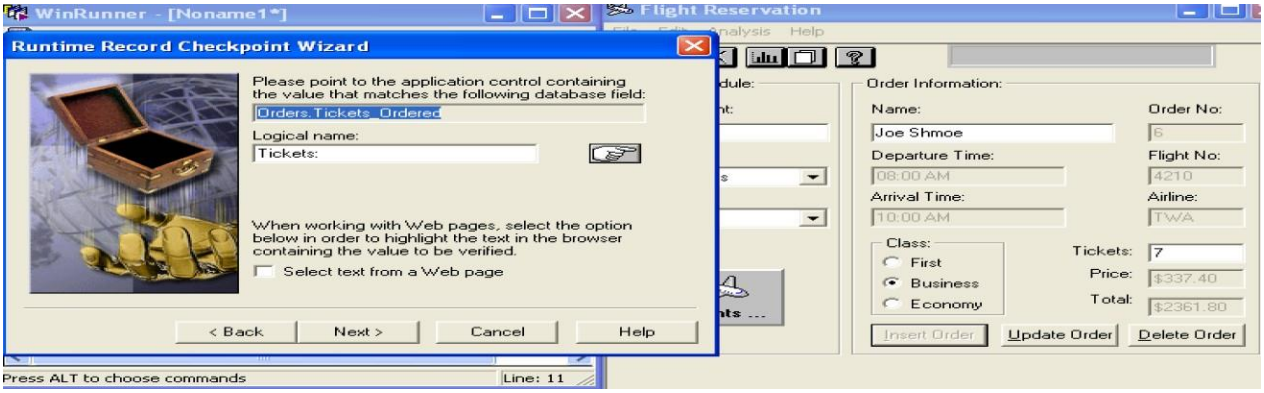




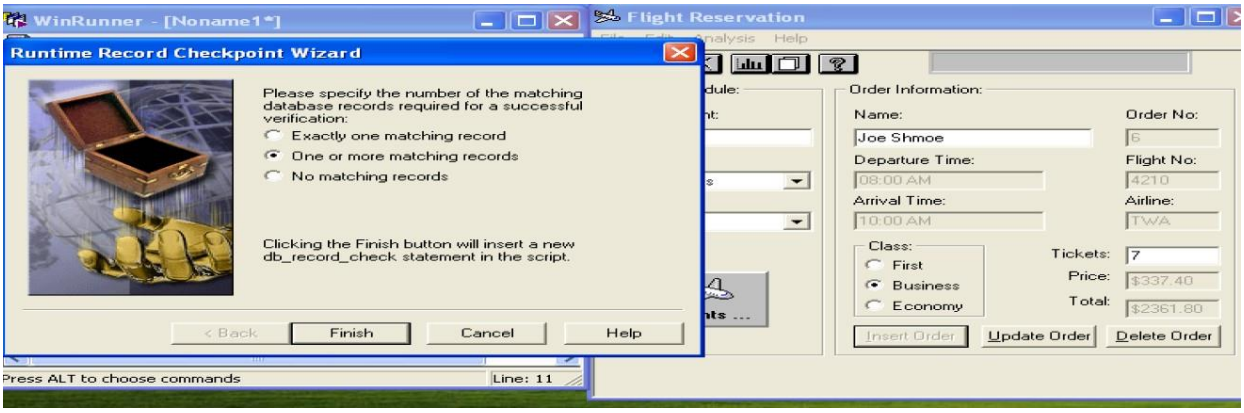
Hand tool is selected and pointed to Order No and double clicked on text field of Order No



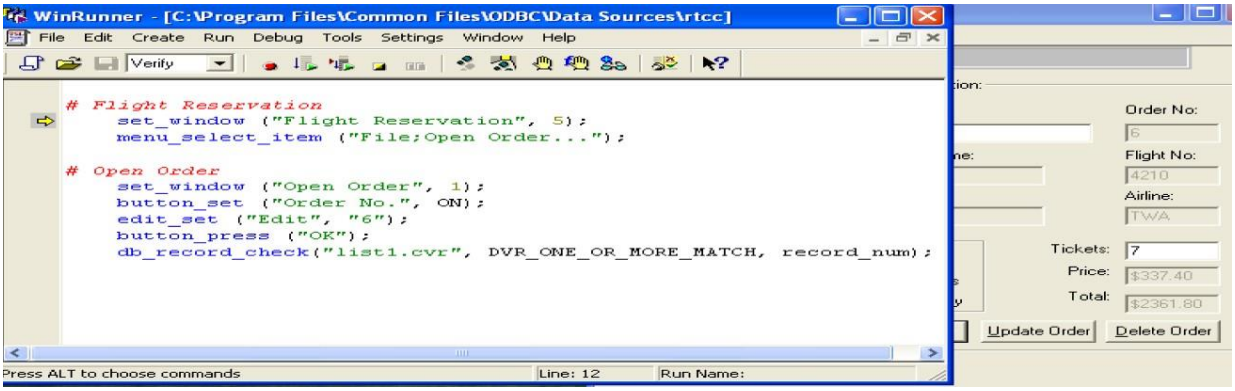
Hand tool is selected and pointed to Ticket and double clicked on text field of Tickets



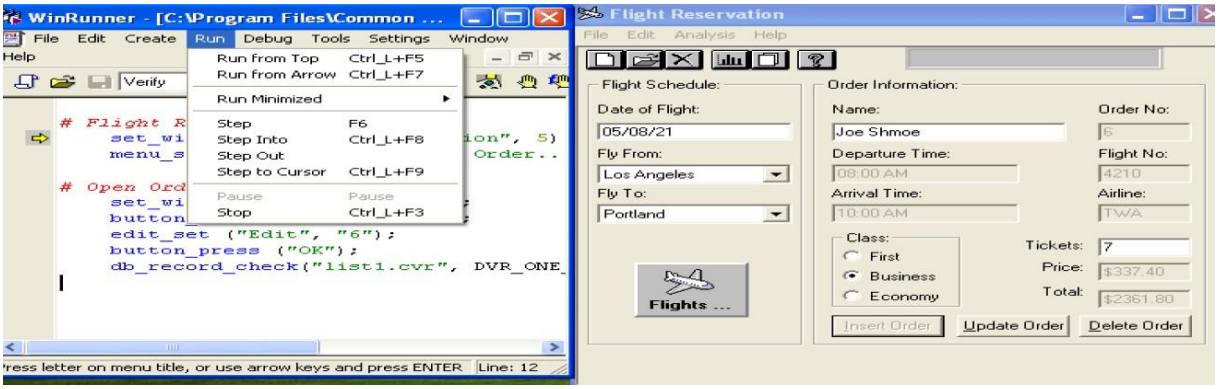
Select one or more matching records option and click finish to complete creation of runtime record checkpoint



