

Output:

```
+++  
;;  
[Line 2 | Token 1] PUNCTUATION   ;;; (Total Tokens: 1)  
+++  
[Line 3 | Token 2] OPERATOR      _+++ (Total Tokens: 2)  
giveback  
check  
[Line 5 | Token 3] KEYWORD       check (Total Tokens: 3)  
otherwise  
[Line 6 | Token 4] KEYWORD       otherwise (Total Tokens: 4)
```

Language Overview:

Language Name: NovaLang.

Nova Lang is a structured, easy to understand programming Language inspired by C++. Its goal is to make programming simpler for beginners by using readable and meaningful English word as keywords. The syntax focuses on clarity while keeping logical structure similar to C++. Nova Lang is perfect for Learning Loops, conditions, and functions without complex syntax.

Keywords:

Keyword	meaning	Example Usage
Check	if condition	check (x > 0) {...}
Otherwise	else condition	otherwise {...}
repeatwhile	while loop	repeat While (i < 10) {...}.
do Task	do loop	doTask (...) until (x==0);
Until	Loop terminate	do Task {...} until (flag == true);
define	define a function	define add (a, b) { give Back a+b;}

give Back	return value	giveback result;
readln	take user Input	readln(age);
Show Out	print output	Show Out ("Hello NovaLong");
number	numeric datatype	number X = 25;
word	string data type	word name = "Sara";
Logic	boolean type	logic valid = true;
constant	fixed value	constant PI=3.14;
continue To	continue to next Loop iteration	continue To;
Stop All	stop program	stop All.

Operators and Punctuation:

Operators:

Category	Symbol	Description.
Operator	_+++	Performs Addition between two numeric or String Values.
Operator	— _	Performs subtraction between two values
Operator	— ***	Multiplies numeric Values to produce a product.
Assign_op	==	Equals too
not_equal_op	!=	Not equals too
less_op	<<	Less than

greater_op	>>	Greater than
Operator	_///_	Divides one number by another and returns the quotient

Punctuation:

Category	Symbol	Description
Punctuation	_<<<	indicates the beginning of a code block (like{).
Punctuation	>>>_	Marks the end of a code block (like }).
Punctuation	_:::_	separate variable, function parameters, or expression

Regular Expressions:

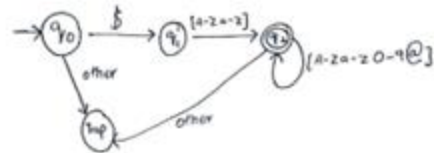
Token Type	Regular Expression	Example
identifier	\\$[A-Z a-Z] [A-Z a-z 0-9. @]*	\$count, \$value, \$user@name
Integer No	[1-9][0-9]*(-[0-9]+)*	45,123_456.
Float No	[0-9]+-[0-9]+\.[0-9]+	12_45.67,3_0.14
Keyword	(check otherwise repeat while doTask until define giveBack)	cheek (x>0), otherwise {...}, repeat while (i<10).
Operator (Addition)	_ \+ \+ \+	_+++
Operator (subtraction)	— _	— _
Operator (Multiplication)	_***_	_***_

Operator (Division)	<code>___</code>	<code>_///_</code>
Punctuation (Block start)	<code>_<<<</code>	<code>_<<<</code>
Punctuation (Block End)	<code>>>>_</code>	<code>>>>_</code>
Punctuation (separator)	<code>_:::_</code>	<code>_:::_</code>
Comment (single Line)	<code>W@.*</code>	<code>//@ This is a Single line Comment</code>
Comment (Multi Line)	<code>V@[\\S\\S]*?@V</code>	<code>/@ This is a Multi line Comment</code>
String Literal	<code>\"[A-Z a-z 0-9_.,! ?@#%&*()\\[\\]-\\]\"</code>	<code>\"Hello NovaLang!\", Total_100</code>

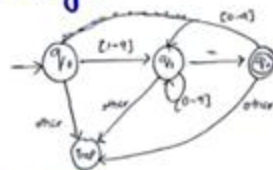
State diagrams:

Finite Automata (State Diagram).

1) identifier:-



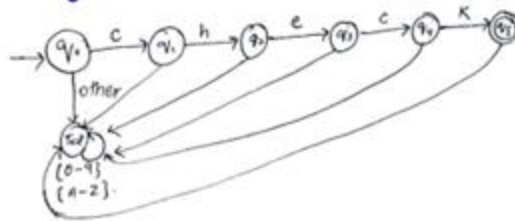
2) Integer



3) Float :-

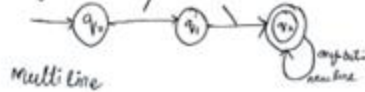


4) Key word:-

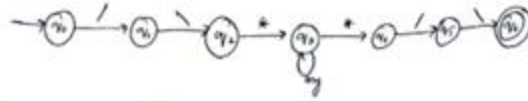


5) Comment :-

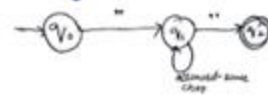
single-line



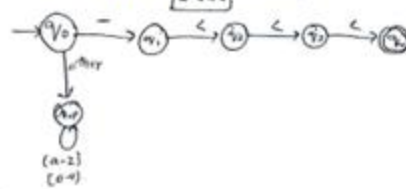
Multi line



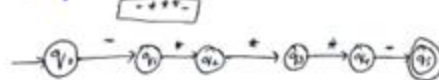
6) String Literal :-



7) Delimiter :- (function)



8) Operator :-



Explanation:

The Keywords of NovaLang were chosen to make Programming more readable and beginner friendly. Each word such as check, repeat while and showOut clearly describes its purpose. These simple English-based Keywords help learners understand logic easily. Novalang uses unique syntax symbols `_<<<`, `_;;` and `_+++` instead of traditional braces or operators.

Such features make NovaLang distinct from C++ and other complex languages. The Language is built to reduce confusion and increase understanding. Overall, Nova Lang is modern, expressive and ideal for students learning programming concepts.