LEXICAL TEST SCRIPT

Reserved Words

Sample Input	Expected Output	Actual Output	Remarks
	Reserved	Words	
hold	Proper delimiter expected: hold	Proper delimiter expected: hold	OK
hold␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
hold↵	Proper delimiter expected: hold	Proper delimiter expected: hold	OK
hold:	Proper delimiter expected: hold Invalid input: :	Proper delimiter expected: hold Invalid input: :	OK
hold.	Proper delimiter expected: hold Invalid input: .	Proper delimiter expected: hold Invalid input: .	OK
hold(	Proper delimiter expected: hold Invalid input: (	Proper delimiter expected: hold Invalid input: (	OK
hold?	Proper delimiter expected: hold Invalid input: ?	Proper delimiter expected: hold Invalid input: ?	OK
hold-	Proper delimiter expected: hold Invalid input: -	Proper delimiter expected: hold Invalid input: -	OK
Hold	Invalid input: Hold	Invalid input: Hold	OK
Unit	Invalid input: Unit	Invalid input: Unit	OK
unit␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
unit↵	Proper delimiter expected: unit	Proper delimiter expected: unit	OK
unit:	Proper delimiter expected: unit	Proper delimiter expected: unit	OK

	Invalid input: :	Invalid input: :	
unit.	Proper delimiter expected: unit Invalid input: .	Proper delimiter expected: unit Invalid input: .	OK
unit(	Proper delimiter expected: unit Invalid input: (	Proper delimiter expected: unit Invalid input: (	OK
unit?	Proper delimiter expected: unit Invalid input: ?	Proper delimiter expected: unit Invalid input: ?	OK
unit-	Proper delimiter expected: unit Invalid input: -	Proper delimiter expected: unit Invalid input: -	OK
unitt	Invalid input: unitt	Invalid input: unitt	OK
digit	Proper delimiter expected: digit	Proper delimiter expected: digit	OK
digit␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
digit ↵	Proper delimiter expected: digit	Proper delimiter expected: digit	OK
digit:	Proper delimiter expected: digit Invalid input: :	Proper delimiter expected: digit Invalid input: :	OK
digit.	Proper delimiter expected: digit Invalid input: .	Proper delimiter expected: digit Invalid input: .	OK
digit<	Proper delimiter expected: digit Invalid input: <	Proper delimiter expected: digit Invalid input: <	OK
digit(	Proper delimiter expected: digit Invalid input: (	Proper delimiter expected: digit Invalid input: (	OK
digit?	Proper delimiter expected: digit Invalid input: ?	Proper delimiter expected: digit Invalid input: ?	OK
digit-	Proper delimiter expected: digit Invalid input: -	Proper delimiter expected: digit Invalid input: -	OK

digit/	Proper delimiter expected: digit Invalid input: /	Proper delimiter expected: digit Invalid input: /	OK
digitt	Invalid input: digitt	Invalid input: digitt	OK
joe	Proper delimiter expected: joe	Proper delimiter expected: joe	OK
joe␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
joe↵	Proper delimiter expected: joe	Proper delimiter expected: joe	OK
joe:	Proper delimiter expected: joe Invalid input: :	Proper delimiter expected: joe Invalid input: :	OK
joe.	Proper delimiter expected: joe Invalid input: .	Proper delimiter expected: joe Invalid input: .	OK
joe(	Proper delimiter expected: joe Invalid input: (	Proper delimiter expected: joe Invalid input: (	OK
joe?	Proper delimiter expected: joe Invalid input: ?	Proper delimiter expected: joe Invalid input: ?	OK
joe-	Proper delimiter expected: joe Invalid input: -	Proper delimiter expected: joe Invalid input: -	OK
joee	Invalid input: joee	Invalid input: joee	OK
company	Proper delimiter expected: company	Proper delimiter expected: company	OK
company␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
company ↵	Proper delimiter expected: company	Proper delimiter expected: company	OK
company:	Proper delimiter expected: company Invalid input: :	Proper delimiter expected: company Invalid input: :	OK
company.	Proper delimiter expected: company	Proper delimiter expected: company	OK

