**LEXICAL TEST SCRIPT**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | | **Actual Output** | | **Remarks** | | | |
|  | **RESERVED WORDS** | | |  |  | | | |
| Array | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1 | | **OK** | | | |
| Array˽ | Token = Array | | Token = Array | | **OK** | | | |
| Array↵ | Token = Array | | Token = Array | | **OK** | | | |
| Array: | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Array. | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Array( | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Array? | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Array- | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Array+ | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Array, | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Array| | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Array[ | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Array; | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Array on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Arrayy | Lexical Analyzer Encountered 1 error/s.  Please try again.  Invalid Input: Arrayy on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Invalid Input: Arrayy on line 1 | | **OK** | | | |
| Array1 | Lexical Analyzer Encountered 1 error/s.  Please try again.  Invalid Input: Array1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Invalid Input: Array1 on line 1 | | **OK** | | | |
| Boolean | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Boolean on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Boolean on line 1 | | **OK** | | | |
| Boolean˽ | Token = Boolean | | Token = Boolean | | **OK** | | | |
| Boolean↵ | Token = Boolean | | Token = Boolean | | **OK** | | | |
| Boolean: | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Boolean on line 1 | | Lexical Analyzer Encountered 1 error/s.  Please try again.  Proper delimiter expected: Boolean on line 1 | | **OK** | | | |
| Boolean. | Lexical Analyzer Encountered 1 error/s.  Token = Boolean  Invalid Input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Boolean  Invalid Input: . on line 1 | | **OK** | | | |
| Boolean< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Boolean( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Boolean? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Boolean- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Boolean+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Boolean, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Boolean| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Boolean[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Boolean; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Boolean/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Boolean on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Booleann | Lexical Analyzer Encountered 1 error/s.  Invalid input: Booleann on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Booleann on line 1 | | **OK** | | | |
| Boolean1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Boolean1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Boolean1 on line 1 | | **OK** | | | |
| by | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1 | | **OK** | | | |
| by˽ | Token = by | | Token = by | | **OK** | | | |
| by↵ | Token = by | | Token = by | | **OK** | | | |
| by: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: : on line 1 | | **OK** | | | |
| by. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: . on line 1 | | **OK** | | | |
| by( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| by? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| by- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: - on line 1 | | **OK** | | | |
| by+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: + on line 1 | | **OK** | | | |
| by, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: , on line 1 | | **OK** | | | |
| by| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: | on line 1 | | **OK** | | | |
| by[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| by; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: by on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| byy | Lexical Analyzer Encountered 1 error/s.  Invalid input: byy on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: byy on line 1 | | **OK** | | | |
| by1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: by1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: by1 on line 1 | | **OK** | | | |
| Char | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1 | | **OK** | | | |
| Char˽ | Token = Char | | Token = Char | | **OK** | | | |
| Char↵ | Token = Char | | Token = Char | | **OK** | | | |
| Char: | Lexical Analyzer Encountered 1 error/s.  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: : on line 1 | | **OK** | | | |
| Char. | Lexical Analyzer Encountered 1 error/s.  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: . on line 1 | | **OK** | | | |
| Char< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Char( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Char? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Char- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Char+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Char, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Char| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Char[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Char; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Char/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Char on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Charr | Lexical Analyzer Encountered 1 error/s.  Invalid input: Charr on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Charr on line 1 | | **OK** | | | |
| Char1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Char1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Char1 on line 1 | | **OK** | | | |
| Clear | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1 | | **OK** | | | |
| Clear˽ | Token = Clear | | Token = Clear | | **OK** | | | |
| Clear↵ | Token = Clear | | Token = Clear | | **OK** | | | |
| Clear: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Clear. | Lexical Analyzer Encountered 1 error/s.  Invalid input: .on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: .on line 1 | | **OK** | | | |
| Clear< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Clear( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Clear? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Clear- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Clear+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Clear, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Clear| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Clear[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Clear; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Clear/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Clear on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Clearr | Lexical Analyzer Encountered 1 error/s.  Invalid input: Clearr on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Clearr on line 1 | | **OK** | | | |
| Clear1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Clear1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Clear1 on line 1 | | **OK** | | | |
| Default | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1 | | **OK** | | | |
| Default˽ | Token = Default | | Token = Default | | **OK** | | | |
| Default↵ | Token = Default | | Token = Default | | **OK** | | | |
| Default: | Lexical Analyzer Encountered 1 error/s.  Token = Default Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Default Invalid input: : on line 1 | | **OK** | | | |
| Default. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Default( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Default? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Default- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Default+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Default, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Default| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Defaulton line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Defaulton line 1  Invalid input: | on line 1 | | **OK** | | | |
| Default[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Default; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Default/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Default on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Defaultt | Lexical Analyzer Encountered 1 error/s.  Invalid input: Defaultt on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Defaultt on line 1 | | **OK** | | | |
| Default1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Default1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Default1 on line 1 | | **OK** | | | |
| Do | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1 | | **OK** | | | |
| Do˽ | Token = Do | | Token = Do | | **OK** | | | |
| Do↵ | Token = Do | | Token = Do | | **OK** | | | |
| Do: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Do. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Do( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Do? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Do- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Do+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Do, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Do| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Do[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Do; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Do on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Doo | Lexical Analyzer Encountered 1 error/s.  Invalid input: Doo on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Doo on line 1 | | **OK** | | | |
| Do1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Do1on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Do1on line 1 | | **OK** | | | |
| Double | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1 | | **OK** | | | |
| Double˽ | Token = Double | | Token = Double | | **OK** | | | |
| Double↵ | Token = Double | | Token = Double | | **OK** | | | |
| Double: | Lexical Analyzer Encountered 1 error/s.  Token = Double Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Double Invalid input: : on line 1 | | **OK** | | | |
| Double. | Lexical Analyzer Encountered 1 error/s.  Token = Double Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Double Invalid input: . on line 1 | | **OK** | | | |
| Double< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Double( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Double? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Double- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Double+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Double, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Double| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Double[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Double; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Double/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Double on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Doublee | Lexical Analyzer Encountered 1 error/s.  Invalid input: Doublee on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Doublee on line 1 | | **OK** | | | |
| Double1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Double1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Double1 on line 1 | | **OK** | | | |
| End | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1 | | **OK** | | | |
| End˽ | Token = End | | Token = End | | **OK** | | | |
| End↵ | Token = End | | Token = End | | **OK** | | | |
| End: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: : on line 1 | | **OK** | | | |
| End. | Lexical Analyzer Encountered 1 error/s.  Token = End Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = End  Invalid input: . on line 1 | | **OK** | | | |
| End< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: < on line 1 | | **OK** | | | |
| End( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| End? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| End- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: - on line 1 | | **OK** | | | |