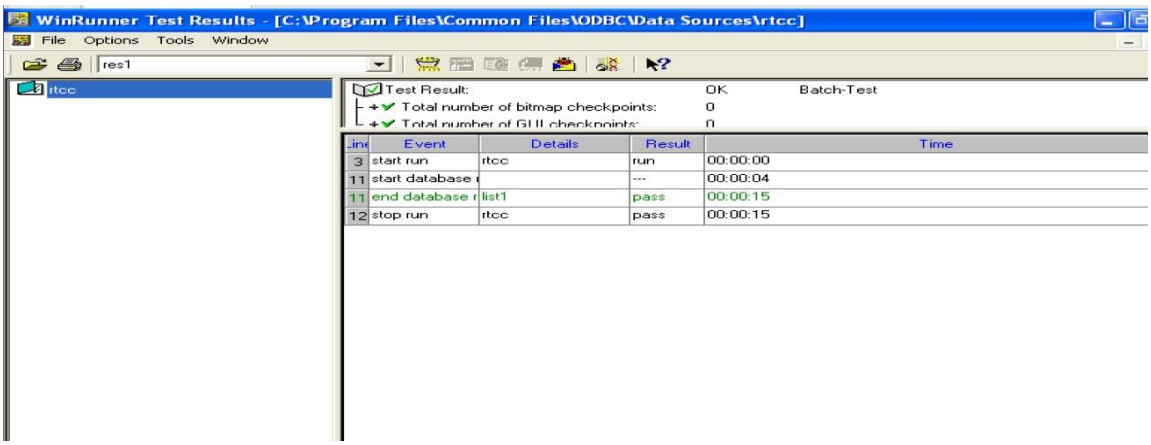
A database runtime record checkpoint is created in test script



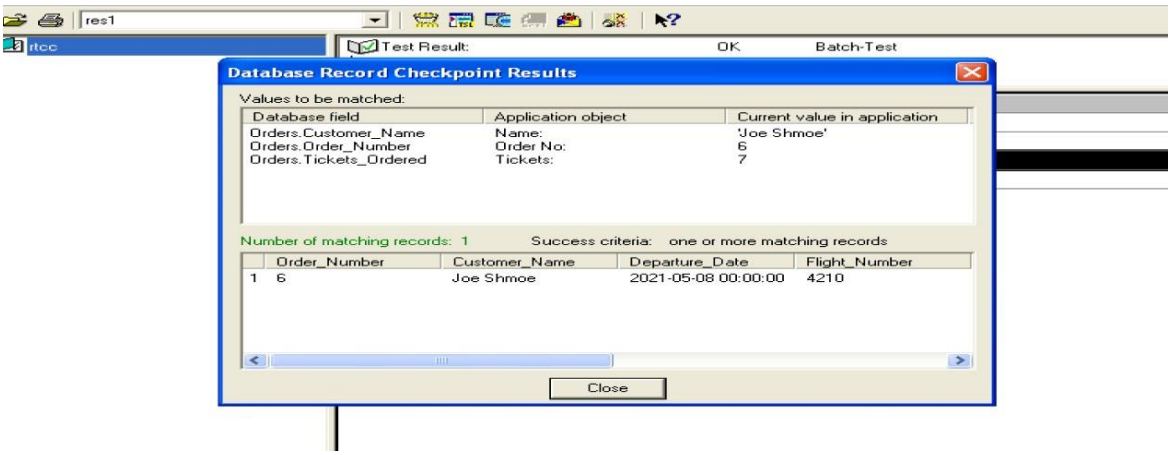
Save the test and run the test



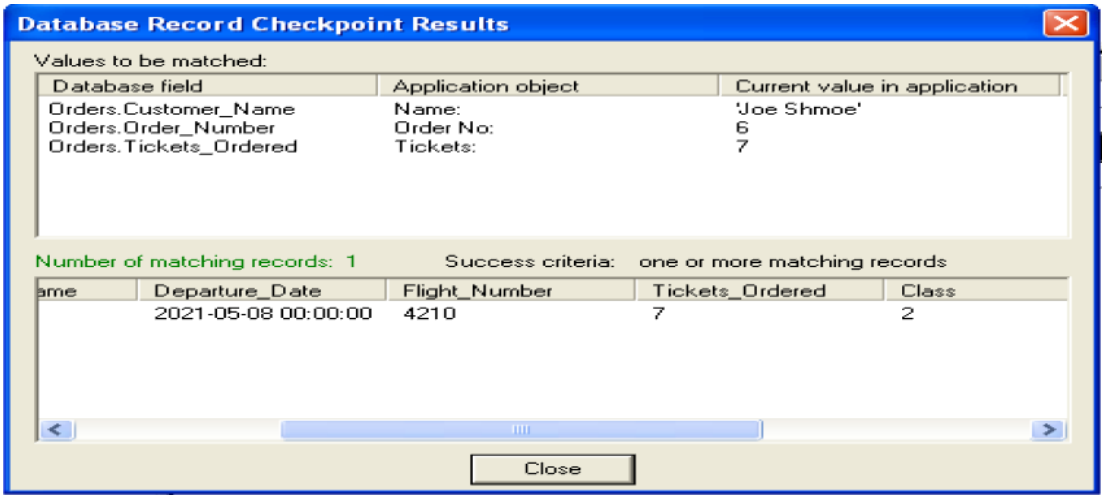
Test is executed and the results are displayed



The database field values are compared with the values that are displayed in the application window during the test run using runtime record checkpoint



Matching values are displayed



Result: We have successfully created a database checkpoint for runtime record check that verifies the values in the application and values stored in the database are matched or not.

Experiment 9: Data driven test for dynamic test data submission

Aim: To Data driven test for dynamic test data submission

Theory:

Data Driven Testing is a software testing method in which test data is stored in table or spreadsheet format. Data driven testing allows testers to input a single test script that can execute tests for all test data from a table and expect the test output in the same table. It is also called table-driven testing or parameterized testing.

Procedure:

□□□Start->Programs->Wirunner->Sample application->Flist 1A
2. Open Flight Reservation Application
3. Go to Winrunner window
4. Create->Start recording
5. Select file->new order, insert the fields; Click the Insert Order
6. Tools->Data Table; Enter different Customer names in one row and Tickets in another row.
7. Default that two column names are Noname1 and Noname2.
8. Tools->Data Driver Winzard
9. Click Next button & select the data table
10. Select Parameterize the test; select Line by Line check box
11. Click Next Button
12. Parameterize each specific values with column names of tables;Repeat for all
13. Finally Click finish button.
14. Save the Test

To Run the Test

1 In WinRunner, check that Verify mode is selected in the Standard toolbar.

2.Choose Run > Run from Top,

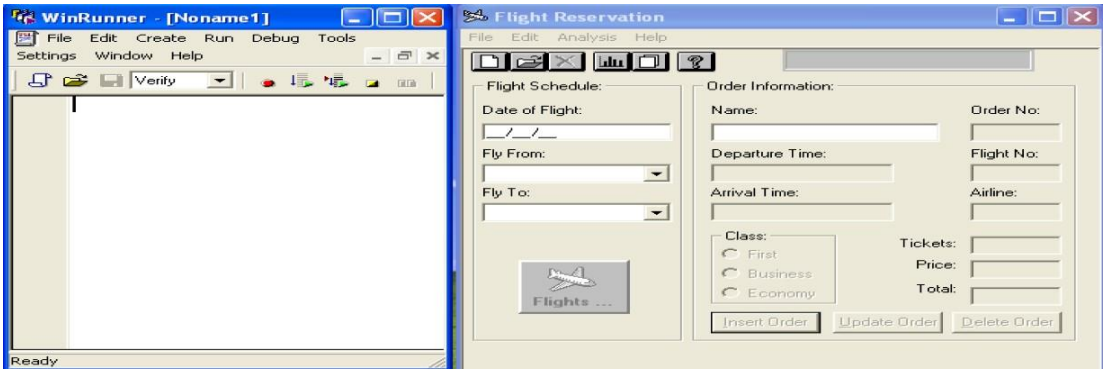
3. The Run Test dialog box opens. Accept the default test run name "res1."

Run the test Click OK in the Run Test dialog box.

