operator	name
(	left parenthesis
)	right parenthesis
[	left bracket
]	right bracket
.	structure member
->	structure pointer
sizeof	sizeof
,	comma
!	negate
~	1s complement
>>	shift right
<<	shift left
^	bitwise XOR
|	bitwise OR
++	increment
--	decrement
+	addition
/	division
||	logical OR
&&	logical AND
?	conditional true
:	conditional false
==	equality test
!=	inequality test
<	less than test
>	greater than test
<=	less than or equal test
>=	greater than or equal test
=	assignment
+=	plus equals
-=	minus equals
*=	times equals
/=	divide equals
%=	mod equals
>>=	shift right equals
<<=	shift left equals
&=	bitwise AND equals
^=	bitwise XOR equals
|=	bitwise OR equals
&	AND/address operator
-	minus/subtract operator
*	multiply/dereference operator

Examples:

```
./tokenizer “array[xyz ] += pi 3.14159e-10”
word: "array"
left bracket: "["
word: "xyz"
right bracket: "]"
plus equals: "+="
word: "pi"
float: "3.141519e-10"
```

```
./tokenzier "numbers 972 0x972 A8 0XA8 XA8 0972 072"
word: "numbers"
decimal integer: "972"
hexadecimal integer: "0x972"
word: "A8"
hexadecimal integer: "0XA8"
word: "XA8"
decimal integer: "0972"
octal integer: "072"


./tokenizer "3.1 03.1 x3.1 0x3.1 30x1.x"
floating point: "3.1"
floating point: "03.1"
word: "x3"
structure member: "."
decimal integer: "1"
hexadecimal integer: "0x3"
structure member: "."
decimal integer: "1"
decimal integer: "30"
word: "x1"
structure member: "x"
word: "x"


./tokenizer "+,++,+++,++++,+++++,++=,+++=="
addition: "+"
comma: ","
increment: "++"
comma: ","
increment: "++"
addition: "+"
comma: ","
increment: "++"
increment: "++"
comma: ","
increment: "++"
increment: "++"
addition: "+"
comma: ","
increment: "++"
assignment: "="
comma: ","
increment: "++"
plus equals: "+="
assignment: "="
```

notes on second example:

Why is "0XA8" a hex number and "XA8" is not?

- any token starting with zero and either a lower- or upper-case 'X' and then followed by numeric characters 0 through 9 or alphabtic A through F (or a through f) is hexadecimal. The second string does not start with a zero, so it is an alphabetic character followed by alphabetic or numeric characters, which is a word.

Why is "0972" a decimal integer while "072" is octal?

- any token starting with a zero and followed by characters 0 through 7 is octal, however "0972" contains a '9'. which is out of bounds for octal, however it is a token consisting of only numeric characters, so it is a decimal integer. "072" starts with a zero and consists of numeric characters 7 or less (0 through 7), so it is octal.