APL64: Using ⎕Inbuf

Table of Contents

Summary

The ⎕Inbuf system function may be used to set a queue of APL statements which will be executed when the APL64 interpreter is idle – in the Ready or Suspended state.

The ⎕Inbuf system function is available in the APL64 developer version and the APL64 runtime version.

In the APL64 developer version GUI, the interpreter is in the Ready state immediately after any APL statement is executed by the APL programmer. The Suspended interpreter state occurs when an exception occurs or when a function stop is reached. In the APL64 runtime version, the idle interpreter state occurs when an exception occurs.

The value of ⎕sys[33] affects the action of ⎕Inbuf when the interpreter state is 'Suspended'. If ⎕sys[33] is zero, the actions in the 'InbufBuffer' will be executed when the interpreter state is 'Suspended. If ⎕sys[33] is 1, the 'InbufBuffer' will be cleared when the interpreter state is 'Suspended'.

⎕Inbuf Syntax

The right argument of ⎕Inbuf is used to set or modify the value of an internal APL64 text variable ('InbufBuffer'). The right argument is a text scalar, character vector or scalar string containing one or more APL executable statements separated by ⎕tcnl.

The value of the 'InbufBuffer' is not persisted when the current APL64 developer version instance ends.

⎕Inbuf Left Argument	**Action on Internal 'InbufBuffer' Variable**
1	Append right argument to 'InbufBuffer'

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 1

0	Replace ‘InbufBuffer’with right argument
¯1	Prepend right argument to ‘InbufBuffer’
2	Return the current value of ‘InbufBuffer’ (Right argument, e.g. '', ignored)

When the APL64 Interpreter is Idle (Ready or Suspended State)

When the APL64 is in the Ready or Suspended state, the ‘InbufBuffer’ content is parsed into individual APL executable statements. Statements which have a □tcnl suffix are executed. If the last or only statement in the ‘InbufBuffer’ has no □tcnl suffix, it will not be not executed.

Interpreter Idle State in the APL64 Developer Version

If the interpreter idle state occurs in the APL64 developer version, the statements in the ‘InbufBuffer’ are rendered to this history pane.

The result, if any, of the interpreter execution of those statements which are suffixed by □tcnl, will also be rendered to the history pane.

For a statement which is not executed, because it is not suffixed by □tcnl, the cursor will be placed immediately after such a non-executed statement, so that it can be user-executed with an Enter keystroke. Statements which are not executed are not included in the session history. If the editable classic history is used, a non-executed statement will be rendered to the history pane, otherwise the non-executed statement will replace the Command Line content.