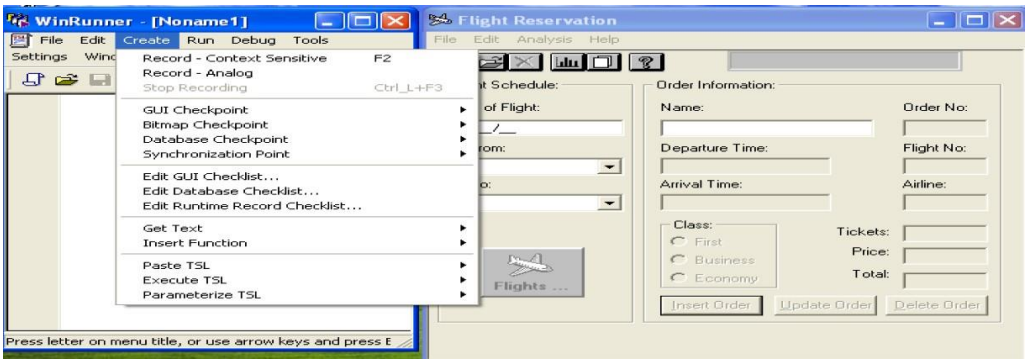
4. Review the results.

Output:

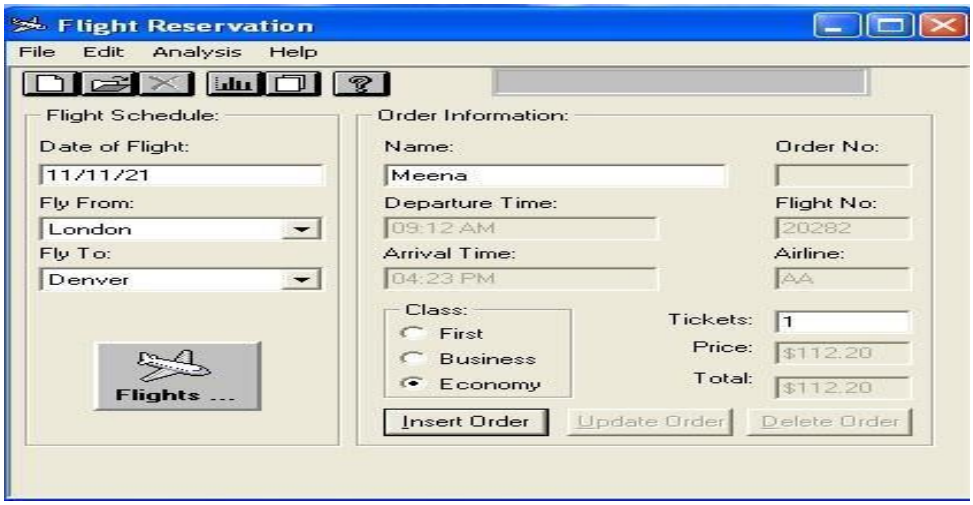
Open WinRunner tool & Application



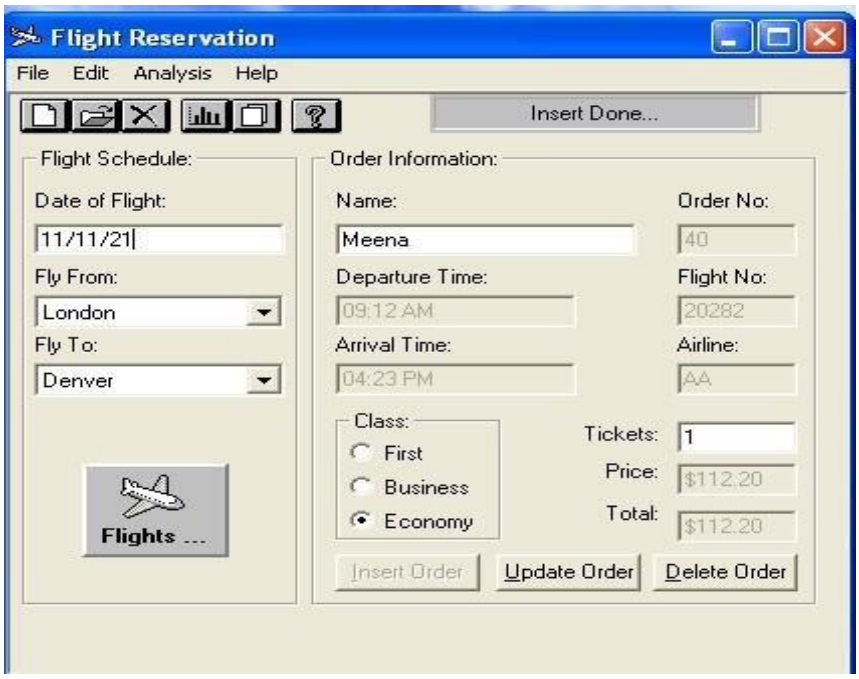
Recording in Context Sensitive Mode



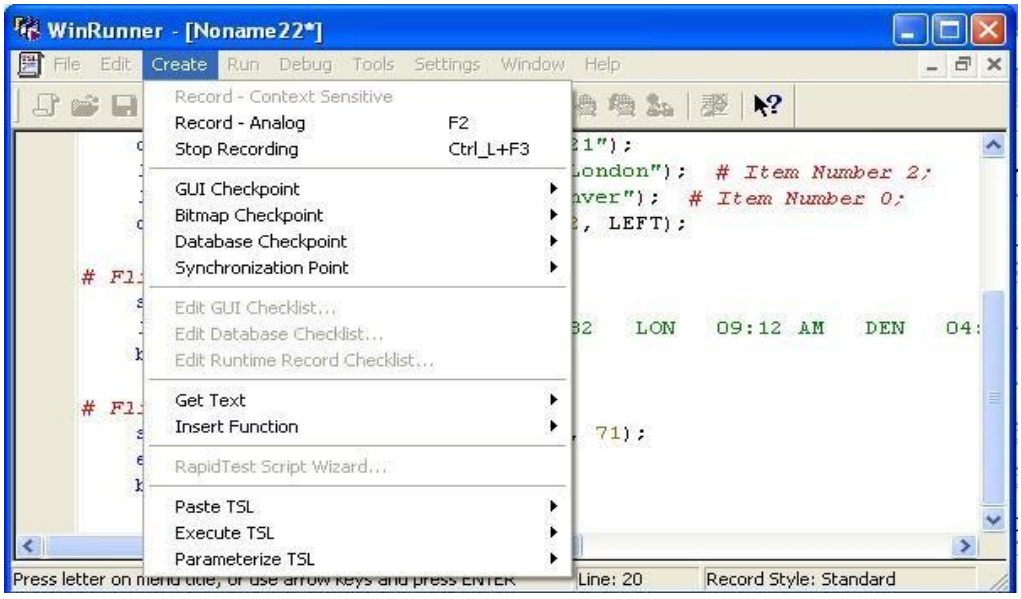
New Order from Flight Reservation Application and insert Fields .



