PassQueen
a.k.a dumpsqueen.com

UiPath Certified Advanced RPA Developer

UiPath UiARD

Version Demo

Total Demo Questions: 7

Total Premium Questions: 77

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, UiPath Studio	58
Topic 2, UiPath Orchestrator	8
Topic 3, Robotic Enterprise Framework (REFramework)	11
Total	77

QUESTION NO: 1

A developer is reviewing a project before publishing it. The project contains an unused package dependency that is no longer required by any workflow.

Where can the developer remove the dependency from the project?

A. Manage Packages

B. Project Settings

C. UI Explorer

D. Workflow Analyzer

ANSWER: A

Explanation:

Manage Packages is correct because it is used to install, update, and remove project dependencies. The developer can open Manage Packages, locate the installed package in the project dependencies, and uninstall the package that is no longer needed.

Removing unused dependencies helps keep the project configuration clean and reduces unnecessary package restoration and deployment content. Studio records package dependencies in the project configuration, so changes made through Manage Packages are reflected when the project is published. See the UiPath Managing Packages documentation.

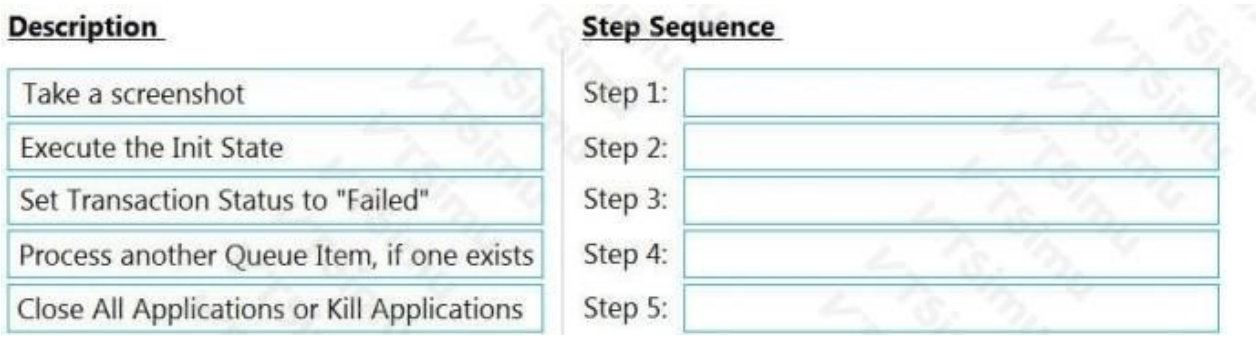
QUESTION NO: 2 - (DRAG DROP)

DRAG DROP

In a Robotic Enterprise (RE) Framework project that is connected to Orchestrator, what is the correct sequence of steps if an application exception occurs on a Queue Item in the Process Transaction state?

NOTE: Drag the Description found on the “Left” and drop on the correct Step Sequence found on the “Right”.

Select and Place:



ANSWER:

Description	Step Sequence	
Take a screenshot	Step 1:	Take a screenshot
Execute the Init State	Step 2:	Set Transaction Status to "Failed"
Set Transaction Status to "Failed"	Step 3:	Close All Applications or Kill Applications
Process another Queue Item, if one exists	Step 4:	Execute the Init State
Close All Applications or Kill Applications	Step 5:	Process another Queue Item, if one exists

Explanation:

When an application exception occurs while a Queue Item is being handled in the Process Transaction state, the recovery path should preserve useful evidence, correctly finalize the current transaction in Orchestrator, restore the application to a known working condition, and then continue processing. First, take a screenshot. This records the application's visible state at the moment of failure, which is particularly helpful for investigating unexpected dialogs, unavailable controls, invalid screens, or other conditions that caused the automation to stop. UiPath documents screenshots as a way to capture visual evidence during automation execution in its Take Screenshot activity documentation.

Next, set the Queue Item transaction status to Failed. This ensures Orchestrator receives the outcome of the item that encountered the exception, along with the relevant error information. Reporting the result is essential because Queue Items must be finalized so their status, retry behavior, and operational reporting remain accurate. UiPath describes this operation in the Set Transaction Status activity documentation.

After the failed status is recorded, close all applications or kill the relevant application processes. This removes the unstable application session and prevents the next transaction from inheriting a broken UI state. The framework can then execute the Init state, which reestablishes the environment and application readiness before work resumes. Finally, if another Queue Item exists, the process proceeds to it. This recovery pattern aligns with the purpose of the Robotic Enterprise Framework: managing transactions consistently while recovering safely from system-level application failures. UiPath's Robotic Enterprise Framework overview explains the framework's transaction-oriented structure and exception handling approach.