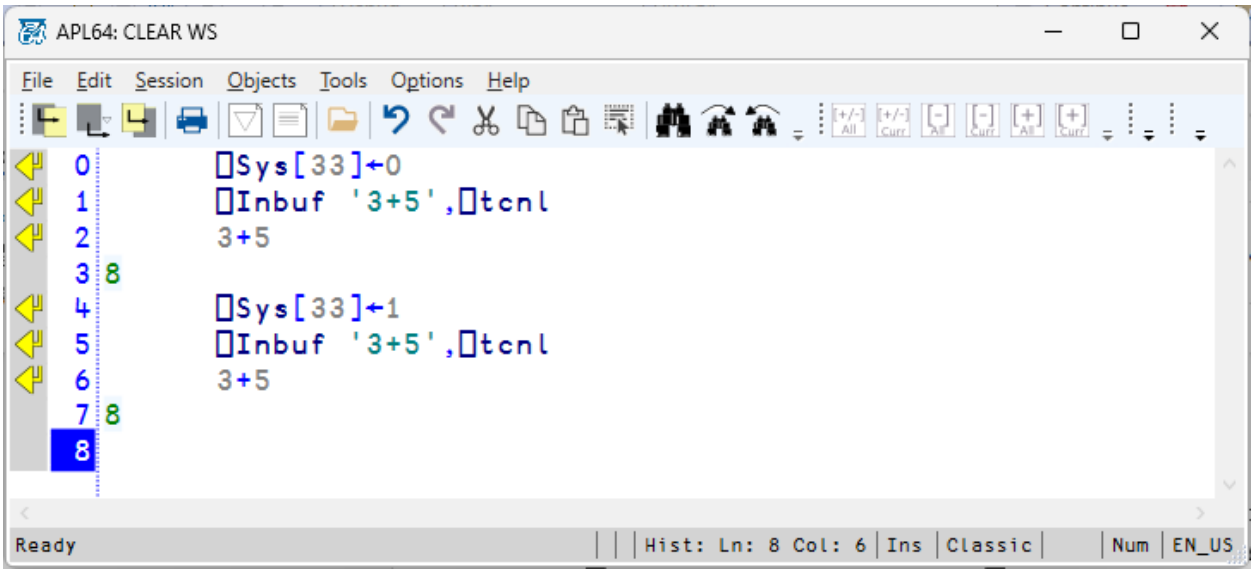
Examples

Example 1: □Inbuf in Immediate Mode

In the APL64 developer version the interpreter is Ready immediately after each user-entered APL statement is executed, so that the content of the ‘InbufBuffer’, if any, is processed at this time. The value of □sys[33] does not affect the action of □Inbuf when the interpreter state is ‘Ready’.

```
□Sys[33]←0
□Inbuf '3+5',□tcnl
□Sys[33]←1
□Inbuf '3+5',□tcnl
```

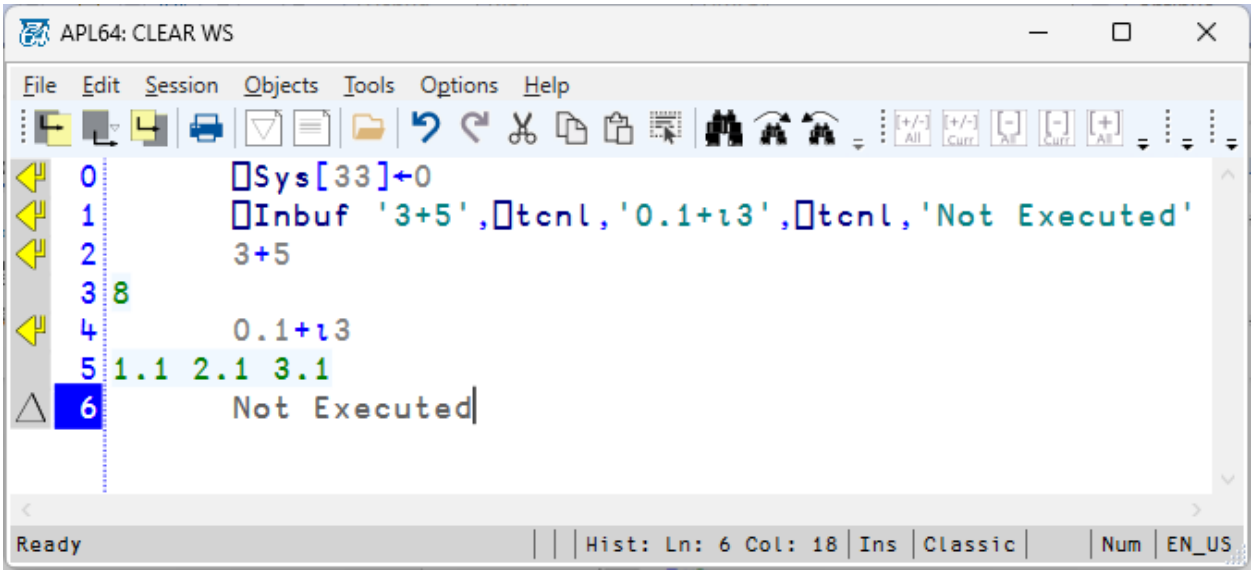
©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 2



Example 2: Multiple APL Statements in 'InbufBuffer'

In the APL64 developer version, all statements in the 'InbufBuffer' will be rendered to the history pane. Statements in the 'InbufBuffer' which have a ⎕tcnl suffix will be executed and their results, if any, will be rendered to the history pane. Statements in the 'InbufBuffer' which do not have a ⎕tcnl suffix will not be executed. The cursor will be placed immediately after a non-executed statement.

```
⎕Sys[33]←0
⎕Inbuf '3+5',⎕tcnl,'0.1+⍳3',⎕tcnl,'Not Executed'
```



