**VARIABLE SWAPPING**

Step 1: Open UiPath Studio.

Step 2: Create a process and name it as “Variable Swapping”

Step 3: Drag a Sequence activity from the Activities panel and drop in the Designer panel.

Step 4: Name the Sequence activity as “Sequence – ‘This code is for swapping two numbers using a third variable’”

Step 5: Insert a Comment activity from the Activities panel within the Sequence activity.

Step 6: Add comment “Taking input of two numbers from the user and swap them by using a third variable.”

Step 7: Drag another Sequence activity from the Activities panel and insert it below the Comment activity.

Step 8: Name the Sequence activity as “Sequence – ‘For prompting the user to give the input'”.

Step 9: Right-click on the Sequence activity container and select *Annotations* from the context menu.

Step 10: Enter an annotation “This code is for swapping two numbers by using a third variable.”

Step 11: Insert an Input Dialog activity within the second Sequence activity and name it as “Input Dialog – ‘First Variable by User’”.

Step 12: Right-click on the Input Dialog activity container and select Annotations from the context menu. Add an annotation : “Taking User input and storing the value in "First\_Input"”.

Step 13: In the Input Dialog activity, enter values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “First Value” | “Please enter the first numeric value: ” |

Step 14: In the Variables panel, create a variable for the above Input Dialog activity as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| First\_Input\_Value | Double | Sequence – ‘This code is for swapping two numbers by using a  third variable’ |  |

Step 15: Go to the Properties panel of the Input Dialog activity and insert

**First\_Input\_Value** in its Output property.

Step 16: Insert a second Input Dialog activity below the previous Input Dialog activity, and name it as “Input Dialog – ‘Second variable by User’”.

Step 17: Right-click on the Input Dialog activity container and select *Annotations* from the context menu. Add an annotation : “Taking User input and storing the value in "Second\_Input\_Value”.

Step 18: In the second Input Dialog activity, enter values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Second Value” | “Please enter the second numeric value: ” |

Step 19: In the Variables panel, create a variable for the second Input Dialog activity as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| Second\_Input\_Value | Double | Sequence – ‘This code is for swapping two numbers by using a third  variable’ |  |

Step 20: Go to the Properties panel of the Input Dialog activity and insert the variable

**Second\_Input\_Value** in its Output property.

Step 21: Insert a Write Line activity from the Activities panel after the second Sequence activity, and name it as “Write Line – ‘Value entered before swapping’”.

Step 22: Right-click on the Write Line activity container and select *Annotations* from the context menu. Add an annotation : “Enter the text to get the result in the Output Panel”.

Step 23: In the text box of the Write Line activity, enter the expression: **“First Value is: ” + First\_Input\_Value.ToString + Environment.NewLine + “Second Value is: ” + Second\_Input\_Value.ToString**

Step 24: Insert another Sequence activity from the Activities panel below the Write Line activity, name it as “Sequence – ‘Swapping of numbers’” and annotate it as “This block of code will swap the values of the numbers entered”.

Step 25: In the Variables panel, create a new variable as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| Swapping\_Support\_Variable | Double | Sequence – ‘This code is for swapping two numbers by using  a third variable’ |  |

Step 26: Insert an Assign activity in the third Sequence activity, name it as “Assign – ‘Move the First\_Input\_Value to Swapping\_Support\_Variable’” and enter the annotation : “Swap Swapping\_Support with First\_Input\_Value”.

Step 27: In the Assign activity, enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| Swapping\_Support\_Variable | First\_Input\_Value |

Step 28: Insert a second Assign activity below the previous Assign activity, name it as “Assign – ‘Move the Second\_Input\_Value to First\_Input\_Value’” and Enter the annotation “Swap First\_Input\_Value with Second\_Input\_Value”.

Step 29: In the second Assign activity, enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| First\_Input\_Value | Second\_Input\_Value |

Step 30: Insert a third Assign activity below the second Assign activity, name it as “Assign – ‘To swap Swapping\_Support\_Variable with Second\_Input\_Value’” and enter annotation: “Swap Second\_Input\_Value with Swapping\_Support”.

Step 31: In the third Assign activity, enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| Second\_Input\_Value | Swapping\_Support\_Variable |

Step 32: Insert a Write Line activity below the third Sequence activity, name it as “Write Line – ‘Swapped Result’” and enter annotation: “Enter the text to get the result in Output Panel”.

Step 33: In the text box of the Write Line activity, enter the expression: **“First Value after swapping is: “ + First\_Input\_Value.ToString + Environment.NewLine + “Second Value after swapping is: “ + Second\_Input\_Value.ToString”**

Step 34: Save and run the workflow

NOTEPAD APPLICATION tw2

Step 1: Open a new Notepad file.