	Invalid input: .	Invalid input: .	
company<	Proper delimiter expected: company Invalid input: <	Proper delimiter expected: company Invalid input: <	OK
company(	Proper delimiter expected: company Invalid input: (	Proper delimiter expected: company Invalid input: (	OK
company?	Proper delimiter expected: company Invalid input: ?	Proper delimiter expected: company Invalid input: ?	OK
company-	Proper delimiter expected: company Invalid input: -	Proper delimiter expected: company Invalid input: -	OK
company/	Proper delimiter expected: company Invalid input: /	Proper delimiter expected: company Invalid input: /	OK
companyy	Invalid input: companyy	Invalid input: companyy	OK
response	Proper delimiter expected: response	Proper delimiter expected: response	OK
response␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
response↵	Proper delimiter expected: response	Proper delimiter expected: response	OK
response:	Proper delimiter expected: response Invalid input: :	Proper delimiter expected: response Invalid input: :	OK
response.	Proper delimiter expected: response Invalid input: .	Proper delimiter expected: response Invalid input: .	OK
response<	Proper delimiter expected: response Invalid input: <	Proper delimiter expected: response Invalid input: <	OK

response(	Proper delimiter expected: response Invalid input: (	Proper delimiter expected: response Invalid input: (	OK
response?	Proper delimiter expected: response Invalid input: ?	Proper delimiter expected: response Invalid input: ?	OK
response-	Proper delimiter expected: response Invalid input: -	Proper delimiter expected: response Invalid input: -	OK
response/	Proper delimiter expected: response Invalid input: /	Proper delimiter expected: response Invalid input: /	OK
responsee	Invalid input: responsee	Invalid input: responsee	OK
PrimaryMission	Proper delimiter expected: PrimaryMission	Proper delimiter expected: PrimaryMission	OK
PrimaryMission␣	Proper delimiter expected: PrimaryMission	Proper delimiter expected: PrimaryMission	OK
PrimaryMission↵	Proper delimiter expected: PrimaryMission	Proper delimiter expected: PrimaryMission	OK
PrimaryMission:	Proper delimiter expected: PrimaryMission Invalid input: :	Proper delimiter expected: PrimaryMission Invalid input: :	OK
PrimaryMission.	Proper delimiter expected: PrimaryMission Invalid input: .	Proper delimiter expected: PrimaryMission Invalid input: .	OK
PrimaryMission()	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
PrimaryMission?	Proper delimiter expected: PrimaryMission	Proper delimiter expected: PrimaryMission	OK

	Invalid input: ?	Invalid input: ?	
PrimaryMission-	Proper delimiter expected: PrimaryMission Invalid input: -	Proper delimiter expected: PrimaryMission Invalid input: -	OK
PrimaryMission/	Proper delimiter expected: PrimaryMission Invalid input: /	Proper delimiter expected: PrimaryMission Invalid input: /	OK
PrimaryMissionn	Invalid input: PrimaryMissionn	Invalid input: PrimaryMissionn	OK
post	Proper delimiter expected: post	Proper delimiter expected: post	OK
post␣	Proper delimiter expected: post	Proper delimiter expected: post	OK
post ↵	Proper delimiter expected: post	Proper delimiter expected: post	OK
post:	Proper delimiter expected: post Invalid input: :	Proper delimiter expected: post Invalid input: :	OK
post.	Proper delimiter expected: post Invalid input: .	Proper delimiter expected: post Invalid input: .	OK
post(" ");	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
post?	Proper delimiter expected: post Invalid input: ?	Proper delimiter expected: post Invalid input: ?	OK
post-	Proper delimiter expected: post Invalid input: -	Proper delimiter expected: post Invalid input: -	OK
postt	Invalid input: postt	Invalid input: postt	OK
capture	Invalid input: captured	Invalid input: captured	OK
capture␣	Proper delimiter expected: capture	Proper delimiter expected: capture	OK
capture(#a);	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK

capture:	Proper delimiter expected: capture Invalid input: :	Proper delimiter expected: capture Invalid input: :	OK
capture.	Proper delimiter expected: capture Invalid input: .	Proper delimiter expected: capture Invalid input: .	OK
capture<	Proper delimiter expected: capture Invalid input: <	Proper delimiter expected: capture Invalid input: <	OK
capture(	Proper delimiter expected: capture Invalid input: (	Proper delimiter expected: capture Invalid input: (	OK
capture?	Proper delimiter expected: capture Invalid input: ?	Proper delimiter expected: capture Invalid input: ?	OK
capture-	Proper delimiter expected: capture Invalid input: -	Proper delimiter expected: capture Invalid input: -	OK
capture/	Proper delimiter expected: capture Invalid input: /	Proper delimiter expected: capture Invalid input: /	OK
capturee	Invalid input: capturee	Invalid input: capturee	OK
inorder	Proper delimiter expected: inorder	Proper delimiter expected: inorder	OK
inorder␣	Proper delimiter expected: inorder	Proper delimiter expected: inorder	OK
inorder(a>b) { }	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
inorder:	Proper delimiter expected: inorder Invalid input: :	Proper delimiter expected: inorder Invalid input: :	OK
inorder.	Proper delimiter expected: inorder Invalid input: .	Proper delimiter expected: inorder Invalid input: .	OK
inorder<	Proper delimiter expected: inorder Invalid input: <	Proper delimiter expected: inorder Invalid input: <	OK
inorder(	Proper delimiter expected: inorder	Proper delimiter expected: inorder	OK

	Invalid input: (	Invalid input: (	
inorder?	Proper delimiter expected: inorder Invalid input: ?	Proper delimiter expected: inorder Invalid input: ?	OK
inorder-	Proper delimiter expected: inorder Invalid input: -	Proper delimiter expected: inorder Invalid input: -	OK
inorder/	Proper delimiter expected: inorder Invalid input: /	Proper delimiter expected: inorder Invalid input: /	OK
Inorderr	Invalid input: inorderr	Invalid input: inorderr	OK
Order	Invalid input: Order	Invalid input: Order	OK
order␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
order↵	Proper delimiter expected: order	Proper delimiter expected: order	OK
order:	Proper delimiter expected: order Invalid input: :	Proper delimiter expected: order Invalid input: :	OK
order.	Proper delimiter expected: order Invalid input: .	Proper delimiter expected: order Invalid input: .	OK
order<	Proper delimiter expected: order Invalid input: <	Proper delimiter expected: order Invalid input: <	OK
order(	Proper delimiter expected: order Invalid input: (	Proper delimiter expected: order Invalid input: (	OK
order?	Proper delimiter expected: order Invalid input: ?	Proper delimiter expected: order Invalid input: ?	OK
order-	Proper delimiter expected: order Invalid input: -	Proper delimiter expected: order Invalid input: -	OK
order/	Proper delimiter expected: order Invalid input: /	Proper delimiter expected: order Invalid input: /	OK