| End+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: + on line 1 | | **OK** | | | |
| End, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: , on line 1 | | **OK** | | | |
| End| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: | on line 1 | | **OK** | | | |
| End[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| End; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| End/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: End on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Endd | Lexical Analyzer Encountered 1 error/s.  Invalid input: Endd on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Endd on line 1 | | **OK** | | | |
| End1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: End1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: End1 on line 1 | | **OK** | | | |
| EndIf | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1 | | **OK** | | | |
| EndIf˽ | Token = EndIf | | Token = EndIf | | **OK** | | | |
| EndIf↵ | Token = EndIf | | Token = EndIf | | **OK** | | | |
| EndIf: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: : on line 1 | | **OK** | | | |
| EndIf. | Lexical Analyzer Encountered 1 error/s.  Token = EndIfInvalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = EndIfInvalid input: . on line 1 | | **OK** | | | |
| EndIf< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: < on line 1 | | **OK** | | | |
| EndIf( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ( on line 1 | | **OK** | | | |
| EndIf? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ? on line 1 | | **OK** | | | |
| EndIf- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: - on line 1 | | **OK** | | | |
| EndIf+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: + on line 1 | | **OK** | | | |
| EndIf, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: , on line 1 | | **OK** | | | |
| EndIf| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: | on line 1 | | **OK** | | | |
| EndIf[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: [ on line 1 | | **OK** | | | |
| EndIf; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: ; on line 1 | | **OK** | | | |
| EndIf/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: EndIfon line 1  Invalid input: / on line 1 | | **OK** | | | |
| EndIff | Lexical Analyzer Encountered 1 error/s.  Invalid input: EndIff on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: EndIff on line 1 | | **OK** | | | |
| EndIf1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: EndIf1on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: EndIf1on line 1 | | **OK** | | | |
| For | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1 | | **OK** | | | |
| For˽ | Lexical Analyzer Encountered 1 error/s.  Token = For | | Lexical Analyzer Encountered 1 error/s.  Token = For | | **OK** | | | |
| For↵ | Token = For | | Token = For | | **OK** | | | |
| For( | Lexical Analyzer Encountered 1 error/s.  Token = For  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = For  Invalid input: ( on line 1 | | **OK** | | | |
| For: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: : on line 1 | | **OK** | | | |
| For. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: . on line 1 | | **OK** | | | |
| For? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| For/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: / on line 1 | | **OK** | | | |
| For< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: < on line 1 | | **OK** | | | |
| For+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: + on line 1 | | **OK** | | | |
| For, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: , on line 1 | | **OK** | | | |
| For| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: | on line 1 | | **OK** | | | |
| For[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| For; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: For on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Forr | Lexical Analyzer Encountered 1 error/s.  Invalid input: Forr on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Forr on line 1 | | **OK** | | | |
| For1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: For1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: For1 on line 1 | | **OK** | | | |
| If | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1 | | **OK** | | | |
| If˽ | Token = If | | Token = If | | **OK** | | | |
| If↵ | Token = If | | Token = If | | **OK** | | | |
| If( | Lexical Analyzer Encountered 1 error/s.  Token = If  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = If  Invalid input: ( on line 1 | | **OK** | | | |
| If: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: : on line 1 | | **OK** | | | |
| If. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: . on line 1 | | **OK** | | | |
| If? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| If/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: / on line 1 | | **OK** | | | |
| If< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: < on line 1 | | **OK** | | | |
| If+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: + on line 1 | | **OK** | | | |
| If, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: , on line 1 | | **OK** | | | |
| If| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: | on line 1 | | **OK** | | | |
| If[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| If; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: If on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Iff | Lexical Analyzer Encountered 1 error/s.  Invalid input: Iff on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Iff on line 1 | | **OK** | | | |
| If1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: If1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: If1 on line 1 | | **OK** | | | |
| Int | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Int on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Int on line 1 | | **OK** | | | |
| Int˽ | Token = Int | | Token = Int | | **OK** | | | |
| Int↵ | Token = Int | | Token = Int | | **OK** | | | |
| Int: | Lexical Analyzer Encountered 1 error/s.  Token = Int  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Int  Invalid input: < on line 1 | | **OK** | | | |
| Int. | Lexical Analyzer Encountered 1 error/s.  Token = Int  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Int  Invalid input: . on line 1 | | **OK** | | | |
| Int< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: < on line 1 | | **OK** | | | |
| Int( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ( online 1 | | **OK** | | | |
| Int? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Int- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: - on line 1 | | **OK** | | | |
| Int+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: + on line 1 | | **OK** | | | |
| Int, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: , on line 1 | | **OK** | | | |
| Int| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: | on line 1 | | **OK** | | | |
| Int[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Int; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Int/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Inton line 1  Invalid input: / on line 1 | | **OK** | | | |
| Intt | Lexical Analyzer Encountered 1 error/s.  Invalid input: Intt on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Intt on line 1 | | **OK** | | | |
| Int1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Int1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Int1 on line 1 | | **OK** | | | |
| is | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1 | | **OK** | | | |
| is˽ | Token = is | | Token = is | | **OK** | | | |
| is↵ | Token = is | | Token = is | | **OK** | | | |
| is: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: : on line 1 | | **OK** | | | |
| is. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: . on line 1 | | **OK** | | | |
| is( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| is? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| is- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: - on line 1 | | **OK** | | | |
| is+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: + on line 1 | | **OK** | | | |
| is, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: , on line 1 | | **OK** | | | |
| is| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: | on line 1 | | **OK** | | | |
| is[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| is; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: is on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| iss | Lexical Analyzer Encountered 1 error/s.  Invalid input: :iss line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: :iss line 1 | | **OK** | | | |
| is1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: : is1 line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: : is1 line 1 | | **OK** | | | |
| Lead | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1 | | **OK** | | | |
| Lead˽ | Token = Lead | | Token = Lead | | **OK** | | | |
| Lead↵ | Token = Lead | | Token = Lead | | **OK** | | | |
| Lead: | Lexical Analyzer Encountered 1 error/s.  Token = Lead  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Lead  Invalid input: : on line 1 | | **OK** | | | |
| Lead. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Lead( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Lead? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Lead- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Lead+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Lead, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Lead| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Lead[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Lead; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Lead< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Lead on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Leadd | Lexical Analyzer Encountered 1 error/s.  Invalid input: Leadd on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Leadd on line 1 | | **OK** | | | |
| Lead1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Lead1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Lead1 on line 1 | | **OK** | | | |
| Let | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1 | | **OK** | | | |
| Let˽ | Token = Let | | Token = Let | | **OK** | | | |
| Let↵ | Token = Let | | Token = Let | | **OK** | | | |
| Let: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Let. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Let( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Let? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Let- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Let+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Let, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Let| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Let[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Let; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Let on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Lett | Lexical Analyzer Encountered 1 error/s.  Invalid input: Lett on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Lett on line 1 | | **OK** | | | |
| Let1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Let1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Let1 on line 1 | | **OK** | | | |
| Loop | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loopon line 1 | | **OK** | | | |
| Loop˽ | Token = Loop | | Token = Loop | | **OK** | | | |
| Loop↵ | Token = Loop | | Token = Loop | | **OK** | | | |
| Loop: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loopon line 1  Invalid input: : on line 1 | | **OK** | | | |
| Loop. | Lexical Analyzer Encountered 1 error/s.  Token = Loop Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Loop Invalid input: . on line 1 | | **OK** | | | |
| Loop< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Loop( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Loop? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Loop- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: Loop on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: Loop on line 1 | | **OK** | | | |