Step 2: Open UiPath Studio.

Step 3: Create a new process and name it as “Input Methods”.

Step 4: Drag and Drop a Sequence activity from the Activities panel into the Designer panel.

Step 5: Name the Sequence activity as “Sequence – ‘This is the code to test the input methods in UiPath’”

Step 6: Right-click on the Sequence activity container and select Annotations from the context menu. Add an annotation "This block of code demonstrates a workflow that uses different Input Methods to input data in a Notepad."

Step 7: Now, manually open a Notepad window in the background.

Step 8: In UiPath Studio, Drag and Drop an Open Application activity within the Sequence activity, rename it to “Open Application - Open Notepad” and Enter annotation: “Open application is used to open the blank notepad file.”.

Step 9: Click on the “Indicate element on screen” link and select the Notepad window.

Step 11: In the Do section of the Use Application/Browser activity, drag and drop a Type Into activity from the Activities panel, and rename it to “Type First Text” and Enter annotation: “Types the first text in notepad file.”.

Step 10: Click on the “Indicate element on screen” link of the Type Into activity and select the editor area of the Notepad.

Step 11: In the text box of the Type Into activity, enter the text “Automation makes life easier”.

Step 12: Drag and drop a Click activity from the Activities panel below the Type Into activity, and rename it to “Click Minimize - Simulate Click”.

Step 13: Right-click on the Click activity container and select *Annotations* from the context menu.

Step 14: Add an annotation “Minimizes the notepad window using SimulateClick.” Step 15: Click on the “Indicate element on screen” of the Click activity, and select the

Minimize button of the Notepad.

Step 16: Drag and Drop another Type Into activity below the Click activity, rename it to “Type Second Text- Send window Messages” and Enter annotation: “Types the second text in Notepad file.”

Step 17: Click on the “Indicate element on screen” link of the Type Into activity and select the editor area of the Notepad file.

Step 18: In the text area of the second Type Into activity, enter the text “Welcome to the new world of automation.”

Step 19: Insert a Keyboard shortcuts activity below the second Type Into activity, rename it to “Send Hotkey - Select all text” and enter annotation: “Ctrl+A selects all the text in the notepad file.”

Step 20: Save and run the workflow.

.

**TW 3.1**

IF ELSE STSEMATET

Step 1: Open UiPath Studio.

Step 2: Create a new process and name it as “If Statement”.

|  |  |
| --- | --- |
| Step 3: | Drag a Sequence activity from the Activities panel and drop it in the Designer panel. |
| Step 4: | Name the Sequence activity as “Sequence – ‘Marshmallow Game’”. |
| Step 5: | Right-click on the Sequence activity container and select Annotations from the context menu. |
| Step 6: | Enter the annotation “This code is to ask the user whether he wants a second  Marshmallow.” |
| Step 7: | Insert an Input Dialog activity within the Sequence activity and name it as “Input Dialog – ‘Question’”. Enter the annotation “Question to User”. |
| Step 8: | In the Input Dialog activity, enter values as shown below: |

|  |  |
| --- | --- |
| Title | Label |
| “Question” | “Do you want to eat your first Marshmallow? Choose among the following options: ” + Environment.NewLine + “1. Now” +  Environment.NewLine + “2. After 5 minutes” |

Step 9: In the Variables panel, create a variable for the above Input Dialog activity as

shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| UserInput | String | Sequence – ‘Marshmallow Game’ |  |

|  |  |
| --- | --- |
| Step 10: | Go to the Properties panel of the Input Dialog activity and insert UserInput in its Output property. |
| Step 11: | Insert an If activity below the Input Dialog activity and name it as “If - To check if the user input is ‘Now’”. Enter annotation: “This activity judges the User Input whether it is "Now", "After 5 minutes" or "Invalid"”. |
| Step 12: | In the condition input area of If activity, enter the expression: UserInput = “Now”. |
| Step 13: | Insert a Message Box activity in the Then section of the If activity and name it as “Message Box - Failed”. Enter annotation: “Prints Fail message”. |
| Step 14: | In the Message Box activity, enter the text “Oops! You will not get the second |

Marshmallow.”

|  |  |
| --- | --- |
| Step 15: | Insert a second If activity, name it as “If - To check User input is 'After 5 minutes'”, add an annotation “Check whether the input contains ‘After 5 minutes’ or Invalid input” within the Else section of the first If activity. |
| Step 16: | In the condition input area of second If activity, enter the expression:  UserInput = “After 5 minutes”. |
| Step 17: | Insert a Message Box activity in the Then section of the second If activity and name it as “Message Box - Success”. Add an annotation: “Prints Success message”. |
| Step 18: | In the Message Box activity, enter the text “Congrats! You will get the second  Marshmallow.” |
| Step 19: | Insert another Message Box activity in the Else section of the second If activity and name it as “Message Box – Invalid Input”. Add an annotation:  “Prints Invalid Input message”. |
| Step 20: | In the Message Box activity, enter the text “Invalid Input” |
| Step 21: | Save and run the workflow. |

