Operators

= += -= *= /= %= &= |= ^= <<= >>= ++ -- - ! + - * / % & | ^ ~ << >> @ $ + == != < > <= >= && ||

Operator: =
Name: assignment
Description: Assigns the value on the right to the variable on the left.
Example:
```
%x = 1;%x = 1
```

Operator: +=
Name: additive assignment
Description: Adds the value on the right to the variable on the left and assigns the result to the variable on the left.
Example:
```
%x = 1; %x += 5;%x = 6
```

Operator: -=
Name: subtractive assignment
Description: Subtracts the value on the right from the variable on the left and assigns the result to the variable on the left.
Example:
```
%x = 5; %x -= 2;%x = 3
```

Operator: *=
Name: multiplicative assignment
Description: Multiplies the value on the right by the variable on the left and assigns the result to the variable on the left.
Example:
```
%x = 2; %x *= 5;%x = 10
```

Operator: /=
Name: divisive assignment
Description: Divides the variable on the left by the value on the right and assigns the result to the variable on the left.
Example:
```
%x = 10; %x /= 5;%x = 2
```

Operator: %=
Name: remainder assignment
Description: Divides the variable on the left by the value on the right and assigns the remainder to the variable on the left.
Example:
```
%x = 10; %x %= 4;%x = 2
```

Operator: &=
Name: bitwise AND assignment
Description: Bitwise ANDs the variable on the left with the value on the right and assigns the result to

the variable on the left. Uses a signed 32 bit word.
Example:
```
%x1 = 12; // 00001100 %x2 = 9; // 00001001 %x1 &= %x2;%x1 = 8
(00001000)
```
Operator: |=
Name: bitwise OR assignment
Description: Bitwise ORs the variable on the left with the value on the right and assigns the result to the variable on the left. Uses a signed 32 bit word.
Example:
```
%x1 = 12; // 00001100 %x2 = 9; // 00001001 %x1 |= %x2;%x1 = 13
(00001101)
```
Operator: ^=
Name: bitwise XOR assignment
Description: Bitwise XORs the variable on the left with the value on the right and assigns the result to the variable on the left. Uses a signed 32 bit word.
Example:
```
%x1 = 10; // 00001010 %x2 = 12; // 00001100 %x1 ^= %x2;%x1 = 6
(00000110)
```
Operator: <<=
Name: shift left assignment
Description: Shifts left the the bits of the variable on the left a number of places equal to the value on the right and assigns the result to the variable on the left. Uses a signed 32 bit word. Bits shifted off the left hand end are lost, and zeros are shifted in at the right. A shifted bit will eventually shift through the sign bit (the leftmost bit) and off the left end.
Example:
```
%x1 = 10; // 00001010 %x1 <<= 2;%x1 = 40 (00101000)
```
Operator: >>=
Name: shift right assignment
Description: Shifts right the the bits of the variable on the left a number of places equal to the value on the right and assigns the result to the variable on the left. Uses a signed 32 bit word. Bits shifted off the right hand end are lost, and zeros are shifted in at the left. The sign bit (the leftmost bit) is never right shifted.
Example:
```
%x1 = 10; // 00001010 %x1 >>= 2;%x1 = 2 (00000010)
```
Operator: ++
Name: increment
Description: Increments the value of the variable by 1. Must be used postfix, but always functions as prefix.
Example:
```
%x1 = 10; %x1++;%x1 = 11
```
Operator: --
Name: decrement
Description: Decrements the value of the variable by 1. Must be used postfix, but always functions as prefix.
Example:

```
%x1 = 10; %x1--;%x1 = 9
```

Operator: -

Name: unary negation
Description: Negates the value of the variable.
Example:
```
%x1 = 10; %x2 = -%x1;%x2 = -10
```

Operator: !

