DUMPSBOSS.

UiPath (ADPv1) Automation Developer Professional Exam

UiPath UiPath-ADPv1

Version Demo

Total Demo Questions: 23

Total Premium Questions: 232

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, General Automation Concepts	8
Topic 2, UiPath Studio	151
Topic 3, UiPath Robots	8
Topic 4, UiPath Orchestrator	34
Topic 5, UiPath Automation Cloud	11
Topic 6, UiPath Test Suite	19
Topic 7, Mix Questions	1
Total	232

QUESTION NO: 1

A developer has created a variable of type String called " MyNumbers " and assigned to it the following value: " 1. 2, 3.4, 5. 6 " . What is the resulting data type for the expression MyNumbers.Split(" . " c)(1)?

A. Array of String

B. String

C. Double

D. Int32

ANSWER: B

Explanation:

`String` is the resulting data type. In a UiPath VB.NET expression, `MyNumbers.Split("."c)` calls the `String.Split` method using a single `Char` delimiter: the period character. This method does not convert the values surrounding the delimiters into numeric values. Instead, it separates the original text into an array whose elements are each strings. For the supplied value, the split operation produces text segments such as `"1"`, `" 2, 3"`, and `"4, 5"`, with whitespace and commas retained as part of the respective text segments. The expression `(1)` then indexes that string array and retrieves its second element. Indexing an array returns the type of one array element, so the final expression evaluates to one `String`, not to the array itself. It also remains text even where its contents appear numeric, because neither parsing nor conversion is included in the expression. UiPath expressions use VB.NET syntax and .NET types; see UiPath's VB.NET automation guidance and the .NET String.Split documentation.

QUESTION NO: 2

Which of the following options is true about the types of robot installation?

A. Both the service and the user modes are recommended for running unattended automations.

B. The service mode is the recommended option for running unattended automatons.

C. Both the service and the user modes are recommended for creating and testing automations, and running attended automations.

D. The service mode is the recommended option for creating and testing automations, and running attended automations.

ANSWER: B

Explanation:

The service mode is the recommended option for running unattended automations. A service-mode Robot is installed as a Windows service and can run independently of a particular user's interactive desktop session. This is the appropriate installation model for unattended workloads because it supports running jobs in the background and enables Orchestrator to start jobs without requiring a user to be logged in. Service mode is therefore intended for automation machines that execute scheduled, triggered, or otherwise centrally managed unattended processes.

UiPath distinguishes this from user-mode installation, which is tied to a specific Windows user profile and is designed for interactive, attended use. For unattended production execution, the service installation provides the operating-system integration and session management behavior required by unattended Robots. UiPath's Robot installation documentation describes selecting service mode when the machine is used to run unattended automations, including jobs launched through Orchestrator. For implementation details and the current installation guidance, see UiPath Robot installation documentation and UiPath guidance on service mode versus user mode.

QUESTION NO: 3

A developer is building a process that types data into input fields using the Hardware Events input method. Which property of the Type Into activity should be modified to reduce the pace at which the input string characters are typed into the fields?

A. Delay before

B. Delay between keys

C. Delay after

D. Alter disabled element

ANSWER: B

Explanation:

Delay between keys is the property that controls the interval between individual keystrokes produced by the Type Into activity. Its value is specified in milliseconds, so increasing it makes UiPath pause longer after sending each character. This is especially useful with the Hardware Events input method, which sends keyboard events in a manner similar to physical typing and may require a slower pace when an application needs time to process each keypress. For example, setting this property to 100 causes approximately a 100-millisecond wait between consecutive characters, helping ensure text is entered reliably into applications with slower or more sensitive user interfaces. This setting governs the character-by-character typing rate itself, rather than adding a wait only before or after the entire activity executes. UiPath documents Delay between keys as a common Type Into activity property, with a default delay of 10 milliseconds and an allowed range up to 1,000 milliseconds. See the UiPath activity reference for Type Into and the documentation on input methods.

QUESTION NO: 4

Given a dataiable " dt " with the following header:

" Surname. Address. Zip Code, Given Name, Phone Number.

What is the correct configuration of the Invoke Method activity so that the resulting header will be:

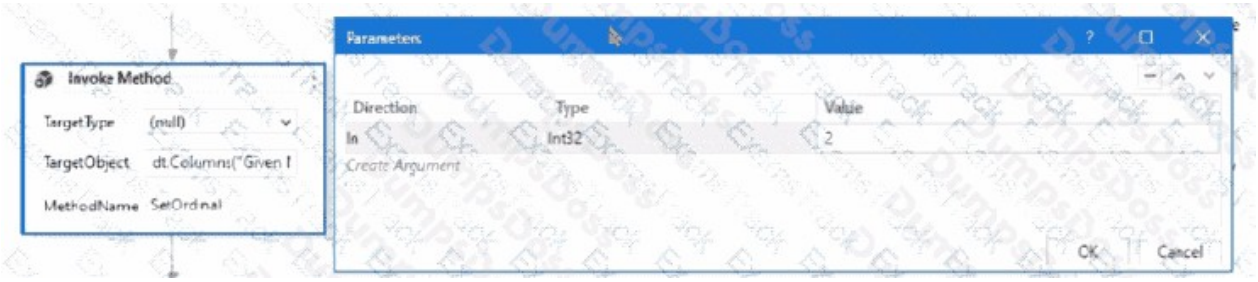
" Surname. Given Name. Address. Zip Code. Phone Number " .