TW 3.2

SWITCH STATEMENT

3.2.3 Step by Step Process

|  |  |
| --- | --- |
| Step 1: | Open UiPath Studio. |
| Step 2: | Create a new process and name it as “Switch Activity” |
| Step 3: | Drag a Sequence activity from the Activities panel and drop it in the Designer panel. |
| Step 4: | Name the Sequence activity as “Sequence – ‘Create a Robot that asks user their eye color’” |
| Step 5: | Right-click on the Sequence activity container and select Annotations from the |

context menu.

Step 6: Enter the annotation “This block of code is executed using Switch Activity.”

Step 7: Insert an Input Dialog activity, name it as “Input Dialog – ‘Question’”, Add an

annotation “Question to User” and enter the values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Question” | “Enter the color of your eye:” |

Step 8: In the Variables panel, create a variable for the above Input Dialog activity as

shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| EyeColor | String | Sequence – ‘Create a Robot that asks user their eye color’ |  |

|  |  |
| --- | --- |
| Step 9: | Go to the Properties panel of the Input Dialog activity and insert EyeColor in its Output property. |
| Step 10: | Drag and drop a Switch activity below the Input Dialog activity, name it as “Switch - Eye Color” and Add an annotation “Switch activity compares input with four different cases – Blue, Green, Gray, and Black.” |
| Step 11: | In the Properties panel of the Switch activity ensure that the TypeArgument property is String. |
| Step 12: | In the Expression text area of the Switch activity, enter the variable EyeColor. |
| Step 13: | In the Default section of the Switch activity, insert a Message Box activity and name it as “Message Box - Default”. Add an annotation: “Prints default message” |
| Step 14: | In the text area of the Message Box activity, enter the text “Can’t recognize that color!” |
| Step 15: | Add a new case in the Switch activity by clicking ‘Add new case’ option available. Enter “Blue” in the text area of ‘Case value’. |
| Step 16: | In the ‘Case Blue’ section, add a Message Box activity, and name it as “Message Box - Case Blue”. Add an annotation “Prints Case Blue message” and in the text area enter “You must be very Brave!” |
| Step 17: | Add a new case and enter “Green” in the text area of the ‘Case value’. |

|  |  |
| --- | --- |
| Step 18: | In the ‘Case Green’ section, add a Message Box activity, name it as “Message Box - Case Green”. Add an annotation “Prints Case Green message” and in the text area enter “You must be very Generous!” |
| Step 19: | Add a new case and enter “Gray” in the text area of the ‘Case value’. |
| Step 20: | In the ‘Case Gray’ section, add a Message Box activity, name it as “Message Box - Case Gray”. Add an annotation “Prints Case Gray message” and in the text area enter “You must be very Wise!” |
| Step 21: | Add a new case and enter “Black” in the text area of the ‘Case value’. |
| Step 22: | In the ‘Case Black’ section, add a Message Box activity, and name it as  “Message Box - Case Black”. Add an annotation “Prints Case Black message”. |
| Step 23: | In the text area, enter “You must be very Bold!” |
| Step 24: | Save and Run the project. |

**TW 4.1 GUESS THE GAME**

Step 1: Open UiPath Studio.

Step 2: Create a new process and name it as “Do While Loop”.

|  |  |
| --- | --- |
| Step 3: | Drag a Sequence activity from the Activities panel and drop it in the Designer panel. |
| Step 4: | Name the Sequence activity as “Sequence – ‘Guessing Game”. |
| Step 5: | Right-click on the Sequence activity container and select Annotations from the context menu. |
| Step 6: | Enter the annotation :  “This block of code demonstrates a workflow using a Do While statement for creating a 'Guessing Game' with the following conditions:   1. Generate a random number and prompt the user to input a number. 2. In case of a wrong input, a message is displayed to the user stating, 'Please enter a lesser/greater number’. 3. The loop keeps on running until the input number equals the generated number.” |
| Step 7: | Create variables using Variables panel as shown below: |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| intRandomNo | Int32 | Sequence – Guessing Game | 25 |
| intGuessedNo | Int32 | Sequence – Guessing Game |  |

|  |  |
| --- | --- |
| Step 8: | Insert a Do While activity within the Sequence activity, name it as “Do While - Guessed Number <> Random Number”, add an annotation “The loop iterates until it reaches the given condition”. |
| Step 9: | Set its condition to int intGuessedNo<> intRandomNo |
| Step 10: | Insert an Input Dialog activity within the Do While activity, name it as “Input |