orderr	Invalid input: orderr	Invalid input: orderr	OK
Inquire	Invalid input: Inquire	Invalid input: Inquire	OK
inquire␣	Proper delimiter expected: inquire	Proper delimiter expected: inquire	OK
inquire(i=0;i<3;i++) { }	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
inquire(	Invalid input: (	Invalid input: (	OK
inquire:	Proper delimiter expected: inquire Invalid input: :	Proper delimiter expected: inquire Invalid input: :	OK
inquire.	Proper delimiter expected: inquire Invalid input: .	Proper delimiter expected: inquire Invalid input: .	OK
inquire?	Proper delimiter expected: inquire Invalid input: ?	Proper delimiter expected: inquire Invalid input: ?	OK
inquire/	Proper delimiter expected: inquire Invalid input: /	Proper delimiter expected: inquire Invalid input: /	OK
inquire<	Proper delimiter expected: inquire Invalid input: <	Proper delimiter expected: inquire Invalid input: <	OK
inquiree	Invalid input: inquiree	Invalid input: inquiree	OK
go	Proper delimiter expected: go	Proper delimiter expected: go	OK
go␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
go(	Proper delimiter expected: go Invalid input: (	Proper delimiter expected: go Invalid input: (	OK
go:	Proper delimiter expected: go Invalid input: :	Proper delimiter expected: go Invalid input: :	OK
go.	Proper delimiter expected: go Invalid input: .	Proper delimiter expected: go Invalid input: .	OK

go?	Proper delimiter expected: go Invalid input: ?	Proper delimiter expected: go Invalid input: ?	OK
go/	Proper delimiter expected: go Invalid input: /	Proper delimiter expected: go Invalid input: /	OK
go<	Proper delimiter expected: go Invalid input: <	Proper delimiter expected: go Invalid input: <	OK
goo	Invalid input: goo	Invalid input: goo	OK
phase	Proper delimiter expected: phase	Proper delimiter expected: phase	OK
phase␣	Proper delimiter expected: phase	Proper delimiter expected: phase	OK
phase ←	Proper delimiter expected: phase	Proper delimiter expected: phase	OK
phase(a>b) { }	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
phase:	Proper delimiter expected: phase Invalid input: :	Proper delimiter expected: phase Invalid input: :	OK
phase.	Proper delimiter expected: phase Invalid input: .	Proper delimiter expected: phase Invalid input: .	OK
phase?	Proper delimiter expected: phase Invalid input: ?	Proper delimiter expected: phase Invalid input: ?	OK
phase/	Proper delimiter expected: phase Invalid input: /	Proper delimiter expected: phase Invalid input: /	OK
phase<	Proper delimiter expected: phase Invalid input: <	Proper delimiter expected: phase Invalid input: <	OK
phasee	Invalid input: phasee	Invalid input: phasee	OK
miss	Proper delimiter expected: miss	Proper delimiter expected: miss	OK

miss␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
miss' ←	Proper delimiter expected: miss	Proper delimiter expected: miss	OK
miss(	Proper delimiter expected: miss Invalid input: (	Proper delimiter expected: miss Invalid input: (	OK
miss:	Proper delimiter expected: miss Invalid input: :	Proper delimiter expected: miss Invalid input: :	OK
miss.	Proper delimiter expected: miss Invalid input: .	Proper delimiter expected: miss Invalid input: .	OK
miss?	Proper delimiter expected: miss Invalid input: ?	Proper delimiter expected: miss Invalid input: ?	OK
miss/	Proper delimiter expected: miss Invalid input: /	Proper delimiter expected: miss Invalid input: /	OK
miss<	Proper delimiter expected: miss Invalid input: <	Proper delimiter expected: miss Invalid input: <	OK
misss	Invalid input: misss	Invalid input: misss	OK
backup	Proper delimiter expected: backup	Proper delimiter expected: backup	OK
backup␣	Proper delimiter expected: backup	Proper delimiter expected: backup	OK
backup();	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
backup(	Invalid input: (	Invalid input: (	OK
backup:	Proper delimiter expected: backup Invalid input: :	Proper delimiter expected: backup Invalid input: :	OK
backup.	Proper delimiter expected: backup Invalid input: .	Proper delimiter expected: backup Invalid input: .	OK
backup?	Proper delimiter expected: backup	Proper delimiter expected: backup	OK