Example 3: Interpreter Suspended in a Function

```
S33 FN1 X
⎕Sys[33]←S33
```

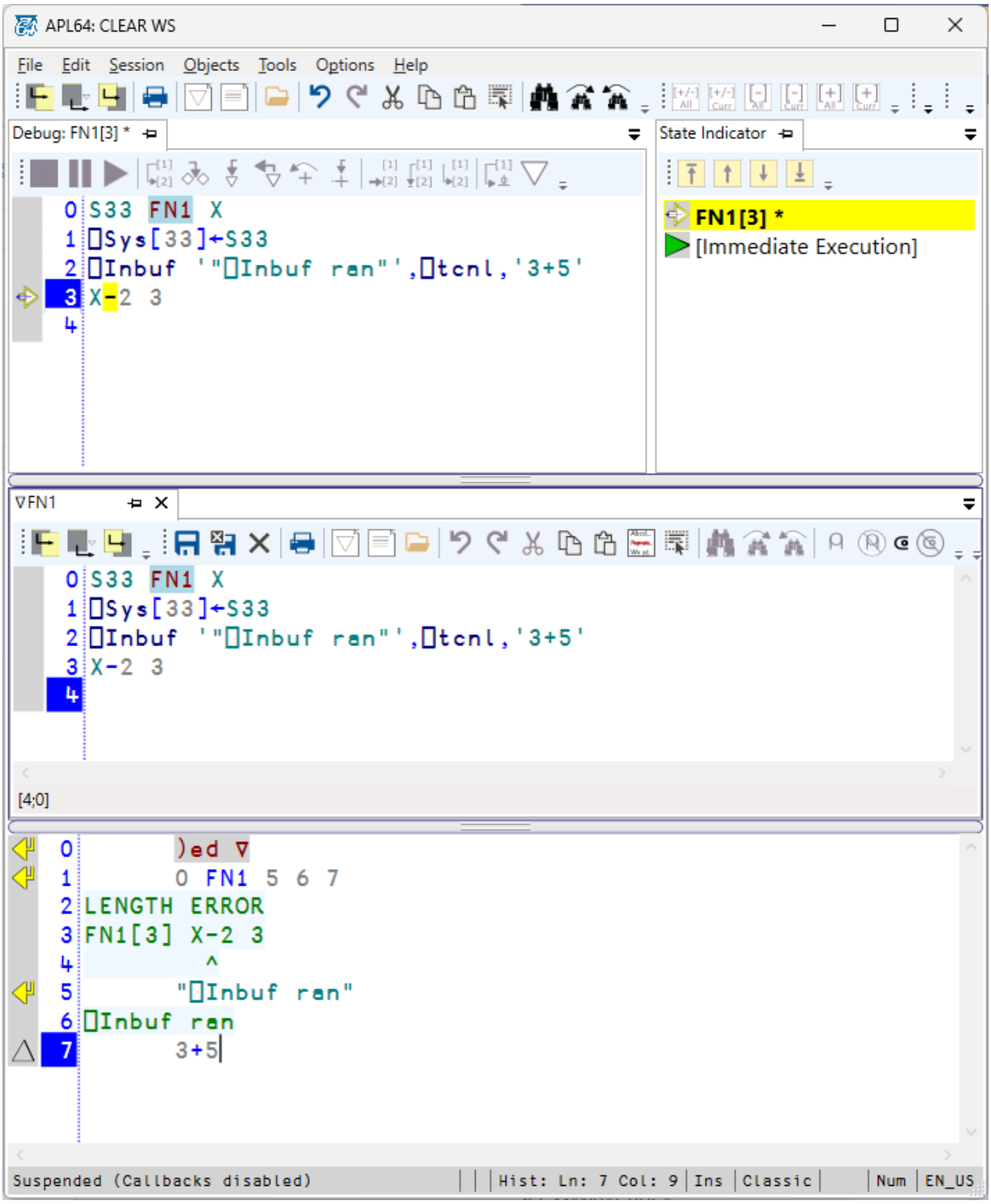
©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 3

```
⎕Inbuf '"⎕Inbuf ran"',⎕tcnl,'3+5'
X-2 3
```

Create the ‘FN1’ function and save it to the current workspace.