A)

B)

C)

D)



A. Option A

B. Option B

C. Option C

D. Option D

ANSWER: C

Explanation:

The correct configuration invokes the `SetOrdinal` method on the `DataColumn` representing `Given Name`, with an integer argument of `1`. A column ordinal is zero-based: `Surname` initially occupies ordinal zero, so assigning ordinal one places `Given Name` immediately after it. The .NET `DataColumn.SetOrdinal(Int32)` method changes the position of that column in the parent `DataTable.Columns` collection and automatically shifts the intervening columns as necessary. Thus, moving `Given Name` from its original position to ordinal one produces the requested order: `Surname`, `Given Name`, `Address`, `Zip Code`, and `Phone Number`.

In UiPath, Invoke Method is appropriate because it can call public .NET instance methods. The TargetObject should therefore resolve to the selected `DataColumn`, such as `dt.Columns("Given Name")`; the method name is `SetOrdinal`; and the supplied parameter is an `Int32` value of `1`. This uses the standard .NET DataTable API directly. See the DataColumn.SetOrdinal documentation and UiPath's Invoke Method activity documentation.

QUESTION NO: 5

What is the default polling interval set for an event trigger?

A. 1 minute

B. 5 minutes

C. 10 minutes

D. 15 minutes

ANSWER: B

Explanation:

The correct answer is **5 minutes**. In UiPath Integration Service, an event trigger monitors a connected external application for new events, such as a newly created or updated object. For connections that use polling-based event detection, UiPath sets the initial polling interval to five minutes by default. Consequently, the connection is queried at five-minute intervals to determine whether an event has occurred and whether the associated automation should start.

This interval is configured at the connection level, so it applies to the triggers associated with that connection. UiPath provides Adjustable Polling Interval settings that allow authorized users to change the interval when the connector supports it. This lets teams balance event responsiveness against API consumption and the limits imposed by the third-party application. Any adjustment affects the relevant triggers on that connection, making the default five-minute setting the baseline behavior when an event trigger is first configured. UiPath documents this default and the available configuration locations in its Integration Service guidance: Adjustable polling interval. For the broader behavior and setup of event-driven automations, see UiPath Orchestrator triggers documentation.

QUESTION NO: 6

A developer places two branches inside a Parallel activity. Both branches update the same Integer variable to maintain a count of completed operations.

What is the main risk of this design?

A. The Parallel activity automatically serializes updates to variables declared in its parent scope.

B. The variable is converted automatically from Integer to Object when both branches update it.

C. The final count can be unpredictable because concurrent branches can overwrite each other's updates.

D. Only the first branch can access variables declared outside the Parallel activity.

ANSWER: C

Explanation:

The updates can produce an unpredictable result because the branches execute concurrently and access the same shared variable. One branch can read and update the value before, during, or after the other branch performs its own update, creating a race condition.

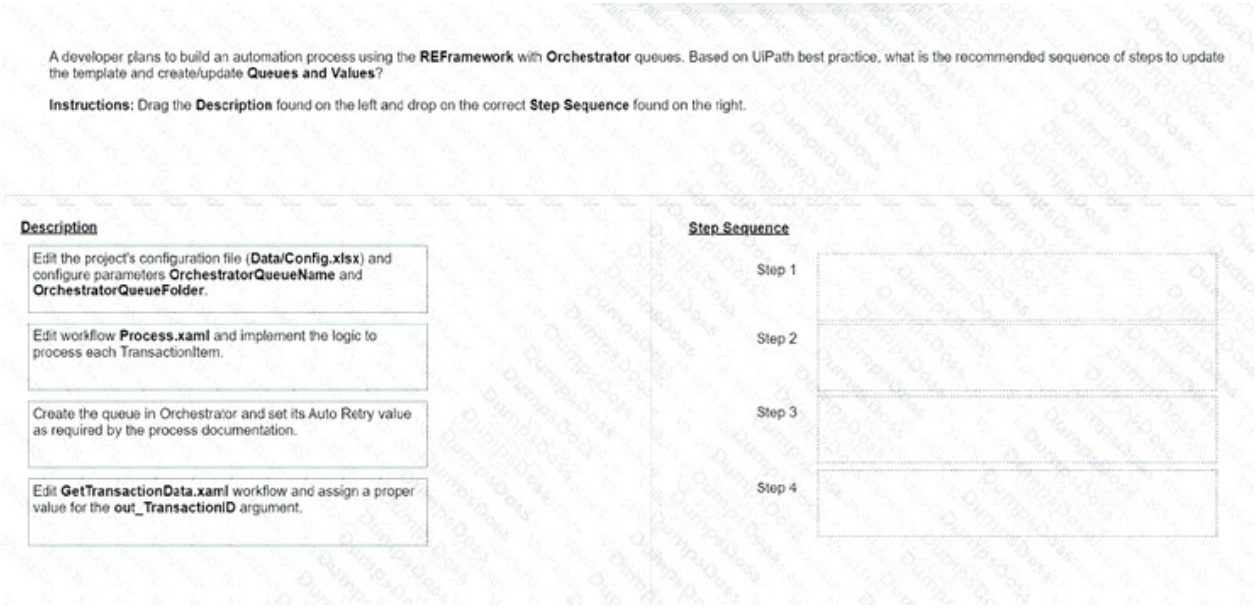
Parallel is appropriate when independent work can execute at the same time. When branches must update shared state, the developer should redesign the workflow to avoid concurrent writes or synchronize access to the shared resource. A variable does not become thread-safe merely because it is defined in the parent workflow scope.