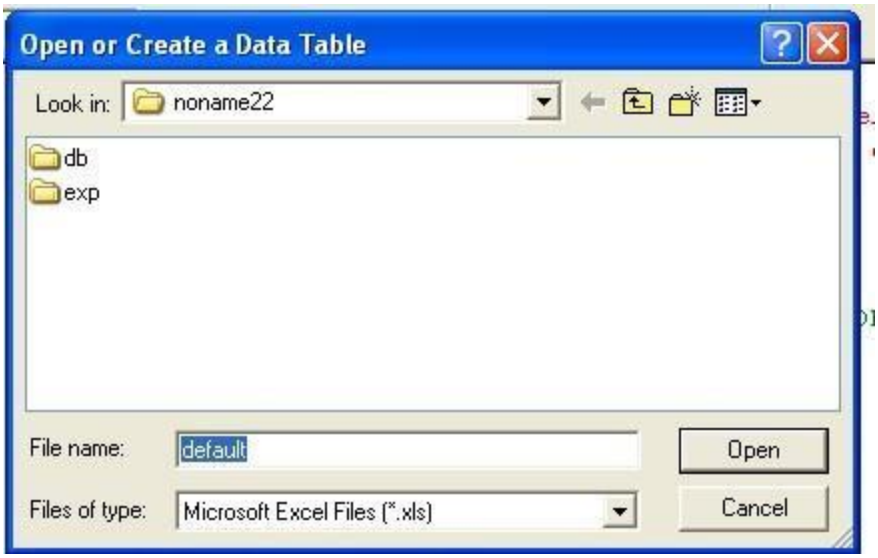
Click on Insert Order



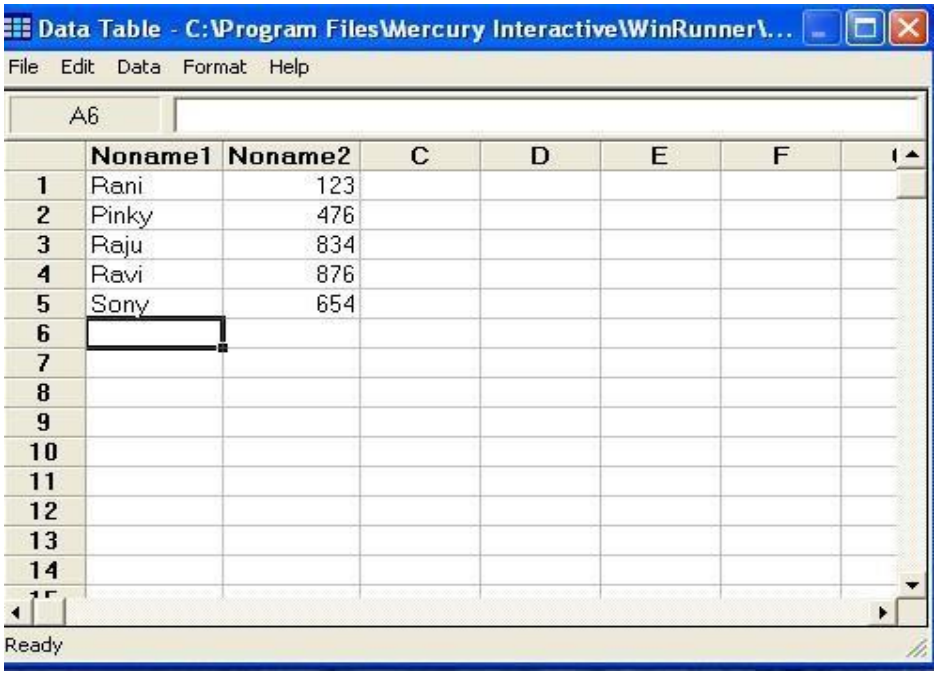
Stop recording



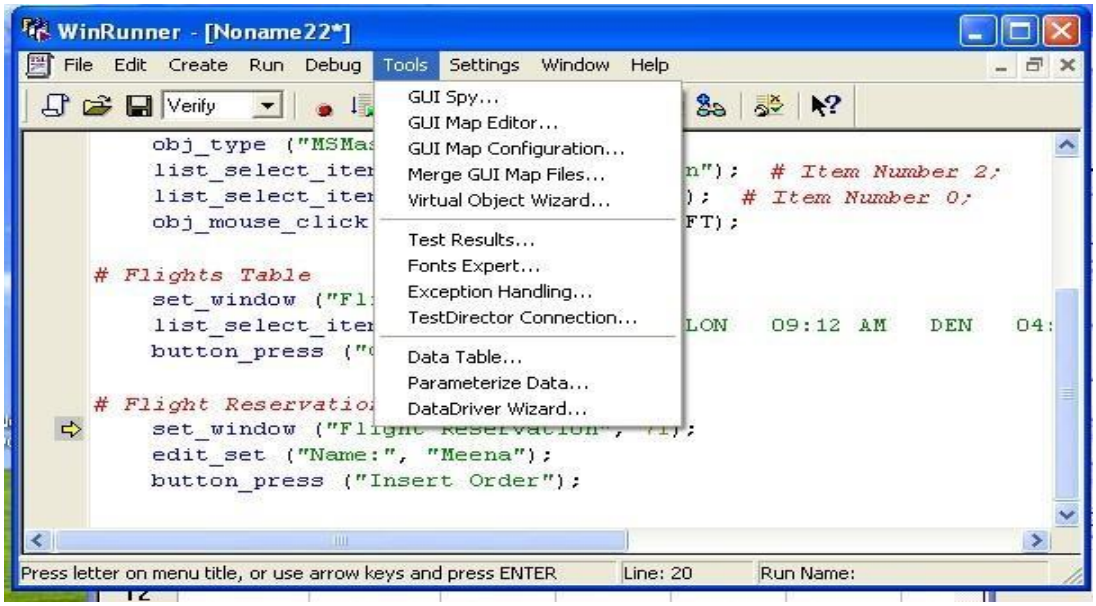
Go to Tool click on Data Table



Click on Open and enter the Fields



Go to Tools click on Data Driver Wizard

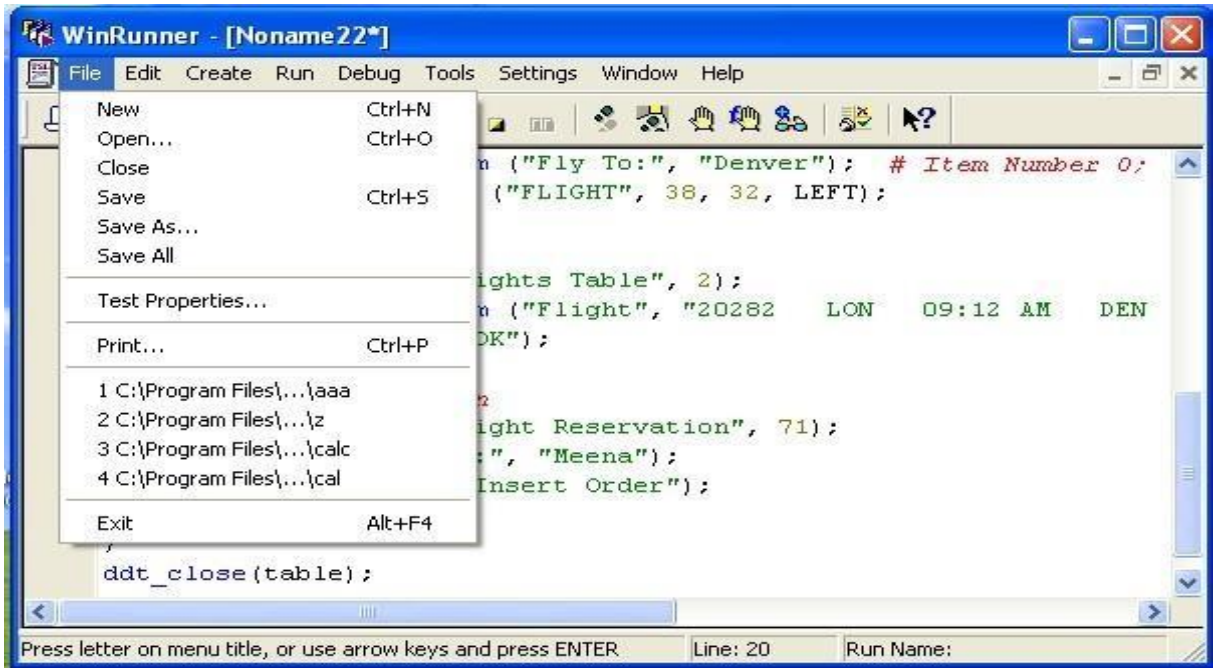


Data Driver wizard window is appear click on Next

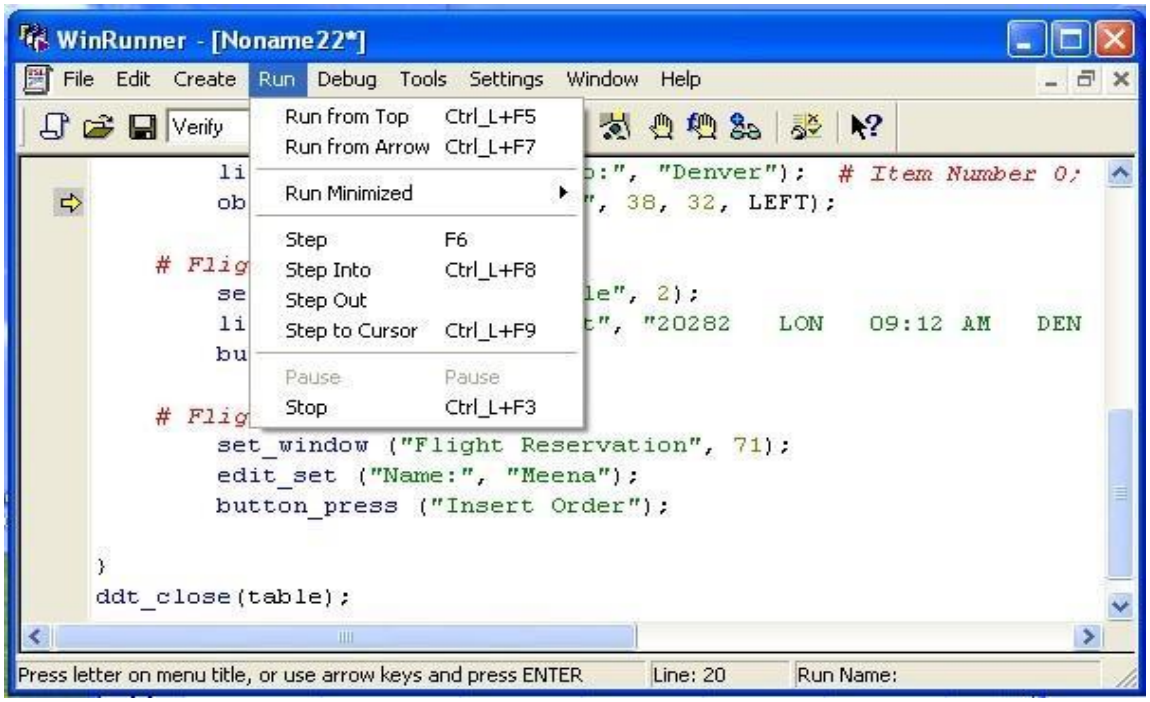
Click on Next

Click on Finish button

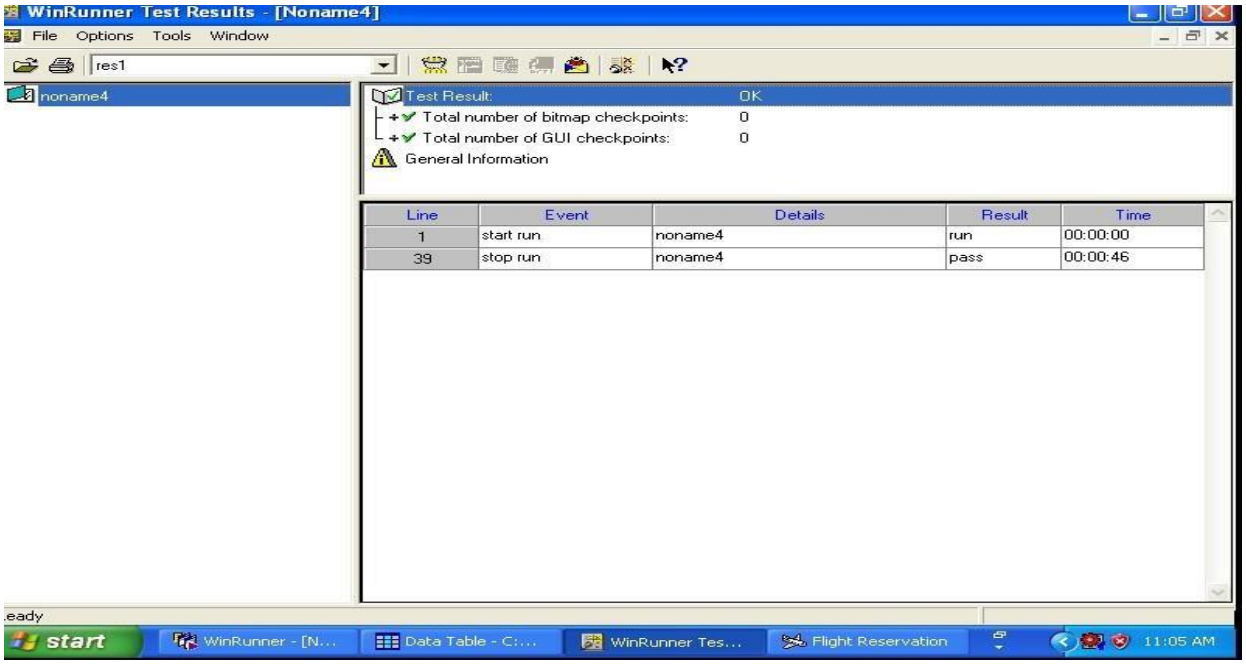
Save the Test



Run the Test



Test Result



Exp-10 : Batch testing without parameter passing

Aim: To create a Batch test without parameter passing.

Theory:

Batch test is used to run several tests and can see the results of all the tests in a single test result. A batch test contains call statements, which open other tests.

For example: call "c:\\qa\\flights\\lesson9"();

During a test run, WinRunner interprets a call statement, and then opens and runs the called test. When the called test is done, WinRunner returns to the batch test and continues the run

We **can pass parameter values from the batch test to a called test**.

Parameter values are defined within the parentheses of a call statement.

call test_name ([parameter1, parameter2, ...]);

Procedure:

1. Start WinRunner, open a new test.
2. Program call statements in the test script that call lesson5, lesson6, lesson7.
   Type the call statements into the new test window in the following format:
   call "c:\\qa\\flights\\lesson5"();
   call "c:\\qa\\flights\\lesson6"();
   call "c:\\qa\\flights\\lesson7"();
3. Define a loop that calls each test 3 times.