Dialog - Guessed Number”, add an annotation: “Take Guessed Number as User input” and enter values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Number” | “Guess a number” |

Step 11: In the Properties panel of the Input Dialog activity, enter intGuessedNo in the Output property.

|  |  |
| --- | --- |
| Step 12: | Insert an If activity below the Input Dialog activity, name it as “If - User input equals Random Number”, add an annotation: “This activity checks whether the User input is equal to the Random Number or not” and enter the condition  intGuessedNo= intRandomNo |
| Step 13: | In the Then section, insert a Message Box activity and name it as “Message Box - Correct Guess”. Add an annotation: “Prints Correct Guess message”. |
| Step 14: | Enter the text “You Guessed it correct”. |
| Step 15: | Insert another If activity, in the Else section of the first If activity, and enter condition intGuessedNo> intRandomNo. Name it as “If- Guessed number is greater or smaller than Random Number”, add an annotation: “This activity checks whether the user input is greater or smaller than the Random number.” |
| Step 16: | In the Then section, insert a Message Box activity, name it as “Message Box - Try Smaller Number”, add an annotation: “Prints Smaller Number message” and enter the text “Please try a smaller number”. |
| Step 17: | In the Else section, insert a Message Box activity, name it as “Message Box - Try Greater Number”, add an annotation: “Prints Greater Number message” and enter the text “Please try a greater number”. |
| Step 18: | Save and run the workflow. |

**TW 4.2 WHILE LOOP PRIME NUMBER**

Step 1: Open UiPath Studio.

|  |  |
| --- | --- |
| Step 2: | Create a new process and name it as “While Activity”. |
| Step 3: | Drag a Sequence activity from the Activities panel and drop it in the Designer panel. |
| Step 4: | Name the Sequence activity as “Sequence – ‘This is the code to test whether the input is a prime number or not.’” |
| Step 5: | Right-click on the Sequence activity container and select Annotations from the context menu. |
| Step 6: | Enter the annotation : “This block of code demonstrates a workflow using  While loop that tells the user if the input is a prime number or not.” |
| Step 7: | Insert an Input Dialog activity within the Sequence activity, name it as “Input Dialog – ‘To take the input from the user’” and add an annotation “Take User input as a Number”. |
| Step 8: | In the Input Dialog activity, enter values as shown below: |

|  |  |
| --- | --- |
| Title | Label |
| “Number” | “Enter a number” |

Step 9: In the Variables panel, create three variables as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| intNumber | Int32 | Sequence – ‘This is the code to test whether the input is a prime number or not.’ |  |
| intRandom | Int32 | Sequence – ‘This is the code to test whether the input is a prime number or not.’ | 2 |
| intCount | Int32 | Sequence – ‘This is the code to test whether the input is a prime number or not.’ | 0 |

Step 10: Go to the Properties panel of the Input Dialog activity and insert intNumber

in its Output property.

Step 11: Insert a While activity below the Input Dialog activity and name it as “While – ‘To check if the number is a prime number or not’”.

|  |  |
| --- | --- |
| Step 12: | Right-click on the While activity container, and select Annotations from the context menu. |
| Step 13: | Add an annotation “This block of code will check whether the number is prime. If it is, it will increment the value of ' intCount’.” |
| Step 14: | Inside the While activity, enter the condition as intRandom < intNumber |
| Step 15: | In the Body section of the While activity drag and drop a Sequence activity. |
| Step 16: | Rename the Sequence activity to “Sequence – ‘Check the number using ‘If’”. |
| Step 17: | Right-click on the Sequence activity container and select Annotations from the context menu. |
| Step 18: | Add an annotation “In this sequence using 'If' activity, the 'Number' is divided by ' intRandom ' until intRandom = intNumber.” |
| Step 19: | Insert an If activity inside the Sequence activity. |
| Step 20: | Inside the If activity, enter the condition as intNumber Mod intRandom = 0. |
| Step 21: | Inside the Then section of the If activity, insert an Assign activity, and enter |

values as shown below:

|  |  |
| --- | --- |
| To | Value |
| intCount | intCount + 1 |

|  |  |
| --- | --- |
| Step 22: | Change the Assign activity name to “Assign – ‘Increment the value of intCount’”. |
| Step 23: | Right-click on the Assign activity container and select Annotations from the context menu. |
| Step 24: | Add an annotation “Incrementing the value of ‘intCount’ when ‘intNumber is found to be a prime number.” |
| Step 25: | Below the If activity, insert another Assign activity and rename it to “Assign- Incrementing the value of ‘intRandom’. |
| Step 26: | In the Assign activity, enter the values as shown below: |