QUESTION NO: 7 - (HOTSPOT)

A developer plans to build an automation process using the REFramework with Orchestrator queues. Based on UiPath best practice, what is the recommended sequence of steps to update

the template and create/update Queues and Values?

Instructions: Drag the Description found on the left and drop on the correct Step Sequence found on the right.



ANSWER:

Explanation:

The correct sequence starts by creating the queue in Orchestrator and setting the Auto Retry value required by the process documentation. This establishes the shared Orchestrator resource and its queue-level behavior before the automation is configured to consume it. Queue settings, including retry behavior, belong to the queue definition and should be deliberately established as part of the process design. UiPath documents queue creation and configuration in its Managing Queues guide.

Next, the REFramework project configuration is updated in **Data/Config.xlsx** with **OrchestratorQueueName** and **OrchestratorQueueFolder**. Keeping these resource identifiers in configuration allows the same reusable framework logic to target the appropriate Orchestrator queue and folder without embedding environment-specific values in workflows. The framework's configuration mechanism is designed to load such settings for use throughout the process.

After the queue connection details are available, **GetTransactionData.xaml** is updated to assign an appropriate value to **out_TransactionID**. In a queue-based REFramework process, this workflow obtains the next transaction item and provides the transaction identity needed by the framework for logging, tracing, and processing context. UiPath describes the framework's transaction-oriented state flow in the REFramework documentation.

Finally, **Process.xaml** receives the business logic that processes each retrieved **TransactionItem**. By this point, the queue exists, its configurable location is known to the project, and transaction retrieval supplies the item context. Implementing the business steps last keeps the solution aligned with REFramework's separation between infrastructure, transaction acquisition, and transaction processing.

QUESTION NO: 8 - (SIMULATION)

What is the correct sequence of steps in a REFramework project that is linked to Orchestrator it an application exception occurs on a Queue Item m the Process Transaction stale?

Instructions: Drag the Description found on the " Left " and drop on the correct Step Sequence found on the Right " .

ANSWER: See the explanation for the answer

Explanation:

Correct step sequence:

For an application exception, REFramework marks the current queue transaction as failed and captures evidence. It then closes/kills the affected application, reinitializes the automation environment, and continues with the next available queue item.

QUESTION NO: 9

What happens when the area selection feature in the UiPath Computer Vision wizard is used?

A. The selected area is treated as a single UI element, with no further analysis of its contents.

B. The selected area is automatically resized to fit all UI elements within it.

C. A duplicated UI can be selected, and the copy is modified in the automation process.

D. A portion of the application UI can be selected, which is helpful when dealing with multiple fields bearing the same label.

ANSWER: D

Explanation:

A portion of the application UI can be selected, which is helpful when dealing with multiple fields bearing the same label. In the Computer Vision wizard, area selection defines a restricted region of the screen in which Computer Vision should identify and interact with the intended target. This gives the automation additional visual context and reduces ambiguity when visually similar controls or repeated labels occur in the same application, such as several input fields next to identical text labels.

The selected region acts as a scope for target detection rather than replacing the target-identification process. At runtime, UiPath Computer Vision analyzes the interface within that indicated region to locate the relevant control based on the activity being configured. This is particularly valuable in remote applications and virtualized environments, where conventional UI selectors can be unavailable or unreliable and the visual layout is the primary source of automation context.

UiPath documents Computer Vision as a UI Automation capability designed to recognize and operate application elements based on what is displayed on screen. The Computer Vision activities and wizard provide the mechanisms to indicate targets and constrain the relevant screen context. See the UiPath Computer Vision documentation and the CV Screen Scope activity documentation for configuration details.

QUESTION NO: 10

How can UIPath Orchestrator help address potential Issues before they become critical problems?

A. By sending customer feedback to UiPath developers.

B. Through proactive monitoring and alerting of detected issues

C. With immediate technical support for any detected Issue.

D. By automatically updating background processes.

ANSWER: B

Explanation:

Through proactive monitoring and alerting of detected issues is correct because UiPath Orchestrator centralizes operational visibility for automations and provides monitoring capabilities that help teams identify abnormal conditions early. Administrators can use alerts to receive notifications about noteworthy events and configured thresholds, such as robot or machine availability changes, process faults, queue-item failures, or license-related events. These notifications make it possible to investigate and remediate an issue while its scope is still limited, rather than discovering it only after a business-critical automation has been significantly disrupted. Orchestrator also exposes monitoring information through dashboards and dedicated monitoring views, enabling teams to track the health and execution status of their automation environment. This operational insight supports timely intervention, such as reviewing job errors, checking infrastructure capacity, or responding to exceptions in a queue. UiPath documents that alerts notify users about important events in Orchestrator, while the Monitoring functionality provides visibility into robots, queues, and processes. Together, these features enable proactive automation operations and help reduce the likelihood that emerging issues become critical incidents. See the UiPath Orchestrator alerts documentation and the UiPath Orchestrator monitoring documentation.