| Loop+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Loop, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Loop| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Loop[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Loop; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Loop/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Loop on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Loopp | Lexical Analyzer Encountered 1 error/s.  Invalid input: Loopp on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Loopp on line 1 | | **OK** | | | |
| Loop1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Loop1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Loop1 on line 1 | | **OK** | | | |
| LoopIf | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1 | | **OK** | | | |
| LoopIf˽ | Token = LoopIf | | Token = LoopIf | | **OK** | | | |
| LoopIf↵ | Token = LoopIf | | Token = LoopIf | | **OK** | | | |
| LoopIf: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: : on line 1 | | **OK** | | | |
| LoopIf. | Token = LoopIf | | Token = LoopIf | | **OK** | | | |
| LoopIf< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: < on line 1 | | **OK** | | | |
| LoopIf( | Lexical Analyzer Encountered 1 error/s.  Token = LoopIf  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = LoopIf  Invalid input: ( on line 1 | | **OK** | | | |
| LoopIf? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| LoopIf- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: - on line 1 | | **OK** | | | |
| LoopIf+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: + on line 1 | | **OK** | | | |
| LoopIf, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: , on line 1 | | **OK** | | | |
| LoopIf| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: | on line 1 | | **OK** | | | |
| LoopIf[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| LoopIf; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| LoopIf/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: LoopIf on line 1  Invalid input: / on line 1 | | **OK** | | | |
| LoopIff | Lexical Analyzer Encountered 1 error/s.  Invalid input: LoopIff on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: LoopIff on line 1 | | **OK** | | | |
| LoopIf1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: LoopIf1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: LoopIf1 on line 1 | | **OK** | | | |
| No | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1 | | **OK** | | | |
| No˽ | Token = No | | Token = No | | **OK** | | | |
| No↵ | Token = No | | Token = No | | **OK** | | | |
| No. | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: . on line 1 | | **OK** | | | |
| No; | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: ; on line 1 | | **OK** | | | |
| No& | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: & on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: & on line 1 | | **OK** | | | |
| No| | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: | on line 1 | | **OK** | | | |
| No) | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: ) on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = No  Invalid input: ) on line 1 | | **OK** | | | |
| No( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: ( online 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: ( online 1 | | **OK** | | | |
| No? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| No: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: : on line 1 | | **OK** | | | |
| No< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: < on line 1 | | **OK** | | | |
| No- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: - on line 1 | | **OK** | | | |
| No+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: + on line 1 | | **OK** | | | |
| No, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: , on line 1 | | **OK** | | | |
| No[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: No on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Noo | Lexical Analyzer Encountered 1 error/s.  Invalid input: Noo on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Noo on line 1 | | **OK** | | | |
| No1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: No1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: No1 on line 1 | | **OK** | | | |
| Null | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1 | | **OK** | | | |
| Null˽ | Token = Null | | Token = Null | | **OK** | | | |
| Null↵ | Token = Null | | Token = Null | | **OK** | | | |
| Null: | Lexical Analyzer Encountered 1 error/s.  Token = Null  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Null  Invalid input: : on line 1 | | **OK** | | | |
| Null. | Lexical Analyzer Encountered 1 error/s.  Token = Null  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Token = Null  Invalid input: . on line 1 | | **OK** | | | |
| Null< | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: < on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Null( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Null? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Null- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Null+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Null, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Null| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Null[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Null; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Null/ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: / on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Null on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Nulll | Lexical Analyzer Encountered 1 error/s.  Invalid input: Nulll on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Nulll on line 1 | | **OK** | | | |
| Null1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Null1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Null1 on line 1 | | **OK** | | | |
| Object | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1 | | **OK** | | | |
| Object˽ | Token = Object | | Token = Object | | **OK** | | | |
| Object↵ | Token = Object | | Token = Object | | **OK** | | | |
| Object: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Object. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Object( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Object? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Object- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Object+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Object, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Object| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Object[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Object; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Object on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Objectt | Lexical Analyzer Encountered 1 error/s.  Invalid input: Objectt on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Objectt on line 1 | | **OK** | | | |
| Object1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Object1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Object1 on line 1 | | **OK** | | | |
| Option | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1 | | **OK** | | | |
| Option˽ | Token = Option | | Token = Option | | **OK** | | | |
| Option↵ | Token = Option | | Token = Option | | **OK** | | | |
| Option: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Option. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Option( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Option? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Option- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Option+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Option, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Option| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Option[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Option; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Option on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Optionn | Lexical Analyzer Encountered 1 error/s.  Invalid input: Optionn on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Optionn on line 1 | | **OK** | | | |
| Option1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Option1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Option1 on line 1 | | **OK** | | | |
| Or | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1 | | **OK** | | | |
| Or˽ | Token = Or | | Token = Or | | **OK** | | | |
| Or↵ | Token = Or | | Token = Or | | **OK** | | | |
| Or: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Or. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Or( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Or? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Or- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Or+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Or, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Or| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Or[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Or; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Or on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Orr | Lexical Analyzer Encountered 1 error/s.  Invalid input: : Orr line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: : Orr line 1 | | **OK** | | | |
| Or1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: : Or1 line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: : Or1 line 1 | | **OK** | | | |
| Otherwise | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1 | | **OK** | | | |
| Otherwise˽ | Token = Otherwise | | Token = Otherwise | | **OK** | | | |
| Otherwise↵ | Token = Otherwise | | Token = Otherwise | | **OK** | | | |
| Otherwise: | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: : on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Otherwise. | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: . on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Otherwise( | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ( on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Otherwise? | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ? on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Otherwise- | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: - on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Otherwise+ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: + on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Otherwise, | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: , on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Otherwise| | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: | on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Otherwise[ | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: [ on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Otherwise; | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ; on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: Otherwise on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Otherwisee | Lexical Analyzer Encountered 1 error/s.  Invalid input: Otherwisee on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Otherwisee on line 1 | | **OK** | | | |
| Otherwise1 | Lexical Analyzer Encountered 1 error/s.  Invalid input: Otherwise1 on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Otherwise1 on line 1 | | **OK** | | | |
| of | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: of on line 1 | | Lexical Analyzer Encountered 1 error/s.  Proper delimiter expected: of on line 1 | | **OK** | | | |
| of˽ | Token = of | | Token = of | | **OK** | | | |
| of↵ | Token = of | | Token = of | | **OK** | | | |
| of: | Proper delimiter expected: of on line 1  Invalid input: : on line 1 | | Proper delimiter expected: of on line 1  Invalid input: : on line 1 | | **OK** | | | |
| of. | Proper delimiter expected: of on line 1  Invalid input: . on line 1 | | Proper delimiter expected: of on line 1  Invalid input: . on line 1 | | **OK** | | | |
| of( | Proper delimiter expected: of on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: of on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| of? | Proper delimiter expected: of on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: of on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| of- | Proper delimiter expected: of on line 1  Invalid input: - on line 1 | | Proper delimiter expected: of on line 1  Invalid input: - on line 1 | | **OK** | | | |