|  |  |
| --- | --- |
| To | Value |
| intRandom | intRandom +1 |

|  |  |
| --- | --- |
| Step 27: | Right-click on the Assign activity container, and select Annotations from the context menu. |
| Step 28: | Add an annotations "Incrementing the value of ' intRandom ' whenever the loop iterates". |
| Step 29: | Below the While activity, insert an If activity and name it as “If – Print the message”. |
| Step 30: | Right-click on the If activity container, and select Annotations from the context menu. |
| Step 31: | Add an annotation “This block of code will print the message in a message box whether the input is Prime or not.” |
| Step 32: | Inside the If activity, enter the condition intCount >0. |
| Step 33: | In the Then section, insert a Message Box activity and name it as “Message Box - Not a prime number”. Add an annotation “Displays that the number is not a prime.” |
| Step 34: | Enter the text “It is not a prime number.” |
| Step 35: | In the Else section, insert another Message Box activity and name it as “Message Box - Is a prime number”. Add an annotation “Displays that the number is not a prime.” |
| Step 36: | Enter the text “It is a prime number.” |
| Step 37: | Save and run the workflow. |

**TW – 5 DATA MANIPULATION**

Step 1: Open UiPath Studio.

Step 2: Create a new process and name it as “String Manipulations - Practice 1”

Step 3: Drag a Sequence activity from the Activities panel and drop it in the Designer panel.

Step 4: Name the Sequence activity as “Sequence – ‘String Manipulation.’”

Step 5: Right-click on the Sequence activity container and select *Annotations* from the context menu.

Step 6: Enter the annotation : “This block of code demonstrates a workflow that performs string manipulation on a pre-defined Message.”

Step 7: In the Variables panel, create three variables as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| message | String | Sequence - String Manipulation |  |
| study | String | Sequence - String Manipulation |  |
| places | List<String> | Sequence - String Manipulation |  |

Step 8: Drag and drop an Assign activity in the Sequence activity and name it as “Assign- Message”. Add an annotation “Pre-defining the message” and enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| message | "You always wanted to study Automation Training. The materials are available in the following places:  UiPath Blog, UiPath Academy." |

Step 9: Insert another Assign activity and name it as “Assign- Extracting (Automation Training)”.

Step 10: Right-click on the Assign activity container, and select *Annotations* from the context menu.

Step 11: Add an annotation shown below:

“1. Split("."c).First.ToString extracts the first sentence of the String and converts it to a String

2. Substring(message.LastIndexOf("study")) extracts the Substring from "study"”

Step 12: In the Assign activity, enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| study | message.Split("."c).First.ToString.Substring(message.LastInde  xOf("study")) |

Step 13: Insert another Assign activity and name it as “Assign – Extracting (UiPath Blog and UiPath Academy)”.

Step 14: Right-click on Assign activity container, and select *Annotations* from the context menu.

Step 15: Add an annotations as shown below:

“1. message.Split("."c)(1).ToString extracts the second sentence of the String and converts it to a String.

2. Split(":"c).Last.ToString splits the remaining String and keeps only the last part of it.

3. Split(","c).ToList takes each string separated by a comma and adds it as an element in the List variable”.

Step 16: In the Assign activity, enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| places | message.Split("."c)(1).ToString.Split(":"c).Last.ToString.Split(  ","c).ToList |

Step 17: Insert a Message Box activity and right-click on Message Box container, and select *Annotations* from the context menu.

Step 18: Add an annotation “In this expression, String.Join is used to extract each element in the “places” List variable and display them”.

Step 19: In the text area of the Message Box activity enter the expression:

#### String.Format(“{0} from: {1}”, study ,String.Join(";“, places))

Step 20: Save and run the workflow.

**TW 6 DATA MANIPULATION IN TABLE**

Step 1: Open UiPath Studio.

Step 2: Create a new process and name it as “Data Table Manipulations”.

Step 3: Drag a Sequence activity from the Activities panel and drop it in the Designer panel.

Step 4: Name the Sequence activity as “Sequence – ‘A workflow using data table activities to join two library databases’”.

Step 5: Right-click on the Sequence activity container and select *Annotations* from the context menu.

Step 6: Add an annotation : “This block of code demonstrates a workflow using data table activities to join two library databases using matching student ID and display output in a message box.”

Step 7: Drag and drop a Build Data Table activity and name it as “Build Data Table – Users”. Add an annotation: “Data table contains Columns Student ID, Student Name”.