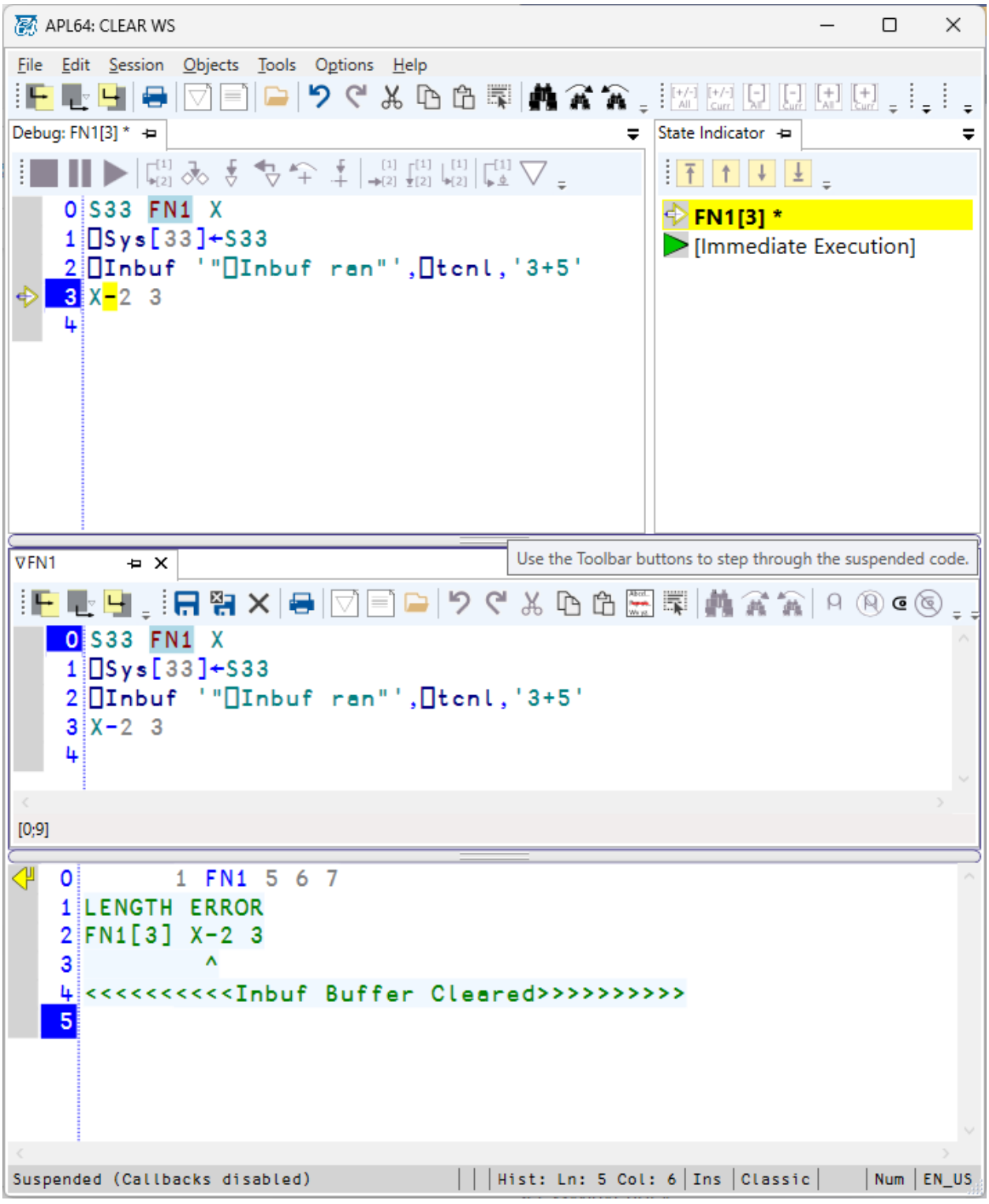
- Execute 0 FN1 5 6 7 [Enter]
  A length error exception will be thrown and illustrated in the debugger pane. The interpreter will be suspended within the function. The first APL statement in the ‘InbufBuffer’ will be executed and it and its result will be rendered to the history pane of the APL64 developer version. The last APL statement in the ‘ExeclIdleBuffer’ will be rendered to the history pane, but this statement will not be executed because it does not have a ⎕tcnl suffix.