QUESTION NO: 11

When developing a process, you were provided with two data tables, " DT1 " and " DT2 " , as shown below:

DT1

ID	Name
1	James
2	Mary
3	Patricia
4	Robert

DT2

ID	Department
1	Account
2	Science
3	Math
4	Literature

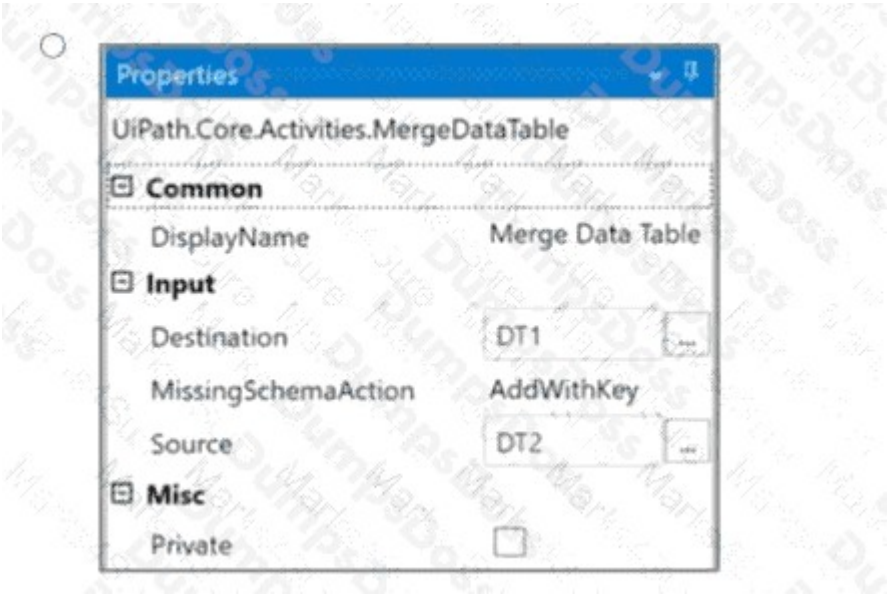
The process documentation specifies that the two data tables need to be manipulated in order to reflect the following " DT2 " :

DT2

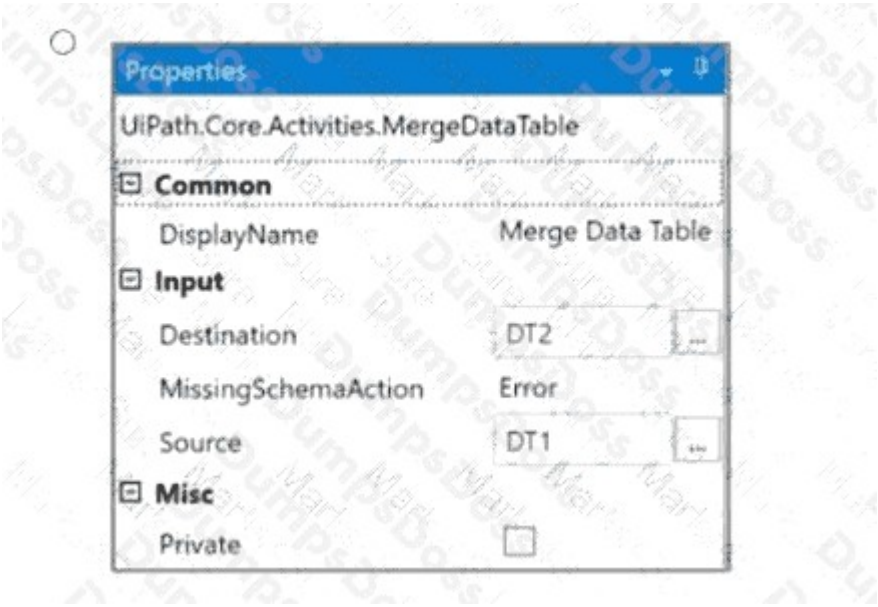
ID	Department	Name
1	Account	
2	Science	
3	Math	
4	Literature	
1		James
2		Mary
3		Patricia
4		Robert

How should the properties of the Merge Data Table activity be configured?

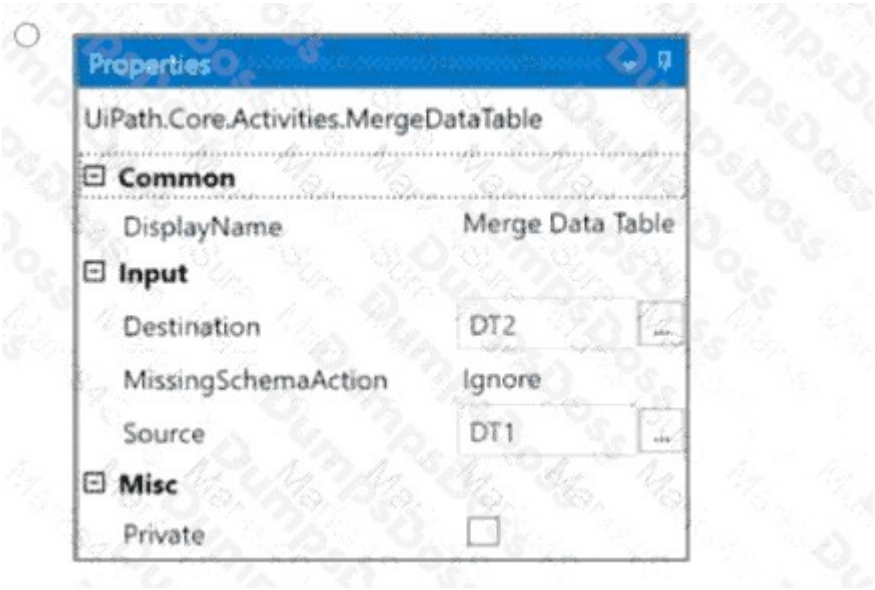
A)