Step 8: Create a variable through Variables panel for the Build Data Table activity as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| dt\_users | DataTable | Sequence - ' A workflow using  data table activities to join two library databases' |  |

Step 9: In the Properties panel of the Build Data Table activity, enter **dt\_users** in the Output property

Step 10: Click on the “Data Table” button and create a Data Table as shown below:

|  |  |
| --- | --- |
| Student ID  (String) | Student Name  (String) |

Step 11: Drag and drop a Build Data Table activity and name it as “Build Data Table – Overdue Books”. Add an annotation: “Datatable contains columns Student ID, Book Name”.

Step 12: Create a variable through Variables panel for the Build Data Table activity as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| dt\_overdueBooks | DataTable | Sequence - ' A workflow  using data table activities to join two library databases' |  |

Step 13: In the Properties panel of the Build Data Table activity, enter

**dt\_overdueBooks** in the Output property.

Step 14: Click on the “DataTable...” button and create a Data Table as shown below:

|  |  |
| --- | --- |
| Student ID  (String) | Book Name  (String) |

Step 15: Drag and drop Join Data Table activity and name it as “Join Data Tables - Join dt\_Users and dt\_overdueBooks Datatable”. Add an annotation: “This activity joins both the data tables resulting in a single datatable”.

Step 16: Click on the “Join Wizard” button.

Step 17: Enter **dt\_users** in Input Data Table 1, and enter **dt\_overdueBooks** in Input Data Table 2 text box.

Step 18: Select the Join Type as Inner, and in the Output Data Table text box, press

**Ctrl + K** to create a new variable called **dt\_borrowedBooks**.

Step 19: In the Column Table 1, enter the column index as ‘0’ (column index of Data Table ‘dt\_users’), in the Column Table 2 enter the column index as ‘0’ (column index of Data Table ‘dt\_overdueBooks’) and press OK.

Step 20: Drag and drop a Remove Data Column activity. Name it as “Remove Data Column of dt\_borrowedBooks”. Add an annotation: “This activity removes the duplicate column of Student ID”. In its Properties panel, enter 2 in ColumnIndex property and enter **dt\_borrowedBooks** in DataTable property.

Step 21: Create a variable through the Variables panel as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| dt\_OuputDataTable | DataTable | Sequence - ' A workflow using data table activities  to join two library |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | databases' |  |

Step 22: Drag and drop a Sort Data Table activity. Name it as “Sort Data Column of dt\_borrowedBooks”. Add an annotation: “This activity sorts the DataTable in an ascending order.” and pass the values in the Properties panel as shown below:

|  |  |
| --- | --- |
| Input DataTable | dt\_borrowedBooks |
| Output DataTable | dt\_OutputDataTable |
| Index | 1 |
| Order | Ascending |

Step 23: Create a variable through the Variables panel as shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| DataTable | String | Sequence - ' a workflow using data table activities to join two  library databases' |  |

Step 24: Drag and drop an Output Data Table activity. Name it as “Output Data Column of dt\_OutputDataTable”., add an annotation: “This activity coverts DataTable's variable type from the datatable to string” and pass the values in the Properties panel as shown below:

|  |  |
| --- | --- |
| Input DataTable | dt\_OutputDataTable |
| Output Text | DataTable |

Step 25: Drag and drop a Message Box activity, name it as “Message Box - Text”, add an annotation: “Prints the String form of DataTable” and in its text area enter the variable **DataTable**.

Step 26: Save and run the workflow.

**TW 7 BASIC RECODING**

* + - * START
      * Open UiPath Studio and create a process. Name it as “basic recording”
      * Add a Sequence activity and name it “Sequence – basic recording”.
      * Open notepad in background.
      * Add a Use application/browser activity in sequence and name it as “Open application- open notepad for recording”.
      * Click on “Indicate application to automate” and click the image file opened. The path will be automatically saved in the Application path.
      * Select the Do section of Use application/browser activity, click on App/Web Recorder button from design tab.
      * Select the typing area on text file as the Target and click on confirm button.
      * A text box is prompted to type the text that has to be typed in text file. Click on confirm after typing the text.
      * After the text is typed into the file, click on Format button from menu and click on Confirm, then click on Font from the dropdown list and click on Confirm. Font window will be opened.
      * Hover the mouse on Font Textbox and choose type into option. Type “Times New Roman. Similarly type “Italic” in Font style and 20 in size.
      * Then click on Ok and confirm it.
      * Now save the recording and return to studio.
      * Reset the changes in notepad and run the workflow. See the output in text file.
      * STOP