	Invalid input: :	Invalid input: :	
backup/	Proper delimiter expected: backup Invalid input: /	Proper delimiter expected: backup Invalid input: /	OK
backup<	Proper delimiter expected: backup Invalid input: <	Proper delimiter expected: backup Invalid input: <	OK
backupp	Invalid input: backupp	Invalid input: backupp	OK
campaign	Proper delimiter expected: campaign	Proper delimiter expected: campaign	OK
campaign␣	Proper delimiter expected: campaign	Proper delimiter expected: campaign	OK
campaign ←	Proper delimiter expected: campaign	Proper delimiter expected: campaign	OK
campaign(switch) {	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
campaign:	Proper delimiter expected: campaign Invalid input: :	Proper delimiter expected: campaign Invalid input: :	OK
campaign.	Proper delimiter expected: campaign Invalid input: .	Proper delimiter expected: campaign Invalid input: .	OK
campaign?	Proper delimiter expected: campaign Invalid input: ?	Proper delimiter expected: campaign Invalid input: ?	OK
campaign/	Proper delimiter expected: campaign Invalid input: /	Proper delimiter expected: campaign Invalid input: /	OK
campaign<	Proper delimiter expected: campaign Invalid input: <	Proper delimiter expected: campaign Invalid input: <	OK

campaignn	Invalid input: campaignn	Invalid input: campaignn	OK
operation	Proper delimiter expected: operation	Proper delimiter expected: operation	OK
operation␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
operation:	Proper delimiter expected: operation Invalid input: :	Proper delimiter expected: operation Invalid input: :	OK
operation(	Proper delimiter expected: operation Invalid input: (	Proper delimiter expected: operation Invalid input: (	OK
operation:	Proper delimiter expected: operation Invalid input: :	Proper delimiter expected: operation Invalid input: :	OK
operation.	Proper delimiter expected: operation Invalid input: .	Proper delimiter expected: operation Invalid input: .	OK
operation?	Proper delimiter expected: operation Invalid input: ?	Proper delimiter expected: operation Invalid input: ?	OK
operation/	Proper delimiter expected: operation Invalid input: /	Proper delimiter expected: operation Invalid input: /	OK
operation<	Proper delimiter expected: operation Invalid input: <	Proper delimiter expected: operation Invalid input: <	OK
operationn	Invalid input: operationn	Invalid input: operationn	OK
abort	Proper delimiter expected: abort	Proper delimiter expected: abort	OK

abort␣	Proper delimiter expected: abort	Proper delimiter expected: abort	OK
abort();	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
abort(	Invalid input: (	Invalid input: (	OK
abort:	Proper delimiter expected: abort Invalid input: :	Proper delimiter expected: abort Invalid input: :	OK
abort.	Proper delimiter expected: abort Invalid input: .	Proper delimiter expected: abort Invalid input: .	OK
abort?	Proper delimiter expected: abort Invalid input: ?	Proper delimiter expected: abort Invalid input: ?	OK
abort/	Proper delimiter expected: abort Invalid input: /	Proper delimiter expected: abort Invalid input: /	OK
abort<	Proper delimiter expected: abort Invalid input: <	Proper delimiter expected: abort Invalid input: <	OK
abortt	Invalid input: abortt	Invalid input: abortt	OK
struct	Proper delimiter expected: struct	Proper delimiter expected: struct	OK
struct␣	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
struct ←	Proper delimiter expected: struct	Proper delimiter expected: struct	OK
struct(	Proper delimiter expected: struct Invalid input: (	Proper delimiter expected: struct Invalid input: (	OK
struct:	Proper delimiter expected: struct Invalid input: :	Proper delimiter expected: struct Invalid input: :	OK
struct.	Proper delimiter expected: struct Invalid input: .	Proper delimiter expected: struct Invalid input: .	OK
struct?	Proper delimiter expected: struct	Proper delimiter expected: struct	OK