| of+ | Proper delimiter expected: of on line 1  Invalid input: + on line 1 | | Proper delimiter expected: of on line 1  Invalid input: + on line 1 | | **OK** | | | |
| of, | Proper delimiter expected: of on line 1  Invalid input: , on line 1 | | Proper delimiter expected: of on line 1  Invalid input: , on line 1 | | **OK** | | | |
| of| | Proper delimiter expected: of on line 1  Invalid input: | on line 1 | | Proper delimiter expected: of on line 1  Invalid input: | on line 1 | | **OK** | | | |
| of[ | Proper delimiter expected: of on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: of on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| of; | Proper delimiter expected: of on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: of on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| off | Invalid input: off on line 1 | | Invalid input: off on line 1 | | **OK** | | | |
| of1 | Invalid input: of1 on line 1 | | Invalid input: of1 on line 1 | | **OK** | | | |
| Read | Proper delimiter expected: Read on line 1 | | Proper delimiter expected: Read on line 1 | | **OK** | | | |
| Read˽ | Token = Read | | Token = Read | | **OK** | | | |
| Read↵ | Token = Read | | Token = Read | | **OK** | | | |
| Read: | Proper delimiter expected: Read on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Read. | Proper delimiter expected: Read on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Read( | Proper delimiter expected: Read on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Read? | Proper delimiter expected: Read on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Read- | Proper delimiter expected: Read on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Read+ | Proper delimiter expected: Read on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Read, | Proper delimiter expected: Read on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Read| | Proper delimiter expected: Read on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Read[ | Proper delimiter expected: Read on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Read; | Proper delimiter expected: Read on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Read on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Readd | Invalid input: Readd on line 1 | | Invalid input: Readd on line 1 | | **OK** | | | |
| Read1 | Invalid input: Read1 on line 1 | | Invalid input: Read1 on line 1 | | **OK** | | | |
| Response | Proper delimiter expected: Response on line 1 | | Proper delimiter expected: Response on line 1 | | **OK** | | | |
| Response˽ | Token = Response | | Token = Response | | **OK** | | | |
| Response↵ | Token = Response | | Token = Response | | **OK** | | | |
| Response: | Proper delimiter expected: Response on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Response. | Proper delimiter expected: Response on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Response( | Proper delimiter expected: Response on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Response? | Proper delimiter expected: Response on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Response- | Proper delimiter expected: Response on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Response+ | Proper delimiter expected: Response on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Response, | Proper delimiter expected: Response on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Response| | Proper delimiter expected: Response on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Response[ | Proper delimiter expected: Response on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Response; | Proper delimiter expected: Response on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Response on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Responsee | Invalid input: Responsee on line 1 | | Invalid input: Responsee on line 1 | | **OK** | | | |
| Response1 | Invalid input: Response1 on line 1 | | Invalid input: Response1 on line 1 | | **OK** | | | |
| Say | Proper delimiter expected: Say on line 1 | | Proper delimiter expected: Say on line 1 | | **OK** | | | |
| Say˽ | Token = Say | | Token = Say | | **OK** | | | |
| Say↵ | Token = Say | | Token = Say | | **OK** | | | |
| Say: | Proper delimiter expected: Say on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Say. | Proper delimiter expected: Say on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Say( | Proper delimiter expected: Say on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Say? | Proper delimiter expected: Say on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Say- | Proper delimiter expected: Say on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Say+ | Proper delimiter expected: Say on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Say, | Proper delimiter expected: Say on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Say| | Proper delimiter expected: Say on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Say[ | Proper delimiter expected: Say on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Say; | Proper delimiter expected: Say on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Say on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Sayy | Invalid input: Sayy on line 1 | | Invalid input: Sayy on line 1 | | **OK** | | | |
| Say1 | Invalid input: Say1 on line 1 | | Invalid input: Say1 on line 1 | | **OK** | | | |
| Skip | Proper delimiter expected: Skip on line 1 | | Proper delimiter expected: Skip on line 1 | | **OK** | | | |
| Skip˽ | Token = Skip | | Token = Skip | | **OK** | | | |
| Skip↵ | Token = Skip | | Token = Skip | | **OK** | | | |
| Skip. | Token = Skip  Invalid input: . on line 1 | | Token = Skip  Invalid input: . on line 1 | | **OK** | | | |
| Skip< | Proper delimiter expected: Skip on line 1  Invalid input: < on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Skip( | Proper delimiter expected: Skip on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Skip? | Proper delimiter expected: Skip on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Skip- | Proper delimiter expected: Skip on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Skip+ | Proper delimiter expected: Skip on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Skip, | Proper delimiter expected: Skip on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Skip| | Proper delimiter expected: Skip on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Skip[ | Proper delimiter expected: Skip on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Skip; | Proper delimiter expected: Skip on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Skip/ | Proper delimiter expected: Skip on line 1  Invalid input: / on line 1 | | Proper delimiter expected: Skip on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Skipp | Invalid input: Skipp on line 1 | | Invalid input: Skipp on line 1 | | **OK** | | | |
| Skip1 | Invalid input: Skip1 on line 1 | | Invalid input: Skip1 on line 1 | | **OK** | | | |
| Start | Proper delimiter expected: Start on line 1 | | Proper delimiter expected: Start on line 1 | | **OK** | | | |
| Start˽ | Token = Start | | Token = Start | | **OK** | | | |
| Start↵ | Token = Start | | Token = Start | | **OK** | | | |
| Start: | Proper delimiter expected: Start on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Start. | Proper delimiter expected: Start on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Start( | Proper delimiter expected: Start on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Start? | Proper delimiter expected: Start on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Start- | Proper delimiter expected: Start on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Start+ | Proper delimiter expected: Start on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Start, | Proper delimiter expected: Start on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Start| | Proper delimiter expected: Start on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Start[ | Proper delimiter expected: Start on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Start; | Proper delimiter expected: Start on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Start on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Startt | Invalid input: Startt on line 1 | | Invalid input: Startt on line 1 | | **OK** | | | |
| Start1 | Invalid input: Start1 on line 1 | | Invalid input: Start1 on line 1 | | **OK** | | | |
| State | Proper delimiter expected: Start on line 1 | | Proper delimiter expected: State on line 1 | | **OK** | | | |
| State˽ | Token = State | | Token = State | | **OK** | | | |
| State↵ | Token = State | | Token = State | | **OK** | | | |
| State: | Proper delimiter expected: State on line 1  Invalid input: : on line 1 | | Proper delimiter expected: State on line 1  Invalid input: : on line 1 | | **OK** | | | |
| State. | Proper delimiter expected: State on line 1  Invalid input: . on line 1 | | Proper delimiter expected: State on line 1  Invalid input: . on line 1 | | **OK** | | | |
| State( | Proper delimiter expected: State on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: State on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| State? | Proper delimiter expected: State on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: State on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| State- | Proper delimiter expected: State on line 1  Invalid input: - on line 1 | | Proper delimiter expected: State on line 1  Invalid input: - on line 1 | | **OK** | | | |
| State+ | Proper delimiter expected: State on line 1  Invalid input: + on line 1 | | Proper delimiter expected: State on line 1  Invalid input: + on line 1 | | **OK** | | | |
| State, | Proper delimiter expected: State on line 1  Invalid input: , on line 1 | | Proper delimiter expected: State on line 1  Invalid input: , on line 1 | | **OK** | | | |
| State| | Proper delimiter expected: State on line 1  Invalid input: | on line 1 | | Proper delimiter expected: State on line 1  Invalid input: | on line 1 | | **OK** | | | |
| State[ | Proper delimiter expected: State on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: State on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| State; | Proper delimiter expected: State on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: State on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Statee | Invalid input: Statee on line 1 | | Invalid input: Statee on line 1 | | **OK** | | | |
| State1 | Invalid input: State1 on line 1 | | Invalid input: State1 on line 1 | | **OK** | | | |
| Stop | Proper delimiter expected: Stop on line 1 | | Proper delimiter expected: Stop on line 1 | | **OK** | | | |
| Stop˽ | Token = Stop | | Token = Stop | | **OK** | | | |
| Stop↵ | Token = Stop | | Token = Stop | | **OK** | | | |
| Stop. | Token = Stop  Invalid input: . on line 1 | | Token = Stop  Invalid input: . on line 1 | | **OK** | | | |
| Stop< | Proper delimiter expected: Stop on line 1  Invalid input: < on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Stop( | Proper delimiter expected: Stop on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Stop? | Proper delimiter expected: Stop on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Stop- | Proper delimiter expected: Stop on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Stop+ | Proper delimiter expected: Stop on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Stop, | Proper delimiter expected: Stop on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Stop| | Proper delimiter expected: Stop on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Stop[ | Proper delimiter expected: Stop on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Stop; | Proper delimiter expected: Stop on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Stop/ | Proper delimiter expected: Stop on line 1  Invalid input: / on line 1 | | Proper delimiter expected: Stop on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Stopp | Invalid input: Stopp on line 1 | | Invalid input: Stopp on line 1 | | **OK** | | | |