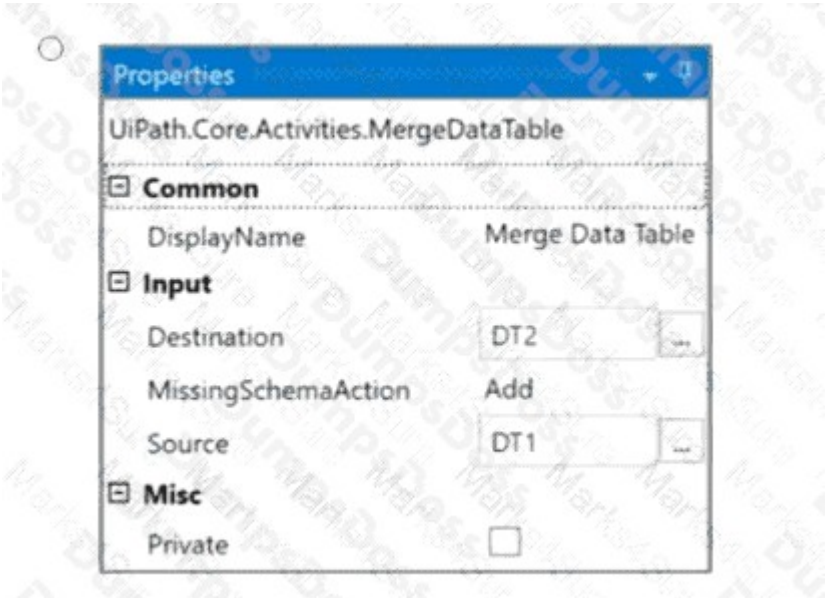
B)


Properties
UiPath.Core.Activities.MergeDataTable
Common
DisplayName
Merge Data Table
Input
Destination
DT2
MissingSchemaAction
Error
Source
DT1
Misc
Private

C)


Properties
UiPath.Core.Activities.MergeDataTable
Common
DisplayName
Merge Data Table
Input
Destination
DT2
MissingSchemaAction
Ignore
Source
DT1
Misc
Private

D)


Properties
UiPath.Core.Activities.MergeDataTable
Common
DisplayName
Merge Data Table
Input
Destination
DT2
MissingSchemaAction
Add
Source
DT1
Misc
Private

A. Option A

B. Option B

C. Option C

D. Option D

ANSWER: B

Explanation:

The correct configuration uses **DT1** as the source and **DT2** as the destination, with **MissingSchemaAction** set to **AddWithKey**. In UiPath, the Merge Data Table activity merges the source DataTable into the destination DataTable; therefore, because the required final table is explicitly DT2, DT2 must be supplied as the destination. DT1 supplies its records and schema information to that destination.

AddWithKey is appropriate when the source contains schema details that are absent from the destination. It adds the required missing columns and preserves key information where available, allowing the destination to represent the combined data structure required by the process. This is important when DT1 and DT2 share an identifier column while each table also contributes distinct fields, such as a name or department field.

The activity is based on the .NET `DataTable.Merge` operation, so the destination table is the table that is modified in place and subsequently used by the workflow. UiPath documents the activity inputs and its schema-handling behavior in the Merge Data Table activity documentation. The underlying merge semantics are also described in the Microsoft DataTable.Merge documentation.

QUESTION NO: 12

What do the percentages from the Test Explorer panel represent?

A. Percent of test data run.

B. Correctness of the code percent.

C. Coverage percent.

D. Passing percent.

ANSWER: D

Explanation:

Passing percent is correct. In UiPath Studio, the Test Explorer panel summarizes the outcome of the test cases included in a test run. Its percentage is the pass rate: the proportion of executed test cases whose result is Passed. This value gives a quick, high-level indication of how successfully the selected tests ran, without requiring the user to inspect every individual test result.

The pass rate is calculated from the number of passed test cases relative to the total relevant test cases in the run, expressed as a percentage. For example, when eight out of ten executed test cases pass, Test Explorer reports an 80% passing percentage. The panel also provides result-oriented information such as test statuses and counts, enabling developers to open failed tests and investigate their execution details. This makes the percentage an execution-quality summary for the test set, rather than a measure of test data consumption or code-analysis output.

UiPath documents Test Explorer as the Studio interface for viewing and managing test cases and their results. See the UiPath Test Explorer documentation and the UiPath Automation Testing overview.

QUESTION NO: 13

A company needs to execute background automation without worrying about managing infrastructure. They want to scale easily and avoid provisioning virtual machines. Based on the information provided, which type of UiPath Automation Cloud™ robot should they use?

A. Hybrid Cloud Robots

B. Serverless Automation Cloud Robots

C. Cloud Robots - VM

D. Physical Servers Robots

ANSWER: B

Explanation:

Serverless Automation Cloud Robots are the appropriate choice because they run unattended, background automations on infrastructure fully managed by UiPath. The organization does not need to create, configure, patch, monitor, or maintain virtual machines. Instead, it configures the serverless machine resources and execution requirements in Automation Cloud, then UiPath provides the compute capacity needed to run the jobs.