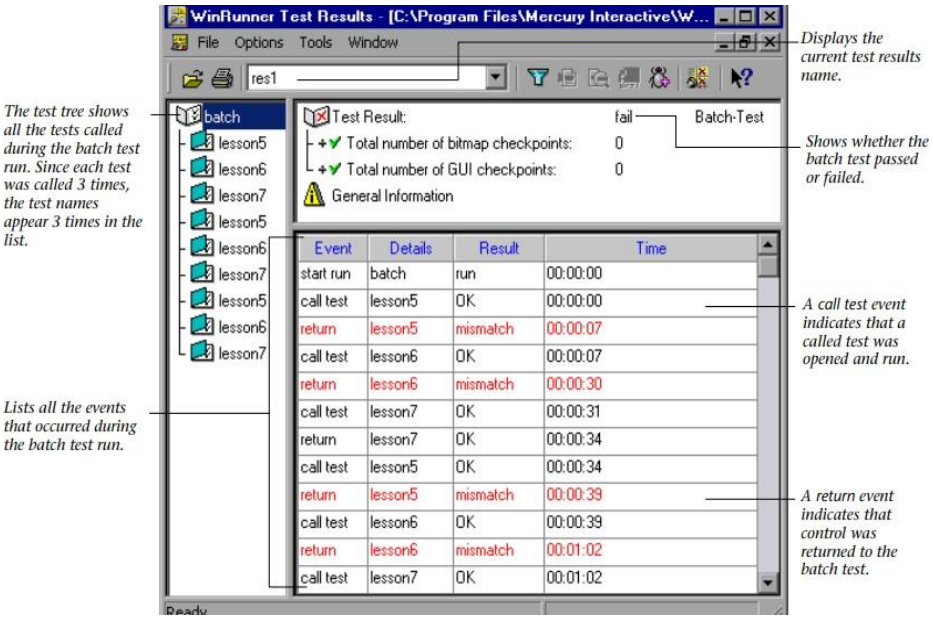
   for(i=0;i<3;i++)
   {
   call "c:\\qa\\flights\\lesson5"();
   call "c:\\qa\\flights\\lesson6"();
   call "c:\\qa\\flights\\lesson7"();
   }
4. Choose the Batch Run option in the General Options dialog box.
   Choose Tools > General Options.
   In the **General Options →the Run category→ Run in batch mode →click ok**
5. Save the batch test.

Running the Batch Test on Flight 1 B version application

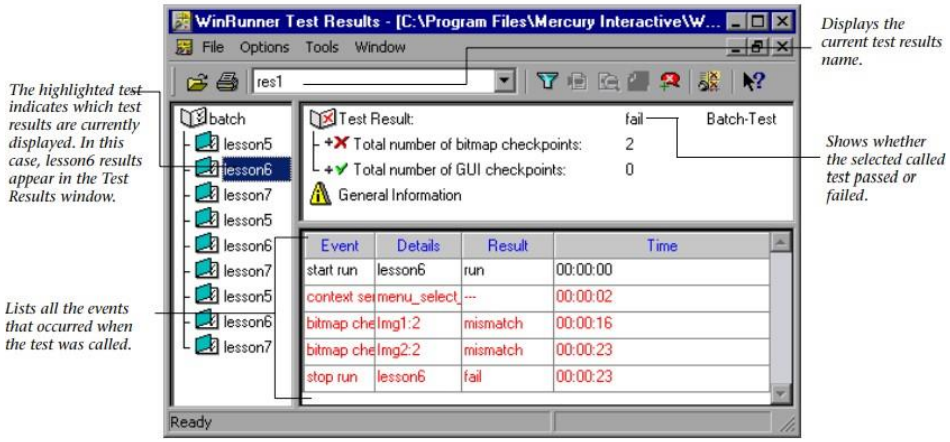
1. Open version 1B of the Flight Reservation application and log in.
   Choose Start > Programs > WinRunner > Sample Applications > Flight 1B.
2. In WinRunner, check that Verify run mode is selected in the Test toolbar.
3. Choose Test > Run from Top
4. Run the test and view the results.

Output:

Test Result



Select a test name in the test tree to view the results of a called test.



.

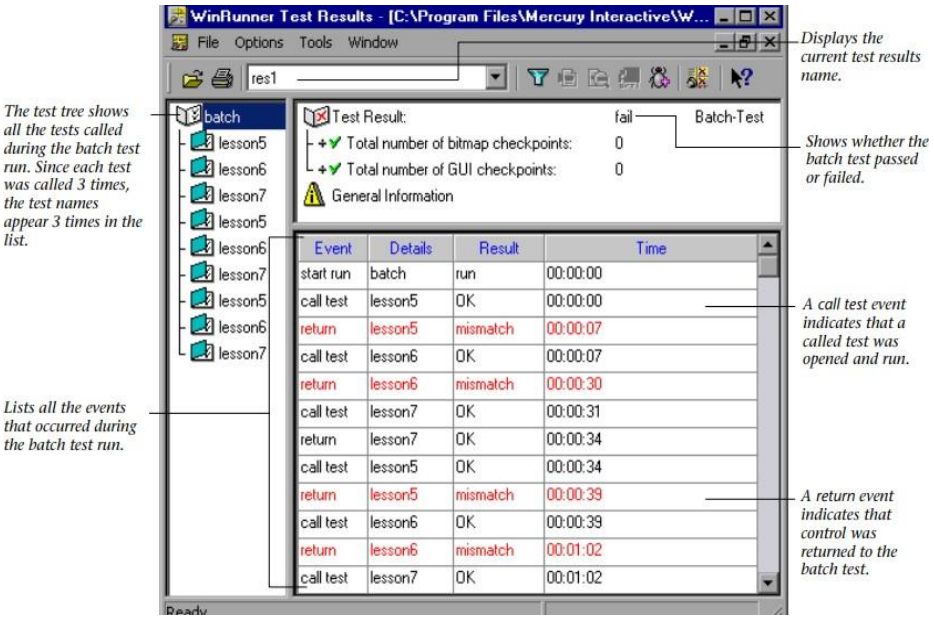
Result: Batch test executed successfully

Experiment-11 : Test case for calculator in windows application

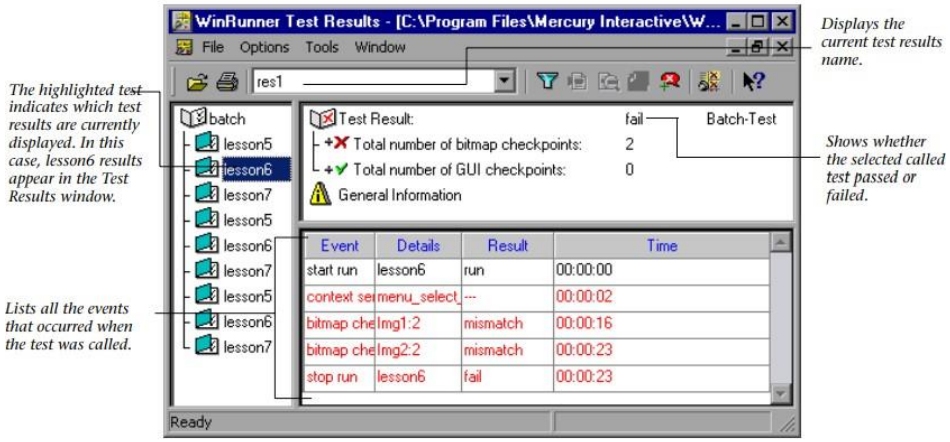
Aim: To create GUI Test case for calculator in windows application

Steps:

Test case : To test the Inverse operation (inverse of 4 using 1/x button)



Select a test name in the test tree to view the results of a called test.



.