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 4



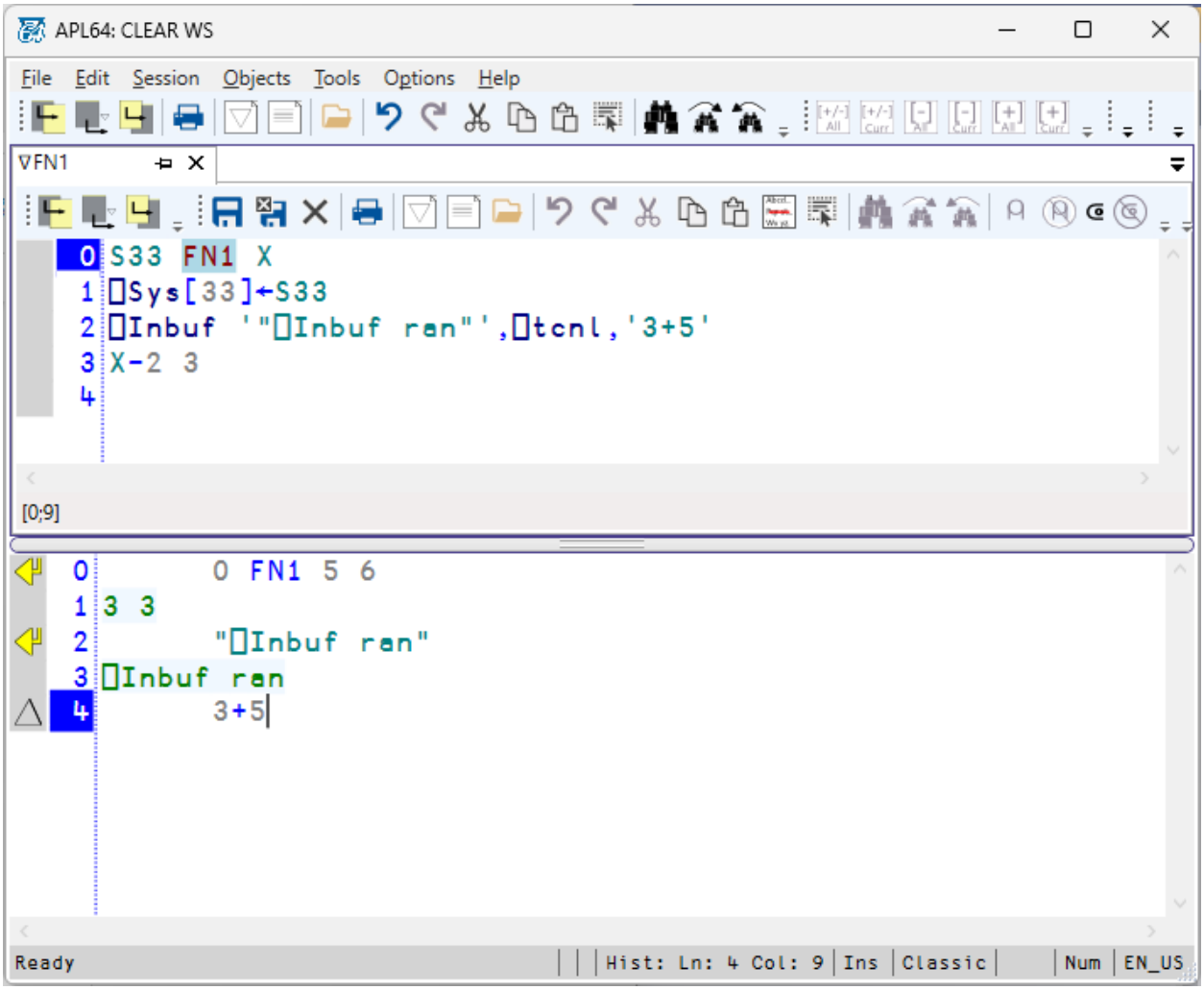
- Execute 1 FN1 5 6 7 [Enter]
  A length error exception will be thrown and illustrated in the debugger pane. The interpreter will be suspended within the function. The 'InbufBuffer' will be cleared, because ⎕Sys[33] is 1.