| Stop1 | Invalid input: Stop1 on line 1 | | Invalid input: Stop1 on line 1 | | **OK** | | | |
| String | Proper delimiter expected: String on line 1 | | Proper delimiter expected: String on line 1 | | **OK** | | | |
| String˽ | Token = String | | Token = String | | **OK** | | | |
| String↵ | Token = String | | Token = String | | **OK** | | | |
| String: | Proper delimiter expected: String on line 1  Invalid input: : on line 1 | | Proper delimiter expected: String on line 1  Invalid input: : on line 1 | | **OK** | | | |
| String. | Token = String  Invalid input: . on line 1 | | Token = String  Invalid input: . on line 1 | | **OK** | | | |
| String< | Proper delimiter expected: String on line 1  Invalid input: < on line 1 | | Proper delimiter expected: String on line 1  Invalid input: < on line 1 | | **OK** | | | |
| String( | Proper delimiter expected: String on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: String on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| String? | Proper delimiter expected: String on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: String on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| String- | Proper delimiter expected: String on line 1  Invalid input: - on line 1 | | Proper delimiter expected: String on line 1  Invalid input: - on line 1 | | **OK** | | | |
| String+ | Proper delimiter expected: String on line 1  Invalid input: + on line 1 | | Proper delimiter expected: String on line 1  Invalid input: + on line 1 | | **OK** | | | |
| String, | Proper delimiter expected: String on line 1  Invalid input: , on line 1 | | Proper delimiter expected: String on line 1  Invalid input: , on line 1 | | **OK** | | | |
| String| | Proper delimiter expected: String on line 1  Invalid input: | on line 1 | | Proper delimiter expected: String on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| String[ | Proper delimiter expected: String on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: String on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| String; | Proper delimiter expected: String on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: String on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| String/ | Proper delimiter expected: String on line 1  Invalid input: / on line 1 | | Proper delimiter expected: String on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Stringg | Invalid input: Stringg on line 1 | | Invalid input: Stringg on line 1 | | **OK** | | | |
| String1 | Invalid input: String1 on line 1 | | Invalid input: String1 on line 1 | | **OK** | | | |
| Task | Proper delimiter expected: Task on line 1 | | Proper delimiter expected: Task on line 1 | | **OK** | | | |
| Task˽ | Token = Task | | Token = Task | | **OK** | | | |
| Task↵ | Token = Task | | Token = Task | | **OK** | | | |
| Task: | Proper delimiter expected: Task on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Task. | Proper delimiter expected: Task on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Task( | Proper delimiter expected: Task on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Task? | Proper delimiter expected: Task on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Task- | Proper delimiter expected: Task on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Task+ | Proper delimiter expected: Task on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Task, | Proper delimiter expected: Task on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Task| | Proper delimiter expected: Task on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Task[ | Proper delimiter expected: Task on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Task; | Proper delimiter expected: Task on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Task on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Taskk | Invalid input: Taskk on line 1 | | Invalid input: Taskk on line 1 | | **OK** | | | |
| Task1 | Invalid input: Task1 on line 1 | | Invalid input: Task1 on line 1 | | **OK** | | | |
| Until | Proper delimiter expected: Until on line 1 | | Proper delimiter expected: Until on line 1 | | **OK** | | | |
| Until˽ | Token = Until | | Token = Until | | **OK** | | | |
| Until↵ | Token = Until | | Token = Until | | **OK** | | | |
| Until( | Token = Until  Invalid input: ( on line 1 | | Token = Until  Invalid input: ( on line 1 | | **OK** | | | |
| Until: | Proper delimiter expected: Until on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: : on line 1 | | **OK** | | | |
| Until. | Proper delimiter expected: Until on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Until? | Proper delimiter expected: Until on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Until- | Proper delimiter expected: Until on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Until+ | Proper delimiter expected: Until on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Until, | Proper delimiter expected: Until on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Until| | Proper delimiter expected: Until on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Until[ | Proper delimiter expected: Until on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Until; | Proper delimiter expected: Until on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Until/ | Proper delimiter expected: Until on line 1  Invalid input: / on line 1 | | Proper delimiter expected: Until on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Untill | Invalid input: Untill on line 1 | | Invalid input: Untill on line 1 | | **OK** | | | |
| Until1 | Invalid input: Until1 on line 1 | | Invalid input: Until1 on line 1 | | **OK** | | | |
| Var | Proper delimiter expected: Var on line 1 | | Proper delimiter expected: Var on line 1 | | **OK** | | | |
| Var˽ | Token = Var | | Token = Var | | **OK** | | | |
| Var↵ | Token = Var | | Token = Var | | **OK** | | | |
| Var: | Proper delimiter expected: Var on line 1  Invalid input: : on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Var. | Proper delimiter expected: Var on line 1  Invalid input: . on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: . on line 1 | | **OK** | | | |
| Var( | Proper delimiter expected: Var on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Var? | Proper delimiter expected: Var on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Var- | Proper delimiter expected: Var on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Var+ | Proper delimiter expected: Var on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Var, | Proper delimiter expected: Var on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Var| | Proper delimiter expected: Var on line 1  Invalid input: | on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: | on line 1 | | **OK** | | | |
| Var[ | Proper delimiter expected: Var on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Var; | Proper delimiter expected: Var on line 1  Invalid input: ; on line 1 | | Proper delimiter expected: Var on line 1  Invalid input: ; on line 1 | | **OK** | | | |
| Varr | Invalid input: Varr on line 1 | | Invalid input: Varr on line 1 | | **OK** | | | |
| Var1 | Invalid input: Var1 on line 1 | | Invalid input: Var1 on line 1 | | **OK** | | | |
| Yes | Proper delimiter expected: Yes on line 1 | | Proper delimiter expected: Yes on line 1 | | **OK** | | | |
| Yes˽ | Token = Yes | | Token = Yes | | **OK** | | | |
| Yes↵ | Token = Yes | | Token = Yes | | **OK** | | | |
| Yes. | Token = Yes  Invalid input: . on line 1 | | Token = Yes  Invalid input: . on line 1 | | **OK** | | | |
| Yes; | Token = Yes  Invalid input: ; on line 1 | | Token = Yes  Invalid input: ; on line 1 | | **OK** | | | |
| Yes& | Token = Yes  Invalid input: & on line 1 | | Token = Yes  Invalid input: & on line 1 | | **OK** | | | |
| Yes| | Token = Yes  Invalid input: | on line 1 | | Token = Yes  Invalid input: | on line 1 | | **OK** | | | |
| Yes) | Token = Yes  Invalid input: ) on line 1 | | Token = Yes  Invalid input: ) on line 1 | | **OK** | | | |
| Yes( | Proper delimiter expected: Yes on line 1  Invalid input: ( on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: ( on line 1 | | **OK** | | | |
| Yes? | Proper delimiter expected: Yes on line 1  Invalid input: ? on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: ? on line 1 | | **OK** | | | |
| Yes- | Proper delimiter expected: Yes on line 1  Invalid input: - on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: - on line 1 | | **OK** | | | |
| Yes+ | Proper delimiter expected: Yes on line 1  Invalid input: + on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: + on line 1 | | **OK** | | | |
| Yes, | Proper delimiter expected: Yes on line 1  Invalid input: , on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: , on line 1 | | **OK** | | | |
| Yes[ | Proper delimiter expected: Yes on line 1  Invalid input: [ on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: [ on line 1 | | **OK** | | | |
| Yes< | Proper delimiter expected: Yes on line 1  Invalid input: < on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: < on line 1 | | **OK** | | | |
| Yes/ | Proper delimiter expected: Yes on line 1  Invalid input: / on line 1 | | Proper delimiter expected: Yes on line 1  Invalid input: / on line 1 | | **OK** | | | |
| Yess | Invalid input: Yess on line 1 | | Invalid input: Yess on line 1 | | **OK** | | | |
| Yes1 | Invalid input: Yes1 on line 1 | | Invalid input: Yes1 on line 1 | | **OK** | | | |
|  | **IDENTIFIER** | | | |  | | |  |
| Num˽ | Token = id | | Token = id | | **OK** | | | |
| AgE˽ | Token = id | | Token = id | | **OK** | | | |
| CTR˽ | Token = id | | Token = id | | **OK** | | | |
| aDD˽ | Lexical Analyzer Encountered 1 error/s.  Invalid input: aDD on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: aDD on line 1 | | **OK** | | | |
| Num | Lexical Analyzer Encountered 1 error/s.  Invalid input: Num on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: Num on line 1 | | **OK** | | | |
| 1age˽ | Lexical Analyzer Encountered 1 error/s.  Invalid input: 1age on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: 1age on line 1 | | **OK** | | | |
| 2Add˽ | Lexical Analyzer Encountered 1 error/s.  Invalid input: 2Add on line 1 | | Lexical Analyzer Encountered 1 error/s.  Invalid input: 2Add on line 1 | | **OK** | | | |
| Ctr1˽ | Token = id | | Token = id | | **OK** | | | |
| A200˽ | Token = id | | Token = id | | **OK** | | | |
| N200um˽ | Token = id | | Token = id | | **OK** | | | |
| Abcdefghijk˽ | Token = id | | Token = id | | **OK** | | | |
| Nu\_m˽ | Token = id | | Token = id | | **OK** | | | |
| Num\_ | Invalid input: Num\_ on line 1 | | Invalid input: Num\_ on line 1 | | **OK** | | | |
| \_Num | Token = id  Invalid input: \_ on line 1 | | Token = id  Invalid input: \_ on line 1 | | **OK** | | | |
| Num\_˽ | Token = id | | Token = id | | **OK** | | | |
| N\_u\_m˽ | Token = id | | Token = id | | **OK** | | | |
| &um˽ | Invalid input: & on line 1  Invalid input: um on line 1 | | Invalid input: & on line 1  Invalid input: um on line 1 | | **OK** | | | |
| N&m˽ | Invalid input: N on line 1  Invalid input: & on line 1  Invalid input: m on line 1 | | Invalid input: N on line 1  Invalid input: & on line 1  Invalid input: m on line 1 | | **OK** | | | |
| Nu&˽ | Invalid input: Nu on line 1  Invalid input: & on line 1 | | Invalid input: Nu on line 1  Invalid input: & on line 1 | | **OK** | | | |
|  | **LITERALS** | | | |  |  | | |
| **INT LITERAL** |  | |  | |  | | | |
| 1˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| 123˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| 1234567890˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| 0000000001˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| ~1˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| ~123˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| ~1234567890˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| ~0000000001˽ | Token = intlit | | Token = intlit | | **OK** | | | |
| **DOUBLE LITERAL** |  | |  | |  | | | |