This model is especially suitable when the automation does not require an interactive desktop session or a dedicated, customer-managed machine. It reduces operational overhead and supports elastic execution capacity: organizations can allocate serverless units and use them to run background jobs without conventional VM provisioning. This directly meets the stated requirement for cloud-native scaling while avoiding infrastructure management.

UiPath documents Serverless Automation Cloud Robots as UiPath-hosted resources for executing unattended automations, with machine sizing and runtime capacity managed through Automation Cloud. See the UiPath Automation Cloud Robots documentation and the Serverless Automation Cloud Robots documentation.

QUESTION NO: 14

When using Profile Execution to analyze performance, what should you keep in mind about the data generated during debugging versus data generated during production runs?

A. Profiling data is only available during production runs, not debugging sessions.

B. Profiling data from debugging sessions may differ from production runs.

C. Profiling data from debugging sessions will always be the same as production runs.

D. Profiling data is stored only for debugging sessions, not production runs.

ANSWER: B

Explanation:

Profiling data from debugging sessions may differ from production runs. Profile Execution collects timing and execution information for workflow activities, which is useful for identifying areas that take significant time and for investigating performance behavior. However, a debugging session does not run under precisely the same conditions as an unattended or otherwise deployed production execution. Attaching the debugger, collecting diagnostic information, adding breakpoints, stepping through activities, and interacting with the running process can affect timing, resource usage, and the execution path observed in the profile. Input data, machine capacity, application responsiveness, network conditions, and concurrent workload can also differ between a developer's test environment and the production environment. Therefore, debugging profile results are valuable for locating likely bottlenecks and guiding optimization, but they should be interpreted as diagnostic evidence rather than an exact representation of production performance. After making an improvement, validate it with representative data and execution conditions, ideally in a controlled environment that closely resembles deployment. UiPath describes Profile Execution as a tool for viewing execution details and duration information in its Profile Execution documentation; its debugging guidance also explains the distinct diagnostic context of debug runs.

QUESTION NO: 15

Which of the options below is not an action in Test Explorer?

A. Run Selected

B. Run Failed Test Cases

C. Run All In View

D. Run Passed Test Cases

ANSWER: D

Explanation:

Run Passed Test Cases is not an available Test Explorer execution action. Test Explorer is designed to help developers organize, filter, and execute test cases during development. Its run commands include executing all tests currently displayed in the active view, executing selected tests, and rerunning failed test cases. Rerunning failures is particularly useful during debugging because it lets a developer quickly validate a fix without repeating tests that already completed successfully. By contrast, running only tests whose prior result was passed is not supplied as a dedicated Test Explorer command. Test outcomes can be inspected, filtered, and managed in Test Explorer, while test runs and their results can also be tracked through UiPath's testing capabilities. For the current Test Explorer command set and its use in Studio, see the UiPath Test Explorer documentation. Additional information about creating and executing automated tests is available in the UiPath automated testing overview.

QUESTION NO: 16

Following UiPath best practices, which project structure is best-suited for complex processes in UiPath Studio?

A. Sequence

B. Flowchart

C. Global Exception Handler

D. State Machine

ANSWER: B

Explanation:

Flowchart is the best answer because UiPath identifies flowcharts as suitable for more complex business processes. A flowchart provides a visual representation of the process logic and makes it practical to model multiple decision points, alternative paths, loops, and the connections between activities. This structure is particularly useful when the automation does not follow one strictly linear path and when a developer or reviewer needs to understand the overall routing of the process at a glance.