QUESTION NO: 3

A process retrieves a Credential asset from Orchestrator by using the Get Credential activity.

What is the data type of the password output?

A. String

B. SecureString

C. Object

D. Char()

ANSWER: B

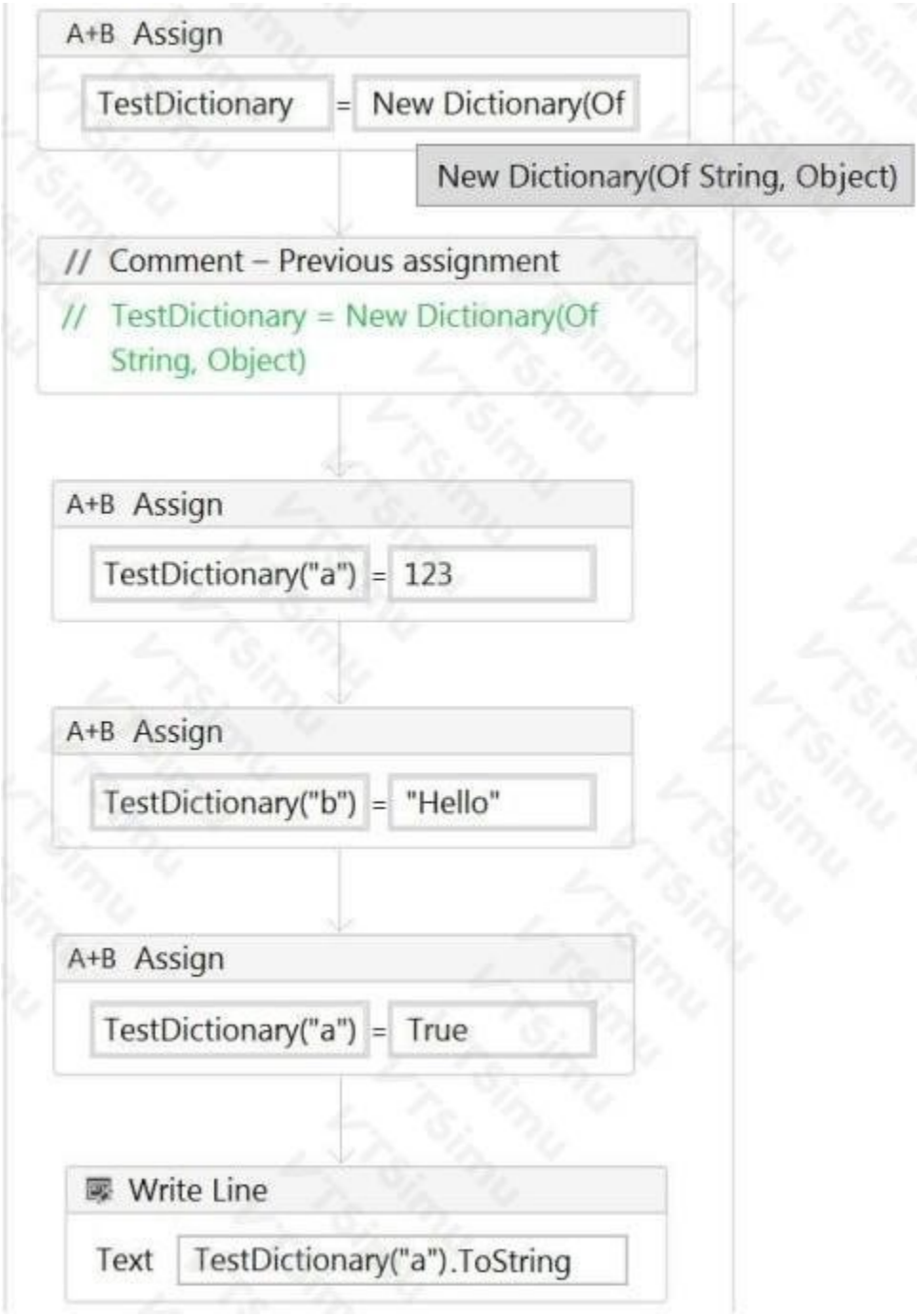
Explanation:

SecureString is correct. The Get Credential activity returns the username as a String and the password as a SecureString. This allows the password to be supplied to compatible activities without converting it to plain text.

Using SecureString helps limit unnecessary exposure of credential values in workflow variables and logs. Developers should avoid converting the value to a String unless a target integration explicitly requires it and appropriate safeguards have been applied. See the Get Credential activity documentation.