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 5



- Execute 0 FN1 5 6 [Enter]
  The function will run to completion without an exception. The interpreter will be in the ready state when the function execution is complete. The first APL statement in the 'InbufBuffer' will be executed. The last APL statement in the 'ExecIIdleBuffer' will not be executed.

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 6

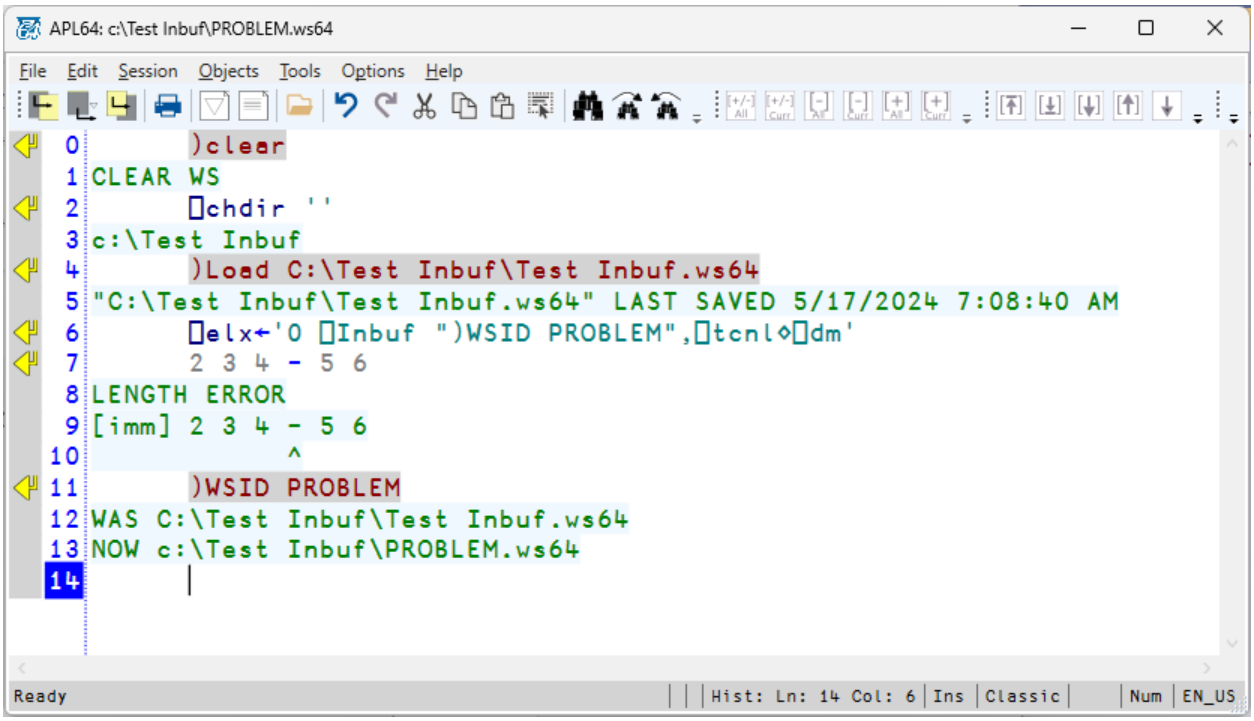


Example 4: Executing a System Command

When an exception occurs, change the workspace name to prevent saving over the original workspace copy with a subsequent ‘)save’ command.

```
⎕elx←'0 ⎕Inbuf ")WSID PROBLEM",⎕tcnl⋄⎕dm'
2 3 4 - 5 6
```

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 7



Compatibility with APL+Win

The ⎕Inbuf system function supports the execution APL statements when the interpreter is idle.

In APL64 ⎕Inbuf and ⎕QQDefault are independent system functions. The ⎕QQDefault system function is exclusively used to set the default response for Quad input and Quote-Quad input which are available only in the developer and not runtime version of APL64.

©APLNow LLC – All Rights Reserved - 2026-07-21 UTC - Pg: 8