Result: Batch test executed successfully

Experiment-11 : Test case for calculator in windows application

Aim: To create GUI Test case for calculator in windows application

Steps:

Test case : To test the Inverse operation (inverse of 4 using 1/x button)

Step 1: Open WinRunner application.
Step 2: Open Calculator application.
Step 3: Create a new document
Step 4: Start recording
Step 5: Select the Calculator application and start recording the actions. a Click "4" on the Click the "1/x" button on the Calculator to find the inverse of 4.

- Calculator The result, 0.25 will be displayed on the Calculator.

Step 6: Stop the Recording process.

Step7: Save the test

To Run the Test

1In WinRunner, check that Verify mode is selected in the Standard toolbar.

2.Choose Run > Run from Top,

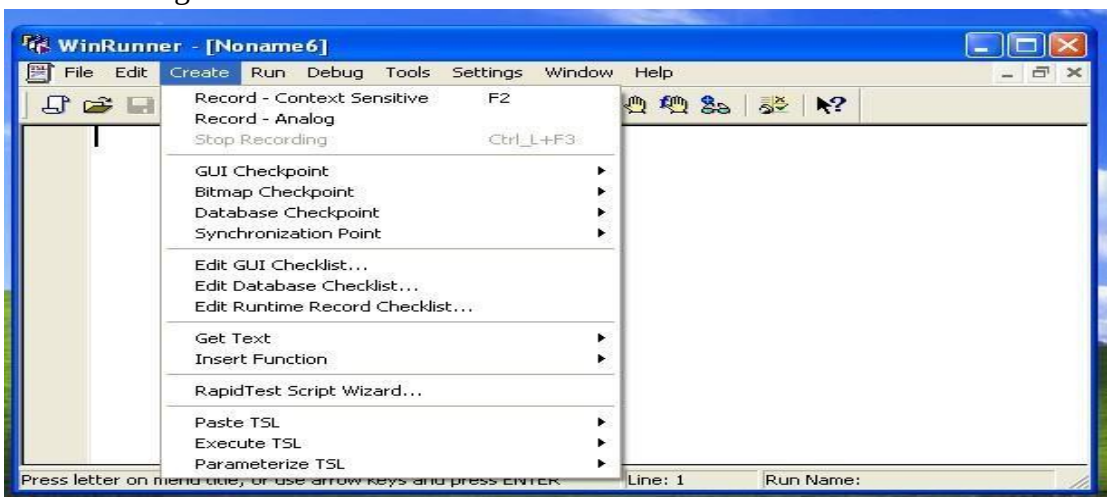
3. The Run Test dialog box opens. Accept the default test run name “res1.”

Run the test Click OK in the Run Test dialog box.

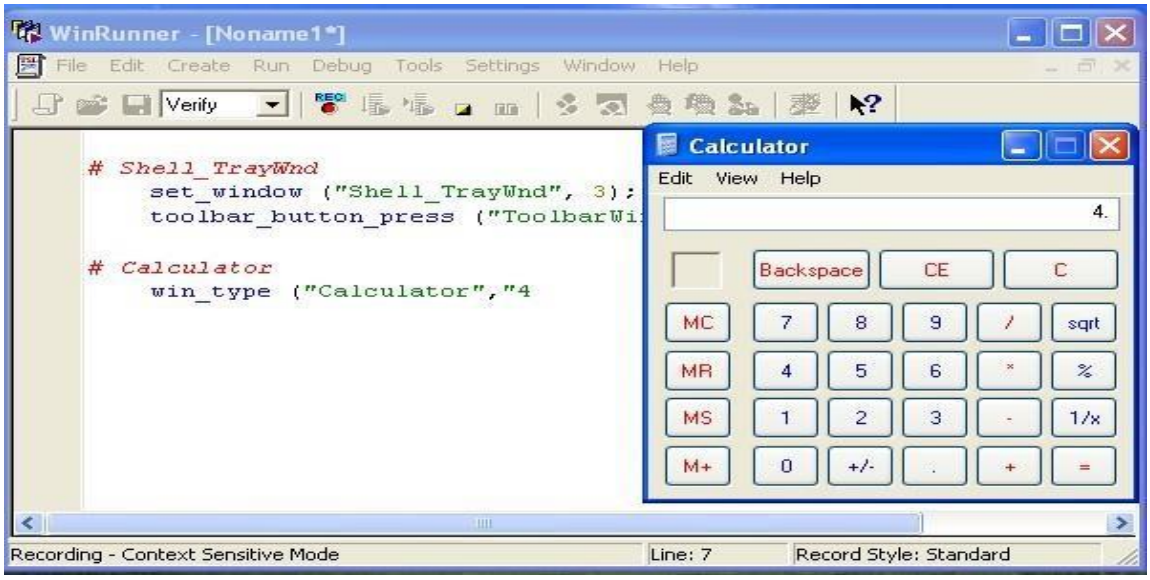
4. Review the results.

Output:

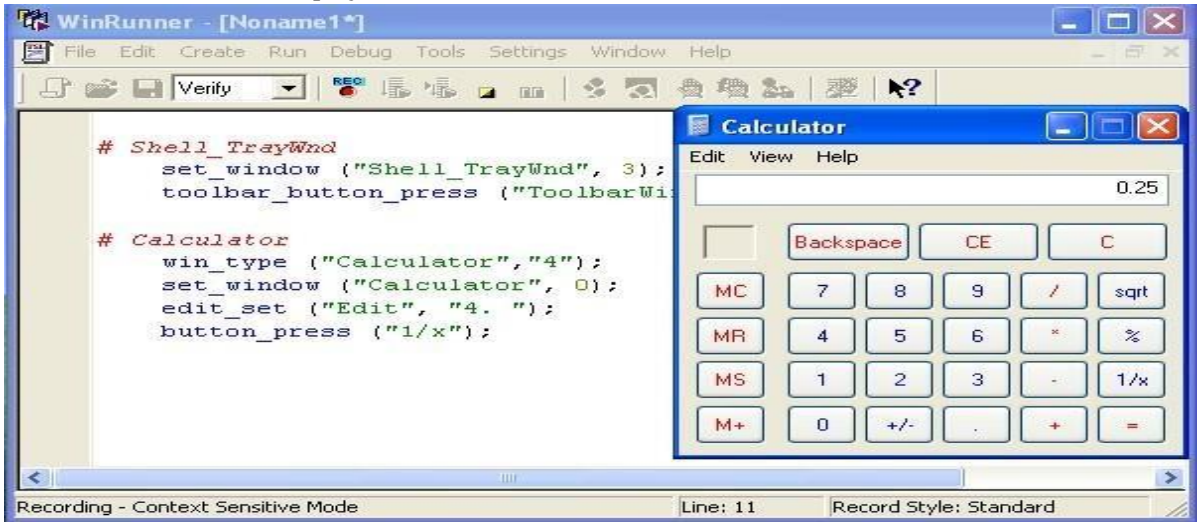
Start recording



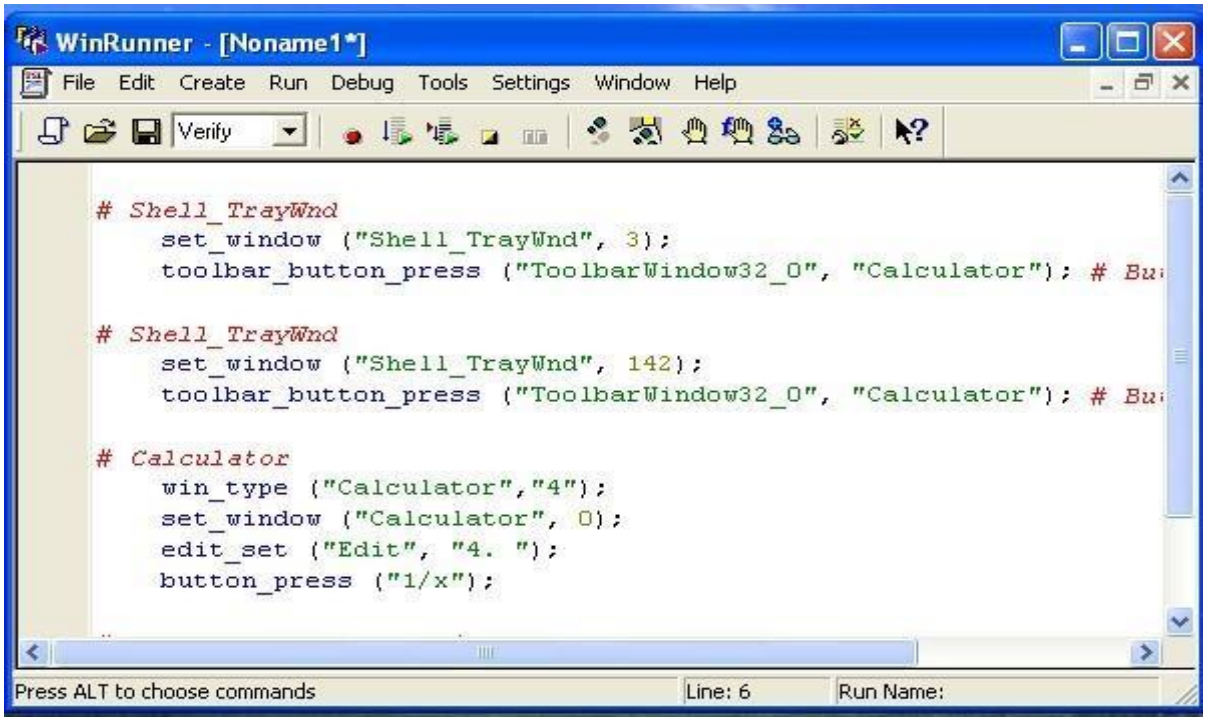
Select the Calculator application and start recording the actions. a Click "4" on the Calculator



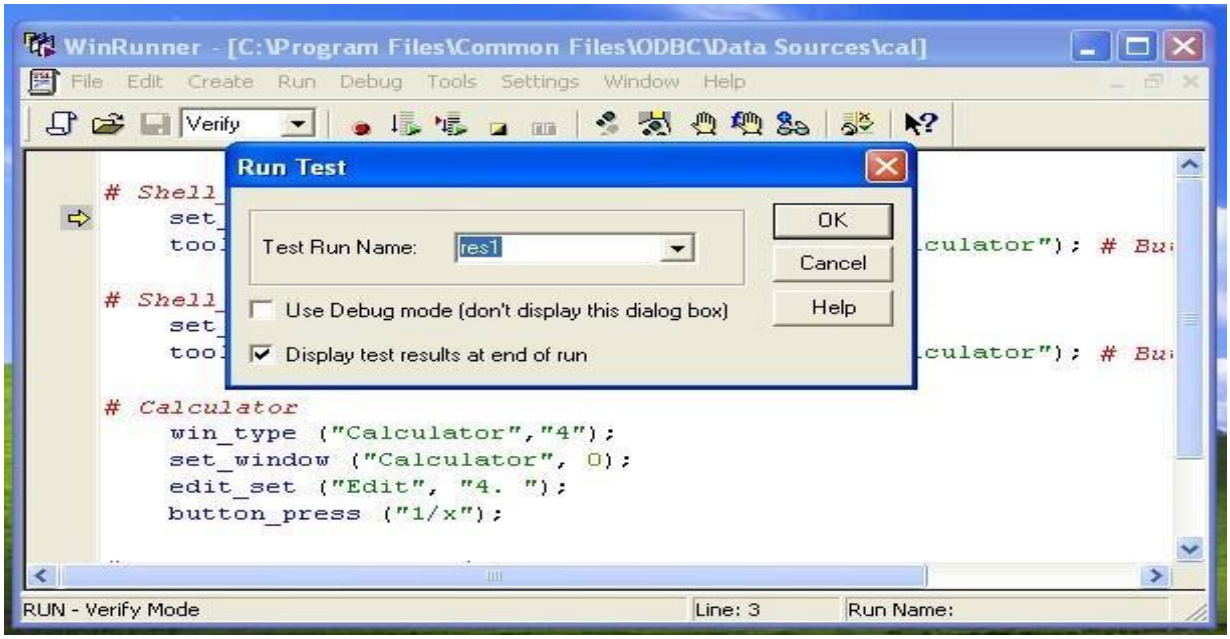
Click the "1/x" button on the Calculator to find the inverse of 4.
The result, 0.25 will be displayed on the Calculator.



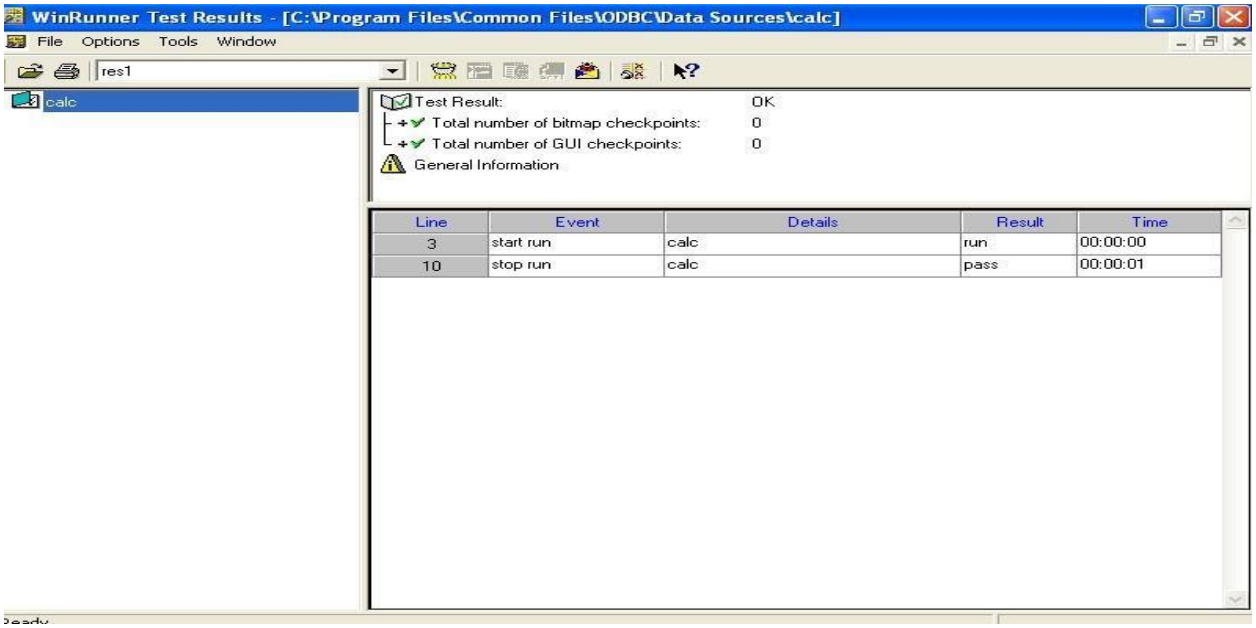
Stop recording and save the test



Run the Test



Test Result



Result: Test case for calculator in windows application are done successfully.