QUESTION NO: 4

Review the following exhibit:



Based on the exhibit, what is the result of the Write Line in the sequence?

A. 123

B. True

C. Hello

D. 123True

ANSWER: D

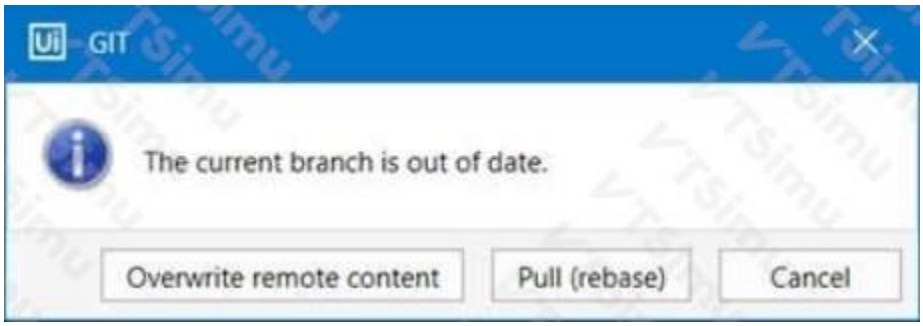
Explanation:

123True is the value written to the Output panel. In the shown expression, the integer value `123` and the Boolean value `True` are combined into one string for the **Write Line** activity. When a non-string value is used in a string-concatenation expression, UiPath/.NET converts it to its textual representation. Therefore, the integer is represented as `123` and the Boolean is represented using its standard .NET text representation, `True`. Because the expression does not include a

separator, space, or line-break character between those values, they appear directly adjacent to each other as `123True`. The Write Line activity then sends that completed string to the Output panel during execution; it does not alter the values or add formatting beyond the line that it writes. This behavior is consistent with UiPath's Write Line activity documentation and with the .NET string-concatenation rules used by UiPath expressions. See UiPath Write Line activity documentation and Microsoft documentation for string concatenation.

QUESTION NO: 5

A developer is using GIT for version control. While the developer is attempting to Commit and Push a local file to the repository, the following pop-up message is displayed:



What is the reason for the pop-up message?

A. Project was not properly checked out in Studio and merge failed into the master.

B. Local repository is not synchronized with the remote one.

C. Opened project was disconnected from the source control.

D. Local version was not connected to any branch.

ANSWER: C

Explanation:

The correct reason is that the opened project was disconnected from source control. UiPath Studio's Git integration operates on a project that is associated with a local Git working copy and its source-control configuration. When that association is removed, unavailable, or the project is opened from a location that is no longer recognized as the connected working copy, Studio cannot use the repository context required for Commit and Push actions. The displayed warning therefore indicates that the project must first be connected or reconnected to its Git repository before changes can be committed and pushed.

Connecting the project restores Studio's access to the repository metadata, including the configured remote repository and the active branch. After the project is connected, the developer can review changes, enter a commit message, commit the selected files locally, and push the resulting commit to the configured remote. UiPath documents Git as a source-control integration that requires opening or connecting a project to a Git repository in Studio. See UiPath: About Git integration and UiPath: Managing Git repositories.

QUESTION NO: 6

A developer retrieves a queue item by using the Get Transaction Item activity.

Which data type should be used for the activity output?