| 0.1˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| 1.12345˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| 12345.5˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| 12345.54321˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| 0.0˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| ~0.1˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| ~0.0˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| ~1.12345˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| ~12345.5˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| ~12345.54321˽ | Token = doublelit | | Token = doublelit | | **OK** | | | |
| **CHAR LITERAL** |  | |  | |  | | | |
| ‘A’˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘a’˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘a˽ | Invalid input: ‘a on line 1 | | Invalid input: ‘a on line 1 | | **OK** | | | |
| ‘a” ˽ | Invalid input: ‘a” on line 1 | | Invalid input: ‘a” on line 1 | | **OK** | | | |
| ‘ABC‘˽ | Invalid input: ‘ABC’ on line 1 | | Invalid input: ‘ABC’ on line 1 | | **OK** | | | |
| ‘1’˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘&’ ˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘;’ ˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘+’ ˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘[6]’ ˽ | Invalid input: ‘[6]’ on line 1 | | Invalid input: ‘[6]’ on line 1 | | **OK** | | | |
| ‘?’ ˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘.’ ˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘’˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| ‘˽’˽ | Token = charlit | | Token = charlit | | **OK** | | | |
| **STRING LITERAL** |  | |  | |  | | | |
| “Hello World” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “hello world” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “H” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “h” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “Hello!” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “Hello?” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “1” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “123” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “&”˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “Hello World’˽ | Invalid input: “Hello World’ on line 1 | | Invalid input: “Hello World’ on line 1 | | **OK** | | | |
| “”˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “˽” ˽ | Token = stringlit | | Token = stringlit | | **OK** | | | |
| “””˽ | Invalid input: “” on line 1  Invalid input: “ on line 1 | | Invalid input: “” on line 1  Invalid input: “ on line 1 | | **OK** | | | |
| **BOOLEAN LITERALS** | |  |  | |  | | | |
| Boolean Num is Yes.˽ | Token = Boolean  Token = id  Token = is  Token = boollit  Token = . | | Token = Boolean  Token = id  Token = is  Token = Yes  Token = . | | **FAILED** | | | |
| Boolean Num is No.˽ | Token = Boolean  Token = id  Token = is  Token = boollit  Token = . | | Token = Boolean  Token = id  Token = is  Token = No  Token = . | | **FAILED** | | | |
| Boolean Num is yes.˽ | Token = Boolean  Token = id  Token = is  Token = .  Invalid input: yes on line 1 | | Token = Boolean  Token = id  Token = is  Token = .  Invalid input: yes on line 1 | | **OK** | | | |
| Boolean Num is no.˽ | Token = Boolean  Token = id  Token = is  Token = .  Invalid input: no on line 1 | | Token = Boolean  Token = id  Token = is  Token = .  Invalid input: no on line 1 | | **OK** | | | |
|  | **RESERVED SYMBOL** | | | |  | |  | |
| && |  | |  | |  | | | |
| &&˽ | Token = && | | Token = && | | **OK** | | | |
| &&””˽ | Token = &&  Token = stringlit | | Token = &&  Token = stringlit | | **OK** | | | |
| &&’’˽ | Token = &&  Token = charlit | | Token = &&  Token = charlit | | **OK** | | | |
| &&(˽ | Token = &&  Token = ( | | Token = &&  Token = ( | | **OK** | | | |
| &&!˽ | Token = &&  Token = ! | | Token = &&  Token = ! | | **OK** | | | |
| &&1˽ | Token = &&  Token = intlit | | Token = &&  Token = intlit | | **OK** | | | |
| &&A˽ | Token = &&  Token = id | | Token = &&  Token = id | | **OK** | | | |
| &&+˽ | Token = +  Invalid input: & on line 1  Invalid input: & on line 1 | | Token = +  Invalid input: & on line 1  Invalid input: & on line 1 | | **OK** | | | |
| &&&˽ | Token = &&  Invalid input: & on line 1 | | Token = &&  Invalid input: & on line 1 | | **OK** | | | |
| || | Invalid input: | on line 1  Invalid input: | on line 1 | | Invalid input: | on line 1  Invalid input: | on line 1 | | **OK** | | | |
| ||˽ | Token = || | | Token = || | | **OK** | | | |
| ||↵ | Token = || | | Token = || | | **OK** | | | |
| ||””˽ | Token = ||  Token = stringlit | | Token = ||  Token = stringlit | | **OK** | | | |
| ||’’˽ | Token = ||  Token = charlit | | Token = ||  Token = charlit | | **OK** | | | |
| ||(˽ | Token = ||  Token = ( | | Token = ||  Invalid input: ( on line 1 | | **OK** | | | |
| ||!˽ | Token = ||  Token = ! | | Token = ||  Token = ! | | **OK** | | | |
| ||~1˽ | Token = ||  Token = intlit | | Token = ||  Token = intlit | | **OK** | | | |
| ||1˽ | Token = ||  Token = intlit | | Token = ||  Token = intlit | | **OK** | | | |
| ||A˽ | Token = ||  Token = id | | Token = ||  Token = id | | **OK** | | | |
|  |  | |  | |  | | | |
| ||+˽ | Token = ||  Invalid input: + on line 1 | | Token = +  Invalid input: | on line 1  Invalid input: | on line 1 | | **OK** | | | |
| |||˽ | Token = ||  Invalid input: | on line 1 | | Token = ||  Invalid input: | on line 1 | | **OK** | | | |
| . | Invalid input: . on line 1 | | Invalid input: . on line 1 | | **OK** | | | |
| . ˽ | Token = . | | Token = . | | **OK** | | | |
| .↵ | Token = . | | Token = . | | **OK** | | | |
| .#˽ | Token = .  Token = # | | Token = .  Token = # | | **OK** | | | |
| .A˽ | Token = .  Token = id | | Token = .  Token = id | | **OK** | | | |
|  |  | |  | |  | | | |
|  |  | |  | |  | | | |
| ..˽ | Token = .  Invalid input: . on line 1 | | Token = .  Invalid input: . on line 1 | | **OK** | | | |
| ! | Invalid input: ! on line 1 | | Invalid input: ! on line 1 | | **OK** | | | |
| ! ˽ | Token = ! | | Token = ! | | **OK** | | | |
| !↵ | Token = ! | | Token = ! | | **OK** | | | |
| !””˽ | Token = ! Token = stringlit | | Token = ! Token = stringlit | | **OK** | | | |
| !’’˽ | Token = ! Token = charlit | | Token = ! Token = charlit | | **OK** | | | |
| !(˽ | Token = !  Token = ( | | Token = !  Token = ( | | **OK** | | | |
|  |  | |  | |  | | | |
| !~1˽ | Token = !  Token = intlit | | Token = !  Token = intlit | | **OK** | | | |
| !1˽ | Token = !  Token = intlit | | Token = !  Token = intlit | | **OK** | | | |
| !A˽ | Token = !  Token = id | | Token = !  Token = id | | **OK** | | | |
| !!˽ | Token = !  Token = ! | | Token = !  Token = ! | | **OK** | | | |
| != | Invalid input: ! on line 1  Invalid input: = on line 1 | | Invalid input: ! on line 1  Invalid input: = on line 1 | | **OK** | | | |
| !=˽ | Token = != | | Token = != | | **OK** | | | |
| !=↵ | Token = != | | Token = != | | **OK** | | | |
| !=!˽ | Token = !=  Token = ! | | Token = !=  Token = ! | | **OK** | | | |
| !=~1˽ | Token = !=  Token = intlit | | Token = !=  Token = intlit | | **OK** | | | |
| !=1˽ | Token = !=  Token = intlit | | Token = !=  Token = intlit | | **OK** | | | |
| !=A˽ | Token = !=  Token = id | | Token = !=  Token = id | | **OK** | | | |
| @ | Invalid input: @ on line 1 | | Invalid input: @ on line 1 | | **OK** | | | |
| @˽ | Invalid input: @ on line 1 | | Invalid input: @ on line 1 | | **OK** | | | |
| @↵ | Invalid input: @ on line 1 | | Invalid input: @ on line 1 | | **OK** | | | |
| @A˽ | Token = @  Token = id | | Token = @  Token = id | | **OK** | | | |
| @&˽ | Invalid input: @ on line 1  Invalid input: & on line 1 | | Invalid input: @ on line 1  Invalid input: & on line 1 | | **OK** | | | |
| @@˽ | Invalid input: @ on line 1  Invalid input: @ on line 1 | | Invalid input: @ on line 1  Invalid input: @ on line 1 | | **OK** | | | |
| ( | Invalid input: ( on line 1 | | Invalid input: ( on line 1 | | **OK** | | | |
| (˽ | Token = ( | | Token = ( | | **OK** | | | |
| (↵ | Token = ( | | Token = ( | | **OK** | | | |
| (!˽ | Token = (  Token = ! | | Token = (  Token = ! | | **OK** | | | |
| (~1˽ | Token = (  Token = intlit | | Token = (  Token = intlit | | **OK** | | | |
| (1˽ | Token = (  Token = intlit | | Token = (  Token = intlit | | **OK** | | | |
| (A˽ | Token = (  Token = id | | Token = (  Token = id | | **OK** | | | |
| (“”˽ | Token = (  Token = stringlit | | Token = (  Token = stringlit | | **OK** | | | |
| (‘’˽ | Token = (  Token = charlit | | Token = (  Token = charlit | | **OK** | | | |
| ((˽ | Token = (  Token = ( | | Token = (  Token = ( | | **OK** | | | |
| )˽ | Token = ) | | Token = ) | | **OK** | | | |
| )↵ | Token = ) | | Token = ) | | **OK** | | | |
| )+˽ | Token = )  Token = + | | Token = )  Token = + | | **OK** | | | |
| )-˽ | Token = )  Token = - | | Token = )  Token = - | | **OK** | | | |
| )\*˽ | Token = )  Token = \* | | Token = )  Token = \* | | **OK** | | | |
| )/˽ | Token = )  Token = / | | Token = )  Token = / | | **OK** | | | |
| )%˽ | Token = )  Token = % | | Token = )  Token = % | | **OK** | | | |
| ).˽ | Token = )  Token = . | | Token = )  Token = . | | **OK** | | | |
| )1˽ | Token = )  Token = intlit | | Token = )  Token = intlit | | **OK** | | | |
| ))˽ | Token = )  Token = ) | | Token = )  Token = ) | | **OK** | | | |
| [ | Invalid input: [ on line 1 | | Invalid input: [ on line 1 | | **OK** | | | |
| [˽ | Token = [ | | Token = [ | | **OK** | | | |
| [↵ | Invalid input: [ on line 1 | | Invalid input: [ on line 1 | | **OK** | | | |
| [1˽ | Token = [  Token = intlit | | Token = [  Token = intlit | | **OK** | | | |
| [A˽ | Token = [  Token = id | | Token = [  Token = ID | | **OK** | | | |
| [[˽ | Token = [  Invalid input: [ on line 1 | | Token = [  Invalid input: [ on line 1 | | **OK** | | | |
| ] | Invalid input: ] on line 1 | | Invalid input: ] on line 1 | | **OK** | | | |
| ]˽ | Token = ] | | Token = ] | | **OK** | | | |
| ]↵ | Token = ] | | Token = ] | | **OK** | | | |
| ][˽ | Token = ]  Token = [ | | Token = ]  Token = [ | | **OK** | | | |
| ]% | Token = ]  Token = % | | Token = ]  Token = % | | **OK** | | | |
| ]+˽ | Token = ]  Token = + | | Token = ]  Token = + | | **OK** | | | |
| ]-˽ | Token = ]  Token = - | | Token = ]  Token = - | | **OK** | | | |
| ]\*˽ | Token = ]  Token = \* | | Token = ]  Token = \* | | **OK** | | | |
| ]/˽ | Token = ]  Token = / | | Token = ]  Token = / | | **OK** | | | |
| ].˽ | Token = ]  Token = . | | Token = ]  Token = . | | **OK** | | | |
| ];˽ | Token = ]  Token = ; | | Token = ]  Token = ; | | **OK** | | | |
| ])˽ | Token = ]  Token = ) | | Token = ]  Token = ) | | **OK** | | | |
| ]]˽ | Token = ]  Invalid input: ] on line 1 | | Token = ]  Invalid input: ] on line 1 | | **OK** | | | |
| = | Invalid input: = on line 1 | | Invalid input: = on line 1 | | **OK** | | | |
| =˽ | Token = = | | Token = = | | **OK** | | | |
| =↵ | Token = = | | Token = = | | **OK** | | | |
| =”˽ | Token = =  Token = “ | | Token = =  Token = “ | | **OK** | | | |
| =’˽ | Token = =  Token = ‘ | | Token = =  Token = ‘ | | **OK** | | | |
| =(˽ | Token = =  Token = ( | | Token = =  Token = ( | | **OK** | | | |
| =!˽ | Token = =  Token = ! | | Token = =  Token = ! | | **OK** | | | |
| =~1˽ | Token = =  Token = intlit | | Token = =  Token = intlit | | **OK** | | | |
| =1˽ | Token = =  Token = intlit | | Token = =  Token = intlit | | **OK** | | | |
| =A˽ | Token = =  Token = id | | Token = =  Token = id | | **OK** | | | |
| == | Invalid input: = on line 1  Invalid input: = on line 1 | | Invalid input: = on line 1  Invalid input: = on line 1 | | **OK** | | | |
| ==˽ | Token = == | | Token = == | | **OK** | | | |
| ==↵ | Token = == | | Token = == | | **OK** | | | |
| ==!˽ | Token = ==  Token = ! | | Token = ==  Token = ! | | **OK** | | | |
| ==~1˽ | Token = ==  Token = intlit | | Token = ==  Token = intlit | | **OK** | | | |
| ==1˽ | Token = ==  Token = intlit | | Token = ==  Token = intlit | | **OK** | | | |
| ==A˽ | Token = ==  Token = id | | Token = ==  Token = id | | **OK** | | | |
| ===˽ | Token = ==  Invalid input: = on line 1 | | Token = ==  Invalid input: = on line 1 | | **OK** | | | |
| > | Invalid input: > on line 1 | | Invalid input: > on line 1 | | **OK** | | | |
| >˽ | Token = > | | Token = > | | **OK** | | | |
| >↵ | Token = > | | Token = > | | **OK** | | | |
| >!˽ | Token = >  Token = ! | | Token = >  Token = ! | | **OK** | | | |
| >~1˽ | Token = >  Token = intlit | | Token = >  Token = intlit | | **OK** | | | |