**TW 8.1 WEB RECORDING**

* + - * START
      * Open UiPath Studio and create a process. Name it as “web recording”
      * Add a Sequence activity and name it “Sequence – web recording on amazon”.
      * Create two variables emailaddress and password as string type and set the default value with valid id and password.
      * Open a browser and open a webpage using URL [www.amazon.in](http://www.amazon.in).
      * Add a Use application/browser activity in sequence and name it as “Open application- open www.amazon.in for recording”.
      * Click on “Indicate application to automate” and click the webpage opened. The path will be automatically saved in the Application path.
      * Select the Do section of Use application/browser activity, click on App/Web Recorder button from design tab.
      * Click on Sign in button and click on confirm button.
      * A text box is prompted to type the email address. Hover the mouse onto that text box and choose ‘type into’ option.
      * Choose ‘Expression’ type for text and type the variable name “emailaddress” and click on confirm.
      * Click on ‘Continue’ and confirm it.
      * A text box is prompted to type the password. Hover the mouse onto that text box and choose ‘type into’ option.
      * Choose ‘Expression’ type for text and type the variable name “password” and click on confirm.
      * Click on ‘Sign in’ and confirm it.
      * Now, hover the mouse on ‘Search amazon.in’ tab and choose ‘type into’ option.
      * Type ‘decoupage paper+[k(Enter)]’ and confirm it.
      * Click on the first item that is listed and click on ‘Buy now’ button.
      * It will redirect to cart.
      * Now save the recording and return to studio.
      * Reset the changes in amazon.in website and run the workflow. See the output in browser.
      * STOP

**Tw 8.2 TEXT IMAGE SCRAPPING**

* + 1. **Process Overview**

Initial settings:

1. Go to Project -> Settings-> OCR -> DefaultOCREngine.

Set the Run value and Debug value as Uipath Screen OCR.

1. Open browser and type the URL as cloud.uipath.com.

On top left corner, click on Product Launcher-> Admin -> Licenses -> Robot & services. Copy the API key under computer vision.

Go to Project -> Settings-> OCR -> Uipath Screen OCR ->ApiKey

Paste the API key copied from the website into run value and debug value under ApiKey then click on Ok.

* + - * START
      * Open UiPath Studio
      * Add a Sequence activity and name it “Sequence – Screen OCR”
      * Open a text image file in background.
      * Add a Use application/browser activity in sequence and name it as “Open application- Open image file to extract text”.
      * Click on “Indicate application to automate” and click the image file opened. The path will be automatically saved in the Application path.
      * In the Do section of Use application/browser activity, add Get OCR Text activity and name it as “Get OCR text- indicate text in image file and extract it”.
      * Click on “Indicate Element on Screen” and click on the text area on the image.
      * Go to Properties panel of Get OCR Text activity. Right click in the text box for Text under Output and click on create variable and name the variable as ocrtext.
      * Open a notepad file in which the extracted text will be written.
      * Add Set Text activity below Use application/browser activity and name it as “Set Text- Write extracted text to text file”.
      * In Text textbox in Set Text activity, type the variable name ocrtext.
      * Now save the workflow and run it. See the output in text file.
      * STOP

**TW 9 EXCEPTION HANDLING**

|  |  |
| --- | --- |
| Step 1: | Open UiPath Studio. |
| Step 2: | Create a new process and name it as “Try Catch activity” |
| Step 3: | Drag a Sequence activity from the Activities panel and drop it in the Designer panel. |
| Step 4: | Name the Sequence activity as “Sequence – 'A workflow using Try Catch activity to catch certain errors'”. |
| Step 5: | Right-click on the Sequence activity container and select Annotations from the |

context menu.

|  |  |
| --- | --- |
| Step 6: | Enter the annotation shown below:  This block of code demonstrates a workflow using the Try Catch activity to do the following:   1. Take the Name, Gender, and Age as user input. 2. Subtract current year with Age value to get the Year of Birth. 3. Handle an error that occurs due to a reckless user input of incorrect age containing an 11-digit number. 4. Continue the process to display the Name, Gender, and Year of Birth of the user in a message box. |
| Step 7: | Drag and drop a Try Catch activity and name it as “Try Catch - 'User Inputs'”.  Add an annotation “Take input and catch error.” |
| Step 8: | In the Try Section, insert a Sequence activity and name it as “Sequence - ‘User  Inputs’”. Add an annotation “Take inputs from user.” |
| Step 9: | Insert an Input Dialog activity within the Sequence activity, name it as “Input |

Dialog - User Name”, add an annotation: “Name as User Input” and enter the values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Name” | “Enter your Name” |

Step 10: Insert a second Input Dialog activity below the previous “Input Dialog - User Name” activity, and name it as “Input Dialog - User Gender”, add an annotation: “User Gender as User Input” and enter the values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Gender” | “Enter your Gender” |

Step 11: Insert a third Input Dialog activity below the previous “Input Dialog - User Gender” activity, and name it as “Input Dialog - User Age”, add an annotation:

“User Age as User input” and enter the values as shown below:

|  |  |
| --- | --- |
| Title | Label |
| “Age” | “Enter your Age” |

Step 12: In the Variables panel, define five new variables as shown shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Variable type | Scope | Default |
| userName | String | Sequence - ‘A workflow using Try Catch activity to catch certain errors’ |  |
| userGender | String | Sequence - ‘A workflow using Try Catch activity to catch certain errors’ |  |
| intUserAge | Int32 | Sequence - ‘A workflow using Try Catch activity to catch certain errors’ |  |
| intYearOfBirth | Int32 | Sequence - ‘A workflow using Try Catch activity to catch certain errors’ |  |
| inputError | String | Sequence - ‘A workflow using Try Catch activity to catch certain errors’ |  |

|  |  |
| --- | --- |
| Step 13: | In the Properties panel of “Input Dialog – User Name” activity, enter userName in the Output property. |
| Step 14: | In the Properties panel of “Input Dialog – User Gender” activity, enter userGender in the Output property. |
| Step 15: | In the Properties panel of “Input Dialog – User Age” activity, enter intUserAge in the Output property. |
| Step 16: | Insert an Assign activity below the “Input Dialog – User Age” activity and |

name it as “Assign - Year of Birth.” Add an annotation: “Present year - User's year of birth”, and enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| intYearOfBirth | Date.Today.Year() - intUserAge |

Step 17: In the Catches section of the Try Catch Activity, select “System.Exception”

from the dropdown.

Step 18: Insert an Assign activity in the Catches section and name it as “Assign – InputError”. Add an annotation “Store error as a string in a variable.” Enter values as shown below:

|  |  |
| --- | --- |
| To | Value |
| inputError | exception.Message |

|  |  |
| --- | --- |
| Step 19: | Insert a Message Box activity below the Try Catch activity and name it as  “Message Box - User Details”. Add an annotation: “Print all the details” |
| Step 20: | In the text area of the Message Box activity, enter the expression: "Name: " +  userName + vbCr + "Gender: "+ userGender + vbCr + "Year of Birth: "  + intYearOfBirth.ToString + vbCr + "Error: " + inputError |
| Step 21: | Save and run the Project |

**TW 10.1 SENDING EMAIL**

🔹 Step 1: Enable 2-Step Verification (One-Time Only)

1. Open: 👉 https://myaccount.google.com/security

2. Under "Signing in to Google", click 2-Step Verification

3. Set up using your mobile number

4. Once set, it will show "2-Step Verification: ON"

🔹 Step 2: Generate Gmail App Password

1. Go to 👉 https://myaccount.google.com/apppasswords

2. Under Select App → choose Mail

3. Under Select Device → choose Windows Computer (or “Other” and type UiPath)

4. Click Generate

5. Copy the 16-character password shown (remove any spaces)

🔹 Step 3: Create a New Project in UiPath

1. Open UiPath Studio

2. Click Process → Name it GmailSender → Create

🔹 Step 4: Add Required Package

1. Go to Manage Packages (top ribbon)

2. Under Official, search and install:

✅ UiPath.Mail.Activities

🔹 Step 5: Add and Configure Send SMTP Mail Message

1. In your Main.xaml → Add a Sequence

2. Drag in Send SMTP Mail Message activity

3. Configure the properties:

Property Value To : "receiver@gmail.com"

Subject : "Test Mail from UiPath"

Body: "Hi, this is a test mail."

Email: "your\_email@gmail.com"

Password: "your\_16\_char\_app\_password" (in quotes)

Server: "smtp.gmail.com"

Port: 587

SecureConnection :Auto or StartTls

🔹 Step 6: Run

Click Run

Check your Sent Mail in Gmail — the email should appear!

**10.2 ACCESSING EMAIL**

Step 1: Open UiPath and create a flowchart.

Step 2: Add Get IMAP Email List Activity and connect it to start symbol in flowchart.

Step 3: Go to properties panel of Get IMAP Email List activity and set the following properties:

1. Turn off “Use Integration Service”
2. Server: “imap.gmail.com”
3. Port: 993
4. Email: [your\_emailid@gmail.com](mailto:your_emailid@gmail.com)
5. Password: "your\_16\_char\_app\_password" (in quotes)
6. Unread only: True
7. Limit email to First: 5
8. Right click in Output Email list textbox and click on create variable.

Set Var: Mails

Step 4: Add For Each activity below Get IMAP Email List and connect to it.

Step 5: Go to properties panel of For Each activity. Type variable name Mails in textbox of In\*

Property. Change the item name to Mail.

Step 6: Double click on For Each activity. Add Message Box activity in Body section.

Step 7: In Text input type Mail.Subject.ToString.

Step 8. Save and Run the flowchart to see 5 unread emails on message box.