A. QueueItem

B. DataRow

C. DataTable

D. String

ANSWER: A

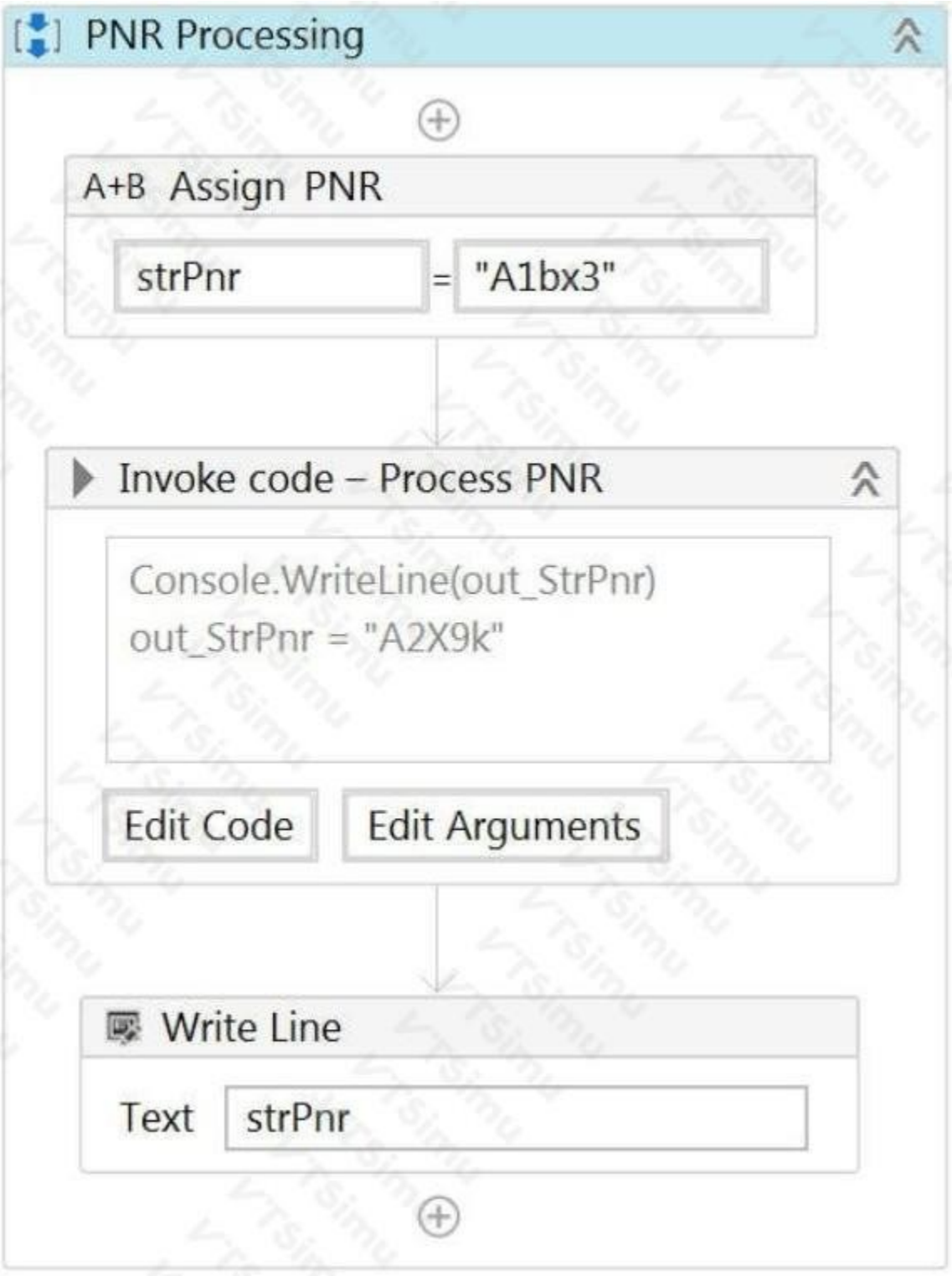
Explanation:

QueueItem is correct. The Get Transaction Item activity retrieves the next available transaction from an Orchestrator queue and returns it as a `UiPath.Core.QueueItem` object. The QueueItem object contains the item reference, priority, specific content, output data, status-related information, and other queue-item properties.

The process can access business data placed in the queue through the QueueItem object's SpecificContent collection. In a queue-based REFramework process, the transaction item variable is commonly of this type. See the UiPath Get Transaction Item activity documentation.

QUESTION NO: 7

Review the following exhibits:


PNR Processing
A+B Assign PNR
strPnr
=
"A1bx3"
Invoke code – Process PNR
Console.WriteLine(out_StrPnr)
out_StrPnr = "A2X9k"
Edit Code
Edit Arguments
Write Line
Text
strPnr


Invoked code arguments
Name
Direction
Type
Value
out_StrPnr
Out
String
strPnr
Create Argument
OK
Cancel

Based on the exhibits, what is the output of the sequence?

A. A2X9k

B. A1bx3
A1bx3

C. A1bx3
A2X9k

D. A2X9k
A2X9k

ANSWER: D

Explanation:

The sequence outputs **A2X9k** twice, on separate lines. The value displayed by each output activity is determined at the moment that activity executes. Based on the assignments and execution flow shown in the exhibits, the relevant variable has the value **A2X9k** when the first output is reached, and it retains that same value when the second output is reached. Therefore, the Output panel contains two consecutive entries with the same text.

UiPath executes activities in the order defined by the workflow sequence unless a control-flow activity changes that order. A Write Line activity evaluates its input expression at runtime and writes the resulting value to the Output panel, making it useful for verifying a variable's current state during execution. Since both Write Line operations evaluate to the same final value here, the expected result is two lines containing **A2X9k**. See UiPath's documentation for Write Line and the overview of Sequences.