| >1˽ | Token = >  Token = intlit | | Token = >  Token = intlit | | **OK** | | | |
| >A˽ | Token = >  Token = id | | Token = >  Token = id | | **OK** | | | |
| >>˽ | Token = >  Invalid input: > on line 1 | | Token = >  Invalid input: > on line 1 | | **OK** | | | |
| >= | Invalid input: > on line 1  Invalid input: = on line 1 | | Invalid input: > on line 1  Invalid input: = on line 1 | | **OK** | | | |
| >=˽ | Token = >= | | Token = >= | | **OK** | | | |
| >=↵ | Token = >= | | Token = >= | | **OK** | | | |
| >=!˽ | Token = >=  Token = ! | | Token = >=  Token = ! | | **OK** | | | |
| >=~1˽ | Token = >=  Token = intlit | | Token = >=  Token = intlit | | **OK** | | | |
| >=1˽ | Token = >=  Token = intlit | | Token = >=  Token = intlit | | **OK** | | | |
| >=A˽ | Token = >=  Token = id | | Token = >=  Token = id | | **OK** | | | |
| < | Invalid input: < on line 1 | | Invalid input: < on line 1 | | **OK** | | | |
| <˽ | Token = < | | Token = < | | **OK** | | | |
| <↵ | Token = < | | Token = < | | **OK** | | | |
| <!˽ | Token = <  Token = ! | | Token = <  Token = ! | | **OK** | | | |
| <~1˽ | Token = <  Token = intlit | | Token = <  Token = intlit | | **OK** | | | |
| <1˽ | Token = <  Token = intlit | | Token = <  Token = intlit | | **OK** | | | |
| <A˽ | Token = <  Token = id | | Token = <  Token = id | | **OK** | | | |
| <<˽ | Token = <  Invalid input: < on line 1 | | Token = <  Invalid input: < on line 1 | | **OK** | | | |
| <= | Invalid input: < on line 1  Invalid input: = on line 1 | | Invalid input: < on line 1  Invalid input: = on line 1 | | **OK** | | | |
| <=˽ | Token = <= | | Token = <= | | **OK** | | | |
| <=↵ | Token = <= | | Token = <= | | **OK** | | | |
| <=!˽ | Token = <=  Token = ! | | Token = <=  Token = ! | | **OK** | | | |
| <=~1˽ | Token = <=  Token = intlit | | Token = <=  Token = intlit | | **OK** | | | |
| <=1˽ | Token = <=  Token = intlit | | Token = <=  Token = intlit | | **OK** | | | |
| <=A˽ | Token = <=  Token = id | | Token = <=  Token = id | | **OK** | | | |
| + | Invalid input: + on line 1 | | Invalid input: + on line 1 | | **OK** | | | |
| +˽ | Token = + | | Token = + | | **OK** | | | |
| +↵ | Token = + | | Token = + | | **OK** | | | |
| +!˽ | Token = +  Token = ! | | Token = +  Token = ! | | **OK** | | | |
| +~1˽ | Token = +  Token = intlit | | Token = +  Token = intlit | | **OK** | | | |
| +1˽ | Token = +  Token = intlit | | Token = +  Token = intlit | | **OK** | | | |
| +A˽ | Token = +  Token = id | | Token = +  Token = id | | **OK** | | | |
| +(˽ | Token = +  Token = ( | | Token = +  Token = ( | | **OK** | | | |
| ++ | Invalid input: + on line 1  Invalid input: + on line 1 | | Invalid input: + on line 1  Invalid input: + on line 1 | | **OK** | | | |
| ++˽ | Token = ++ | | Token = ++ | | **OK** | | | |
| ++↵ | Token = ++ | | Token = ++ | | **OK** | | | |
| ++)˽ | Token = ++  Token = ) | | Token = ++  Token = ) | | **OK** | | | |
| ++.˽ | Token = ++  Token = . | | Token = ++  Token = . | | **OK** | | | |
| ++,˽ | Token = ++  Token = , | | Token = ++  Token = , | | **OK** | | | |
| ++;˽ | Token = ++  Token = ; | | Token = ++  Token = ; | | **OK** | | | |
| ++A˽ | Token = ++  Token = id | | Token = ++  Token = id | | **OK** | | | |
|  |  | |  | |  | | | |
|  |  | |  | |  | | | |
| +++˽ | Token = ++  Invalid input: + on line 1 | | Token = ++  Invalid input: + on line 1 | | **OK** | | | |
| - | Invalid input: - on line 1 | | Invalid input: - on line 1 | | **OK** | | | |
| -˽ | Token = - | | Token = - | | **OK** | | | |
| -↵ | Token = - | | Token = - | | **OK** | | | |
| -!˽ | Token = -  Token = ! | | Token = -  Token = ! | | **OK** | | | |
| -~1˽ | Token = -  Token = intlit | | Token = -  Token = intlit | | **OK** | | | |
| -1˽ | Token = -  Token = intlint | | Token = -  Token = intlint | | **OK** | | | |
| -A˽ | Token = -  Token = id | | Token = -  Token = id | | **OK** | | | |
| -(˽ | Token = -  Token = ( | | Token = -  Token = ( | | **OK** | | | |
| -- | Invalid input: - on line 1  Invalid input: - on line 1 | | Invalid input: - on line 1  Invalid input: - on line 1 | | **OK** | | | |
| --˽ | Token = -- | | Token = -- | | **OK** | | | |
| --↵ | Token = -- | | Token = -- | | **OK** | | | |
| --)˽ | Token = --  Token = ) | | Token = --  Token = ) | | **OK** | | | |
| --.˽ | Token = --  Token = . | | Token = --  Token = . | | **OK** | | | |
| --,˽ | Token = --  Token = , | | Token = --  Token = , | | **OK** | | | |
|  |  | |  | |  | | | |
| --A˽ | Token = --  Token = id | | Token = --  Token = id | | **OK** | | | |
| ---˽ | Token = --  Invalid input: - on line 1 | | Token = --  Invalid input: - on line 1 | | **OK** | | | |
| \* | Invalid input: \* on line 1 | | Invalid input: \* on line 1 | | **OK** | | | |
| \*˽ | Token = \* | | Token = \* | | **OK** | | | |
| \*↵ | Token = \* | | Token = \* | | **OK** | | | |
| \*!˽ | Token = \*  Token = ! | | Token = \*  Token = ! | | **OK** | | | |
| \*~1˽ | Token = \*  Token = intlit | | Token = \*  Token = intlit | | **OK** | | | |
| \*1˽ | Token = \*  Token = intlit | | Token = \*  Token = intlit | | **OK** | | | |
| \*A˽ | Token = \*  Token = id | | Token = \*  Token = id | | **OK** | | | |
| \*(˽ | Token = \*  Token = ( | | Token = \*  Token = ( | | **OK** | | | |
|  |  | |  | |  | | | |
| \*=˽ | Token = \* Invalid input: = on line | |  | | **OK** | | | |
| \*\*˽ |  | |  | |  | | | |
| / |  | |  | |  | | | |
| /˽ |  | |  | |  | | | |
| /↵ |  | |  | |  | | | |
| /!˽ |  | |  | |  | | | |
| /~1˽ | Token = /  Token = intlit | | Token = /  Token = intlit | | **OK** | | | |
| /1˽ |  | |  | |  | | | |
| /A˽ |  | |  | |  | | | |
| /(˽ |  | |  | |  | | | |
| /;˽ |  | |  | |  | | | |
| /=˽ |  | |  | |  | | | |
| //˽ |  | |  | |  | | | |
| ; | Invalid input: ; on line 1 | | Invalid input: ; on line 1 | | **OK** | | | |
| ;˽ | Token = ; | | Token = ; | | **OK** | | | |
| ;↵ | Token = ; | | Token = ; | | **OK** | | | |
| ;””˽ | Token = ;  Token = stringlit | | Token = ;  Token = stringlit | | **OK** | | | |
| ;’’˽ | Token = ;  Token = charlit | | Token = ;  Token = charlit | | **OK** | | | |
| ;( ˽ | Token = ;  Token = ( | | Token = ;  Token = ( | | **OK** | | | |
| ;! ˽ | Token = ;  Token = ! | | Token = ;  Token = ! | | **OK** | | | |
| ;~1 ˽ | Token = ;  Token = intlit | | Token = ;  Token = intlit | | **OK** | | | |
| ;1˽ | Token = ;  Token = intlit | | Token = ;  Token = intlit | | **OK** | | | |
| ;A˽ | Token = ;  Token = stringlit | | Token = ;  Token = stringlit | | **OK** | | | |
|  |  | |  | |  | | | |
|  |  | |  | |  | | | |
| ;;˽ | Token = stringlit  Invalid input: ; on line 1 | | Token = stringlit  Invalid input: ; on line 1 | | **OK** | | | |
| : | Invalid input: : on line 1 | | Invalid input: : on line 1 | | **OK** | | | |
| :˽ | Token = : | | Token = : | | **OK** | | | |
| :↵ | Token = : | | Token = : | | **OK** | | | |
| :: ˽ | Token = :  Invalid input: : on line 1 | | Token = :  Invalid input: : on line 1 | | **OK** | | | |
| , ˽ | Token = , | | Token = , | | **OK** | | | |
| ,↵ | Token = , | | Token = , | | **OK** | | | |
| ,”” ˽ | Token = ,  Token = stringlit | | Token = ,  Token = stringlit | | **OK** | | | |
| ,A˽ | Token = ,  Token = stringlit | | Token = ,  Token = stringlit | | **OK** | | | |
| ,,˽ | Token = ,  Invalid input: , on line 1 | | Token = ,  Invalid input: , on line 1 | | **OK** | | | |

**SYNTAX TEST SCRIPT**

**General Structure of the Program**Syntax: <program> → <global>Lead : Start <body> End . <taskdef>

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **Actual Output** | **Remarks** |
| Lead:  Start  Var Int Num is 11.  End.# | Syntax Analyzer Succeeded… | Syntax Analyzer Succeeded... | **OK** |
| Lead:  Start  End.# | Syntax Analyzer Succeeded… | Syntax Analyzer Succeeded… | **OK** |
| Let Int Num is 11. Lead:  Start  End.# | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = # | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = # | **OK** |
| Task Int Age.  Lead:  Start  End.# | Token = Task  Token = Int  Token = id  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = # | Token = Task  Token = Int  Token = id  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = # | **OK** |
| Task Int Age.  Task null Log.  Lead:  Start  End.# | Token = Task  Token = Int  Token = id  Token = .  Token = Task  Token = id  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = #  Invalid input: null on line 2 | Token = Task  Token = Int  Token = id  Token = .  Token = Task  Token = id  Token = .  Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = #  Invalid input: null on line 2 | **OK** |
| Lead:  Start  End.  Task String Tree:  Start  Response “Hello”.  End.# | Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = Task  Token = String  Token = id  Token = :  Token = Start  Token = Response  Token = stringlit  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = End  Token = .  Token = Task  Token = String  Token = id  Token = :  Token = Start  Token = Response  Token = stringlit  Token = .  Token = End  Token = .  Token = # | **OK** |
| Lead:  Start.  Clear  End# | Token = Lead  Token = :  Token = .  Token = End  Token = #  Invalid input: Start on line 2  Invalid input: End on line 4 | Token = Lead  Token = :  Token = .  Token = End  Token = #  Invalid input: Start on line 2  Invalid input: End on line 4 | **OK** |

**Constant Declaration**Syntax: <varlet> → Let <vardec>

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **ActualOutput** | **Remarks** |
| Let Int Age is 25.**˽** | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = . | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = . | **OK** |
| Let Int Age is 35; Num is 45.**˽** | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = ;  Token = id  Token = is  Token = intlit  Token = . | Token = Let  Token = Int  Token = id  Token = is  Token = intlit  Token = ;  Token = id  Token = is  Token = intlit  Token = . | **OK** |
| Let Int age; Num is 55.**˽** | Token = Let  Token = Int  Token = ;  Token = id  Token = is  Token = intlit  Token = .  Invalid input: age on line 1 | Token = Let  Token = Int  Token = ;  Token = id  Token = is  Token = intlit  Token = .  Invalid input: age on line 1 | **OK** |
| Let Double Grade  is 1.5.**˽** | Token = Let  Token = Double  Token = id  Token = is  Token = doublelit  Token = . | Token = Let  Token = Double  Token = id  Token = is  Token = doublelit  Token = . | **OK** |
| Let Double Grade is 1.5; Mile is 2.5.**˽** | Token = Let  Token = Double  Token = id  Token = is  Token = doublelit  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | Token = Let  Token = Double  Token = id  Token = is  Token = doublelit  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | **OK** |
| Let double 1Grade; Mile is 2.5.**˽** | Token = Let  Token = ;  Token = id  Token = is  Token = doublelit  Token = .  Invalid input: double on line 1  Invalid input: 1Grade on line 1 | Token = Let  Token = ;  Token = id  Token = is  Token = doublelit  Token = .  Invalid input: double on line 1  Invalid input: 1Grade on line 1 | **OK** |
| Let Char Letter is ‘A’.**˽** | Token = Let  Token = Char  Token = id  Token = is  Token = charlit  Token = . | Token = Let  Token = Char  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Let Char Letter is ‘A’; Vowel is ‘E’.**˽** | Token = Let  Token = Char  Token = id  Token = is  Token = charlit  Token = ;  Token = id  Token = is  Token = charlit  Token = . | Token = Let  Token = Char  Token = id  Token = is  Token = charlit  Token = ;  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Let Char Letter; Vowel is ‘E’.**˽** | Token = Let  Token = Char  Token = id  Token = ;  Token = id  Token = is  Token = charlit  Token = . | Token = Let  Token = Char  Token = id  Token = ;  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Let Char Letter is ‘ANB’.**˽** | Token = Let  Token = Char  Token = id  Token = .  Invalid input: ‘ANB’ on line 1 | Token = Let  Token = Char  Token = id  Token = .  Invalid input: ‘ANB’ on line 1 | **OK** |