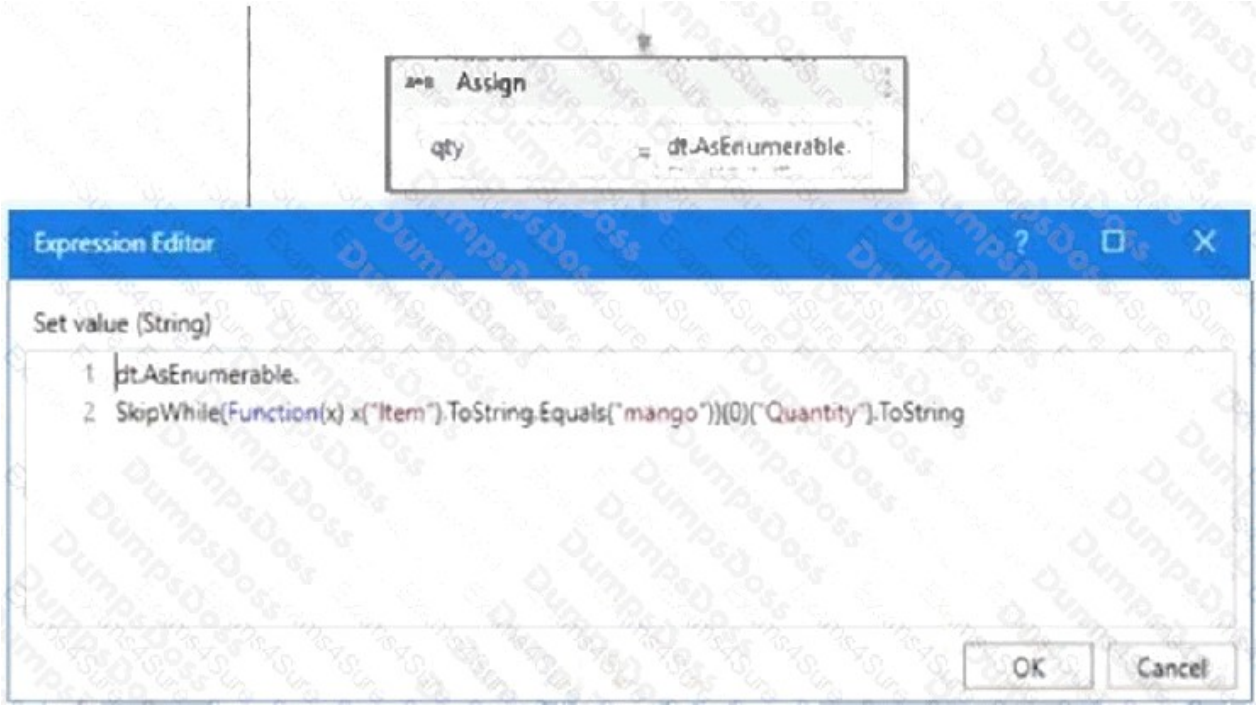
In UiPath Studio, flowcharts use activities such as Flow Decision and Flow Switch to direct execution according to evaluated conditions. This lets an automation clearly express branching business rules while keeping the workflow's paths visible in one design surface. A flowchart can also call other workflows, allowing the complex process to remain organized by placing detailed reusable logic into separate files. UiPath's workflow documentation specifically describes flowcharts as appropriate for complex business processes with multiple branching operators. See the UiPath documentation on workflow types and the UiPath guide to flowcharts.

QUESTION NO: 17

The following table is stored in a variable called " dt " .

	A	B	C
1	No.	Item	Quantity
2		1 apple	10
3		2 orange	20
4		3 mango	5
5		4 kiwi	80
6		5 pear	1

What will the value of the qty variable be after executing the Assign activity?



A. 5

B. 10

C. 80

D. null

ANSWER: C

Explanation:

The value of `qty` is **80**. The Assign activity evaluates the expression on its right side and stores the resulting value in the variable named on its left side. Here, the expression aggregates the relevant numeric values from the `qty` column in the `dt` DataTable. The total produced by that aggregation is 80, so that is the value assigned to `qty`.

In UiPath, an Assign activity is commonly used to calculate and store values, including values returned by .NET DataTable operations. When a DataTable expression uses an aggregate such as `SUM`, the calculation is performed over the specified column and returns the aggregate result. That result must then be compatible with the destination variable type; in this case, the resulting numeric total is assigned to `qty`. UiPath documents Assign as the activity for setting a variable or argument to an evaluated expression, while the underlying .NET `DataTable.Compute` API supports aggregate expressions over DataTable data. See UiPath Assign activity documentation and DataTable.Compute documentation.

QUESTION NO: 18

What is created automatically when you create a coded automation in UiPath?

A. A namespace using the name of the Studio project.

B. A new activity package with the name of the Studio project.

C. A helper class using the name of the Studio project.

D. A folder with the name of the Studio project.

ANSWER: A

Explanation:

A namespace using the name of the Studio project is created automatically. Coded automations use .NET code files and follow the standard .NET practice of grouping related classes under a namespace. UiPath Studio therefore initializes a default namespace based on the project name when a coded automation is created. This namespace provides a consistent logical scope for the coded workflow and any code elements added to the automation, helping avoid naming conflicts and making classes easier to organize and reference as the project grows. The automatically generated namespace is part of the coded-automation project structure; it is not a separate reusable package or a helper class. Developers can work within this namespace when adding coded workflows, coded test cases, or other supported coded automation types, and can subsequently organize their code according to the project's needs. UiPath's documentation on coded automations describes their .NET-based structure and how Studio supports code-first development alongside low-code workflows. See UiPath documentation on coded automations and UiPath documentation on Studio projects.

QUESTION NO: 19

You have to create a testcase for an attended RPA process. At some point, the attended process asks the user to input a specific token for the execution to continue, as shown in the screenshot below.

What testing concept (included in UiPath.Testing.Activities) can be used to isolate and replace the input part, without modifying the original workflow?

A. Application Testing

B. Data-Driven Testing

C. Mock Testing

D. RPA Testing

ANSWER: C

Explanation:

Mock Testing is the appropriate concept because it lets a test isolate a dependency or interaction in the workflow and substitute controlled behavior during execution. In this attended scenario, the dialog that waits for a user to enter a token creates an interactive dependency that would otherwise prevent an unattended, repeatable test run. A mock can provide the expected token or simulated result at the test level, allowing the rest of the process to continue and be validated without editing the production workflow itself.

UiPath Test Suite supports mocking through the UiPath.Testing.Activities package. Mocking is particularly valuable for workflow portions that require user interaction, access external systems, or produce results that are difficult to reproduce consistently. The test can therefore focus on verifying the workflow behavior after the token is supplied, while retaining a stable and deterministic test setup. This separation also helps preserve the original automation's implementation, which is important when the same workflow is used in production.

For UiPath guidance on configuring and using mocked workflow behavior in test cases, see the UiPath documentation on mocking and the UiPath Test Automation overview.