	Invalid input: ?	Invalid input: ?	
struct/	Proper delimiter expected: struct Invalid input: /	Proper delimiter expected: struct Invalid input: /	OK
struct<	Proper delimiter expected: struct Invalid input: <	Proper delimiter expected: struct Invalid input: <	OK
structt	Invalid input: structt	Invalid input: structt	OK
deploy	Proper delimiter expected: deploy	Proper delimiter expected: deploy	OK
commence	Proper delimiter expected: commence	Proper delimiter expected: commence	OK
commence.	Proper delimiter expected: commence Invalid input: .	Proper delimiter expected: commence Invalid input: .	OK
commence,	Proper delimiter expected: commence Invalid input: ,	Proper delimiter expected: commence Invalid input: ,	OK
commence:	Proper delimiter expected: commence Invalid input: :	Proper delimiter expected: commence Invalid input: :	OK
commence;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
commence(	Proper delimiter expected: commence Invalid input: (	Proper delimiter expected: commence Invalid input: (	OK
deploy␣	Proper delimiter expected: deploy	Proper delimiter expected: deploy	OK
deploy();	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
deploy(	Proper delimiter expected: deploy Invalid input: (	Proper delimiter expected: deploy Invalid input: (	OK

deploy:	Proper delimiter expected: deploy Invalid input: :	Proper delimiter expected: deploy Invalid input: :	OK
deploy.	Proper delimiter expected: deploy Invalid input: .	Proper delimiter expected: deploy Invalid input: .	OK
deploy?	Proper delimiter expected: deploy Invalid input: ?	Proper delimiter expected: deploy Invalid input: ?	OK
deploy/	Proper delimiter expected: deploy Invalid input: /	Proper delimiter expected: deploy Invalid input: /	OK
deploy<	Proper delimiter expected: deploy Invalid input: <	Proper delimiter expected: deploy Invalid input: <	OK
deployyy	Invalid input: deployyy	Invalid input: deployyy	OK

Unit Literals

Sample Input	Expected Output	Actual Output	Remarks
unit a = 100;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
unit b = one;	Invalid input: one	Invalid input: one	OK
unit c = 7ven;	Invalid input: 7ven	Invalid input: 7ven	OK
unit d = 736;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
unit e = lol;	Invalid input: lol	Invalid input: lol	OK

Digit Literals

Sample Input	Expected Output	Actual Output	Remarks
digit a = 1.0;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
digit b = 3.zero;	Invalid input: 3.zero	Invalid input: 3.zero	OK
digit c = abc;	Invalid input: abc	Invalid input: abc	OK
digit d = 100.03;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
digit e = kiss;	Invalid input: kiss	Invalid input: kiss	OK

Company Literals

Sample Input	Expected Output	Actual Output	Remarks
company a = “World “;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
company b = “123;	Invalid input: “123	Invalid input: “123	OK
company c = “hello;	Invalid input: “hello	Invalid input: “hello	OK
company d = “hello “;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
company e = “ “;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK

Joe Literals

Sample Input	Expected Output	Actual Output	Remarks
joe a = ‘AB ‘;	Invalid input: ’AB’	Invalid input: ’AB’	OK
joe b = ‘ ‘;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
joe c = ‘ C.C ‘;	Invalid input: ’C.C’	Invalid input: ’C.C’	OK
joe d = ‘ 1D ‘;	Invalid input: ’1D’	Invalid input: ’1D’	OK
joe e = ‘ E ‘;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK

Response Literals

Sample Input	Expected Output	Actual Output	Remarks
response a = AFFIRMATIVE;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK
response b = 4ffirm4tiv3;	Invalid input: 4ffirm4tiv3	Invalid input: 4ffirm4tiv3	OK
response c = Affirmative;	Invalid input: Affirmative	Invalid input: Affirmative	OK
response d = Negative;	Invalid input: Negative	Invalid input: Negative	OK
response e = NEGATIVE;	Lexical Analyzer Successfully Executed.	Lexical Analyzer Successfully Executed.	OK