| Let String Name is “Joe”.**˽** | Token = Let  Token = String  Token = id  Token = is  Token = stringlit  Token = . | Token = Let  Token = String  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Let String Name is “Joe”; Surname is “Sam”.**˽** | Token = Let  Token = String  Token = id  Token = is  Token = stringlit  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | Token = Let  Token = String  Token = id  Token = is  Token = stringlit  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Let String Name; Surname is “Sam”.**˽** | Token = Let  Token = String  Token = id  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | Token = Let  Token = String  Token = id  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Let string Name is “Joe”.**˽** | Token = Let  Token = id  Token = is  Token = stringlit  Token = .  Invalid input: string on line 1 | Token = Let  Token = id  Token = is  Token = stringlit  Token = .  Invalid input: string on line 1 | **OK** |
| Let Boolean Case is No.**˽** | Token = Let  Token = Boolean  Token = id  Token = is  Token = boollit  Token = . | Token = Let  Token = Boolean  Token = id  Token = is  Token = No  Token = . | **FAILED** |
| Let Boolean Case is No; Status is Yes.**˽** | Token = Let  Token = Boolean  Token = id  Token = is  Token = boollit  Token = ;  Token = id  Token = is  Token = boollit  Token = . | Token = Let  Token = Boolean  Token = id  Token = is  Token = No  Token = ;  Token = id  Token = is  Token = Yes  Token = . | **FAILED** |
| Let Boolean Case; Status is Yes.**˽** | Token = Let  Token = Boolean  Token = id  Token = ;  Token = id  Token = is  Token = boollit  Token = . | Token = Let  Token = Boolean  Token = id  Token = ;  Token = id  Token = is  Token = Yes  Token = . | **FAILED** |
| Let boolean Case No.**˽** | Token = Let  Token = id  Token = boollit  Token = .  Invalid input: boolean on line 1 | Token = Let  Token = id  Token = No  Token = .  Invalid input: boolean on line 1 | **FAILED** |

**Variable Declaration**Syntax: <varlet> → Var<vardec>

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **Actual Output** | **Remarks** |
| Var Int Age is 25. | Token = Var  Token = Int  Token = id  Token = is  Token = intlit  Token = . | Token = Var  Token = Int  Token = id  Token = is  Token = intlit  Token = . | **OK** |
| Var Int Age is 35; Num is 45. | Token = Var  Token = Int  Token = id  Token = is  Token = intlit  Token = ;  Token = id  Token = is  Token = intlit  Token = . | Token = Var  Token = Int  Token = id  Token = is  Token = intlit  Token = ;  Token = id  Token = is  Token = intlit  Token = . | **OK** |
| Var Int Age; Num is 55. | Token = Var  Token = Int  Token = id  Token = ;  Token = id  Token = is  Token = intlit  Token = . | Token = Var  Token = Int  Token = id  Token = ;  Token = id  Token = is  Token = intlit  Token = . | **OK** |
| Var int Age is 25. | Token = Var  Token = id  Token = is  Token = intlit  Token = .  Invalid input: int on line 1 | Token = Var  Token = id  Token = is  Token = intlit  Token = .  Invalid input: int on line 1 | **OK** |
| Var Double Grade  is 1.5. | Token = Var  Token = Double  Token = id  Token = is  Token = doublelit  Token = . | Token = Var  Token = Double  Token = id  Token = is  Token = doublelit  Token = . | **OK** |
| Var Double Grade is 1.5; Mile is 2.5. | Token = Var  Token = Double  Token = id  Token = is  Token = doublelit  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | Token = Var  Token = Double  Token = id  Token = is  Token = doublelit  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | **OK** |
| Var Double Grade; Mile is 2.5. | Token = Var  Token = Double  Token = id  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | Token = Var  Token = Double  Token = id  Token = ;  Token = id  Token = is  Token = doublelit  Token = . | **OK** |
| Var double Grade is 1.5. | Token = Var  Token = id  Token = is  Token = doublelit  Token = .  Invalid input: double on line 1 | Token = Var  Token = id  Token = is  Token = doublelit  Token = .  Invalid input: double on line 1 | **OK** |
| Var Char Letter is ‘A’. | Token = Var  Token = Char  Token = id  Token = is  Token = charlit  Token = . | Token = Var  Token = Char  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Var Char Letter is ‘A’; Vowel is ‘E’. | Token = Var  Token = Char  Token = id  Token = is  Token = charlit  Token = ;  Token = id  Token = is  Token = charlit  Token = . | Token = Var  Token = Char  Token = id  Token = is  Token = charlit  Token = ;  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Var Char Letter; Vowel is ‘E’. | Token = Var  Token = Char  Token = id  Token = ;  Token = id  Token = is  Token = charlit  Token = . | Token = Var  Token = Char  Token = id  Token = ;  Token = id  Token = is  Token = charlit  Token = . | **OK** |
| Var char Letter is ‘AA’. | Token = Var  Token = id  Token = is  Token = .  Invalid input: char on line 1  Invalid input: ‘AA’ on line 1 | Token = Var  Token = id  Token = is  Token = .  Invalid input: char on line 1  Invalid input: ‘AA’ on line 1 | **OK** |
| Var String Name is “Joe”. | Token = Var  Token = String  Token = id  Token = is  Token = stringlit  Token = . | Token = Var  Token = String  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Var String Name is “Joe”; Surname is “Sam”. | Token = Var  Token = String  Token = id  Token = is  Token = stringlit  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | Token = Var  Token = String  Token = id  Token = is  Token = stringlit  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Var String Name; Surname is “Sam”. | Token = Var  Token = String  Token = id  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | Token = Var  Token = String  Token = id  Token = ;  Token = id  Token = is  Token = stringlit  Token = . | **OK** |
| Var string Name is “Joe”. | Token = Var  Token = id  Token = is  Token = stringlit  Token = .  Invalid input: string on line 1 | Token = Var  Token = id  Token = is  Token = stringlit  Token = .  Invalid input: string on line 1 | **OK** |
| Var Boolean Case is No. | Token = Var  Token = Boolean  Token = id  Token = is  Token = boollit  Token = . | Token = Var  Token = Boolean  Token = id  Token = is  Token = No  Token = . | **FAILED** |
| Var Boolean Case is No; Status is Yes. | Token = Var  Token = Boolean  Token = id  Token = is  Token = boollit  Token = ;  Token = id  Token = is  Token = boollit  Token = . | Token = Var  Token = Boolean  Token = id  Token = is  Token = No  Token = ;  Token = id  Token = is  Token = Yes  Token = . | **FAILED** |
| Var Boolean Case; Status is Yes. | Token = Var  Token = Boolean  Token = id  Token = ;  Token = id  Token = is  Token = boollit  Token = . | Token = Var  Token = Boolean  Token = id  Token = ;  Token = id  Token = is  Token = Yes  Token = . | **FAILED** |
| Var boolean Case is No. | Token = Var  Token = id  Token = is  Token = boollit  Token = .  Invalid input: boolean on line 1 | Token = Var  Token = id  Token = is  Token = No  Token = .  Invalid input: boolean on line 1 | **FAILED** |

**Read Statement**Syntax: <input> → Read id

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **Actual Output** | **Remarks** |
| Read Age. | Token = Read  Token = id  Token = . | Token = Read  Token = id  Token = . | **OK** |
| Read Age; Num. | Token = Read  Token = id  Token = ;  Token = id  Token = . | Token = Read  Token = id  Token = ;  Token = id  Token = . | **OK** |
|  |  |  | **OK** |
| Read age. | Token = Read  Token = .  Invalid input: age on line 1 | Token = Read  Token = .  Invalid input: age on line 1 | **OK** |
| Read Age; num. | Token = Read  Token = id  Token = ;  Token = .  Invalid input: num on line 1 | Token = Read  Token = id  Token = ;  Token = .  Invalid input: num on line 1 | **OK** |
| Lead:  Start  Read Age.  End.# | Token = Lead  Token = :  Token = Start  Token = Read  Token = id  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Read  Token = id  Token = .  Token = End  Token = .  Token = # | **OK** |
| Lead:  Start  Read Age; Num.  End.# | Token = Lead  Token = :  Token = Start  Token = Read  Token = id  Token = ;  Token = id  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Read  Token = id  Token = ;  Token = id  Token = .  Token = End  Token = .  Token = # | **OK** |

**Say Statement**Syntax: <output> → Say <input\_statement>

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **Actual Output** | **Remarks** |
| Say “Hello”. | Token = Say  Token = stringlit  Token = . | Token = Say  Token = stringlit  Token = . | **OK** |
| Say “Hello. | Token = Say  Invalid input: “Hello. On line 1 | Token = Say  Invalid input: “Hello. On line 1 | **OK** |
| Say ‘Hello’. | Token = Say  Token = .  Invalid input: ‘Hello’ On line 1 | Token = Say  Token = .  Invalid input: ‘Hello’ On line 1 | **OK** |
| Say “Hello’. | Token = Say  Invalid input: “Hello’. On line 1 | Token = Say  Invalid input: “Hello’. On line 1 | **OK** |
| Say Hello. | Token = Say  Token = id  Token = . | Token = Say  Token = id  Token = . | **OK** |
| Say “Hello”, “World”. | Token = Say  Token = stringlit  Token = ,  Token = stringlit  Token = . | Token = Say  Token = stringlit  Token = ,  Token = stringlit  Token = . | **OK** |
| Say “Hello”, World. | Token = Say  Token = stringlit  Token = ,  Token = id  Token = . | Token = Say  Token = stringlit  Token = ,  Token = id  Token = . | **OK** |
| Say Hello, “World”. | Token = Say  Token = id  Token = ,  Token = stringlit  Token = . | Token = Say  Token = id  Token = ,  Token = stringlit  Token = . | **OK** |
| Lead:  Start  Say “Hello”.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = stringlit  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Say  Token = stringlit  Token = .  Token = End  Token = .  Token = # | **OK** |
| Lead:  Start  Say Hello.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = # | **OK** |
| Say Hello@World. | Token = Say  Token = id  Token = @  Token = id  Token = . | Token = Say  Token = id  Token = @  Token = id  Token = . | **OK** |
| Lead:  Start  Say Hello@world.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = #  Invalid input: @ on line 3  Invalid input: world on line 3 | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = #  Invalid input: @ on line 3  Invalid input: world on line 3 | **OK** |

**Say Statement**Syntax: <output> → Say <input\_statement>

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Input** | **Expected Output** | **Actual Output** | **Remarks** |
| Say “Hello”. | Token = Say  Token = stringlit  Token = . | Token = Say  Token = stringlit  Token = . | **OK** |
| Say “Hello. | Token = Say  Invalid input: “Hello. On line 1 | Token = Say  Invalid input: “Hello. On line 1 | **OK** |
| Say ‘Hello’. | Token = Say  Token = .  Invalid input: ‘Hello’ On line 1 | Token = Say  Token = .  Invalid input: ‘Hello’ On line 1 | **OK** |
| Say “Hello’. | Token = Say  Invalid input: “Hello’. On line 1 | Token = Say  Invalid input: “Hello’. On line 1 | **OK** |
| Say Hello. | Token = Say  Token = id  Token = . | Token = Say  Token = id  Token = . | **OK** |
| Say “Hello”, “World”. | Token = Say  Token = stringlit  Token = ,  Token = stringlit  Token = . | Token = Say  Token = stringlit  Token = ,  Token = stringlit  Token = . | **OK** |
| Say “Hello”, Worl0d. | Token = Say  Token = stringlit  Token = ,  Token = id  Token = . | Token = Say  Token = stringlit  Token = ,  Token = id  Token = . | **OK** |
| Say Hello, “World”. | Token = Say  Token = id  Token = ,  Token = stringlit  Token = . | Token = Say  Token = id  Token = ,  Token = stringlit  Token = . | **OK** |
| Lead:  Start  Say “Hello”.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = stringlit  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Say  Token = stringlit  Token = .  Token = End  Token = .  Token = # | **OK** |
| Lead:  Start  Say Hello.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = # | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = # | **OK** |
| Say Hello@World. | Token = Say  Token = id  Token = @  Token = id  Token = . | Token = Say  Token = id  Token = @  Token = id  Token = . | **OK** |
| Lead:  Start  Say Hello@world.  End.# | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = #  Invalid input: @ on line 3  Invalid input: world on line 3 | Token = Lead  Token = :  Token = Start  Token = Say  Token = id  Token = .  Token = End  Token = .  Token = #  Invalid input: @ on line 3  Invalid input: world on line 3 | **OK** |