QUESTION NO: 20

How are custom log fields used in the REFramework?

A. Custom log fields are used to automatically retry failed transactions with real-time updates.

B. Custom log fields are used to store sensitive information like credentials and personal data of users.

C. Custom log fields are used to define the variable types for transactions, replacing the default QueueItem type.

D. Custom log fields are included in log messages and used to add more data about each transaction for reporting and troubleshooting purposes.

ANSWER: D

Explanation:

Custom log fields are included in log messages and used to add more data about each transaction for reporting and troubleshooting purposes. In UiPath, the *Add Log Fields* activity adds key-value pairs to the logging context, so the fields are included in subsequent log messages generated by the robot. The REFramework uses this capability to make transaction processing easier to monitor and diagnose by attaching meaningful business context, such as a transaction identifier, transaction reference, business process name, or other non-sensitive details relevant to the automation.

Because the fields travel with the robot's logs, they can be used when reviewing execution data in Orchestrator and in supported external logging systems. This lets teams search, filter, correlate, and analyze log entries based on transaction-specific information rather than relying solely on the log message text. Adding this context is particularly valuable when a process handles many queue items or records, since it helps identify the exact transaction associated with a warning, error, or business exception. Fields should be selected thoughtfully and must not expose secrets or sensitive personal data in logs. UiPath documents the activity and its persistent logging context behavior in the Add Log Fields activity documentation. For the framework's transaction-oriented architecture and operational design, see the REFramework documentation.

QUESTION NO: 21

In order for a developer to utilize the default REFramework without relying on Orchestrator queues, what is the essential prerequisite to ensure that the project does not interact with Orchestrator?

A. Remove the Get Transaction Data state from the Main state machine. Remove the OrchestratorQueueName setting from Config.xlsx & change the variable type.

B. Eliminate the Get Transaction Data state from the Main state machine. Exclude the Get Transaction Item activity from the project & change the variable type.

C. Exclude the Get Transaction Item activity from the project. Eliminate the three SetTransactionStatus activities from the SetTransactionStatus workflow & change the variable type.

D. Omit the OrchestratorQueueName setting from the Config.xlsx file. Exclude the three SetTransactionStatus activities from the SetTransactionStatus workflow & change the variable type.

ANSWER: C

Explanation:

Exclude the Get Transaction Item activity from the project. Eliminate the three SetTransactionStatus activities from the SetTransactionStatus workflow & change the variable type. This is the required adaptation because the default REFramework is designed to process Orchestrator queue items: its transaction item is normally represented by the `QueueItem` type, retrieved through **Get Transaction Item**, and its final state is reported through **Set Transaction Status**. Those activities are the direct integration points that call Orchestrator queue services. When a process instead obtains transactions from a local source, such as a spreadsheet, database, or in-memory collection, the framework must use a transaction-item variable appropriate for that source rather than `QueueItem`. The Get Transaction Data workflow can then supply the local item, while the status-handling workflow should no longer contain the queue-status calls. UiPath documents that Get Transaction Item retrieves the next item from an Orchestrator queue and that Set Transaction Status changes the status of an Orchestrator queue item; removing these queue-specific operations prevents the framework from making those Orchestrator interactions. See UiPath Get Transaction Item documentation and UiPath Set Transaction Status documentation.

QUESTION NO: 22

When a developer runs a process using the REFramework, with the process utilizing Orchestrator queues and a queue already created with the Name provided and the Auto Retry function disabled, which states will be executed without errors?

A. Initialization - > Get Transaction Data - > Process Transaction - > End Process

B. Initialization - > Get Transaction Data - > End Process

C. Initialization - > Process Transaction - > End Process

D. Initialization - > End Process

ANSWER: A

Explanation:

Initialization - > Get Transaction Data - > Process Transaction - > End Process is the standard successful execution path for a queue-based transaction in the Robotic Enterprise Framework. During Initialization, the framework loads configuration and initializes the applications and services required for the run. It then enters Get Transaction Data, where it retrieves the next available queue item from the specified Orchestrator queue. Because the queue exists and its configured name is available to the process, the transaction can be retrieved successfully. The framework then moves to Process Transaction to perform the business steps for that queue item and to set its resulting status. Once the available work has been handled, the state machine transitions to End Process, where cleanup activities are performed. Disabling queue Auto Retry does not prevent this normal path; it simply means that a transaction will not be automatically retried by Orchestrator after it is marked with an application or business exception. UiPath describes the framework state-machine flow and its queue-transaction handling in the Robotic Enterprise Framework documentation and documents queue-item retrieval through Get Transaction Item.

QUESTION NO: 23 - (SIMULATION)

What are the steps to publish a project from UiPath Studio?

Instructions: Drag the Description found on the " Left " and drop on the correct Step Sequence found on the " Right " .

ANSWER: See the explanation for the answer

Explanation:

Correct publishing sequence:

1. In UiPath Studio, open the automation project to be published.
2. On the **Design** tab, select **Publish**.
3. In the **Publish Process** window, select/configure the publishing destination (for example, **Orchestrator Tenant Processes Feed** or a custom NuGet feed) and verify the package details, such as name and version.
4. Click **Publish** to create and upload the project package to the selected location.

After publishing to Orchestrator, the package is available in the selected folder/feed and can then be used to create or update a process.