Name: boolean NOT
Description: Performs a NOT on the associated variable. The variable must contain a boolean value.
Example:
```
%x1 = true; %x2 = !%x1;%x2 = false
```

Operator: +

Name: addition
Description: Adds the value on the right to the value on the left.
Example:
```
%x1 = 1 + 2;%x1 = 3
```

Operator: -

Name: subtraction
Description: Subtracts the value on the right from the value on the left.
Example:
```
%x1 = 3 - 2;%x1 = 1
```

Operator: *

Name: multiplication
Description: Multiplies the value on the right with the value on the left.
Example:
```
%x1 = 2 * 3;%x1 = 6
```

Operator: /

Name: division
Description: Divides the value on the left by the value on the right.
Example:
```
%x1 = 6 / 2;%x1 = 3
```

Operator: %

Name: remainder
Description: Divides the value on the left by the value on the right and returns the remainder.
Example:
```
%x1 = 10 / 4;%x1 = 2
```

Operator: &

Name: bitwise AND
Description: Bitwise ANDs the value on the left with the value on the right. Uses a signed 32 bit word.
Example:
```
%x1 = 12; // 00001100 %x2 = 9; // 00001001 %x3 = %x1 & %x2;%x3 = 8
(00001000)
```

Operator: |

Name: bitwise OR

Description: Bitwise ORs the value on the left with the value on the right. Uses a signed 32 bit word.
Example:
```
%x1 = 12; // 00001100 %x2 = 9; // 00001001 %x3 = %x1 | %x2;%x3 = 13 (00001101)
```

Operator: ^
Name: bitwise XOR
Description: Bitwise XORs the value on the left with the value on the right. Uses a signed 32 bit word.
Example:
```
%x1 = 10; // 00001010 %x2 = 12; // 00001100 %x3 = %x1 ^ %x2;%x3 = 6 (00000110)
```

Operator: ~
Name: bitwise complement
Description: Reurns the bitwise complement of the value to the right.The bitwise complement is (-x)-1, where x is the value operated on.
Example:
```
%x1 = 2; %x2 = ~%x1;%x2 = -3
```

Operator: <<
Name: shift left
Description: Shifts left the the bits of the value on the left a number of places equal to the value on the right. Uses a signed 32 bit word. Bits shifted off the left hand end are lost, and zeros are shifted in at the right. A shifted bit will eventually shift through the sign bit (the leftmost bit) and off the left end.
Example:
```
%x1 = 10; // 00001010 %x2 = %x1 << 2;%x2 = 40 (00101000)
```

Operator: >>
Name: shift right
Description: Shifts right the the bits of the value on the left a number of places equal to the value on the right. Uses a signed 32 bit word. Bits shifted off the right hand end are lost, and zeros are shifted in at the left. The sign bit (the leftmost bit) is never right shifted.
Example:
```
%x1 = 10; // 00001010 %x2 = %x1 >> 2;%x2 = 2 (00000010)
```

Operator: @
Name: string concatenation
Description: Concatenates the value on the right to the end of the value on the left. If either value is not a string, it will be converted.
Example:
```
%x1 = "con"; %x2 = "catenate"; %x3 = %x1 @ %x2;%x3 = "concatenate"
```

Operator: $+
Name: string concatenation
Description: Just like @, concatenates the value on the right to the end of the value on the left. If either value is not a string, it will be converted.
Example:
```
%x1 = "con"; %x2 = "catenate"; %x3 = %x1 $+ %x2;%x3 = "concatenate"
```

Operator: ==
Name: equal conditional
Description: Returns true if the value on the right is equal to the value on the left; otherwise returns

false.
Examples:
```
%x = (1 == 1);%x = true%x = (1 == 2);%x = false
```

Operator: !=

Name: not equal conditional
Description: Returns true if the value on the right is not equal to the value on the left; otherwise returns false.
Examples:
```
%x = (1 != 2);%x = true%x = (1 != 1);%x = false
```

Operator: <

Name: less than conditional
Description: Returns true if the value on the left is less than the value on the right; otherwise returns false.
Examples:
```
%x = (1 < 2);%x = true%x = (2 < 1);%x = false%x = (1 < 1);%x = false
```

Operator: >

Name: greater than conditional
Description: Returns true if the value on the left is greater than the value on the right; otherwise returns false.
Examples:
```
%x = (2 > 1);%x = true%x = (1 > 2);%x = false%x = (1 > 1);%x = false
```

Operator: <=

Name: less than or equal to conditional
Description: Returns true if the value on the left is less than or equal to the value on the right; otherwise returns false.
Examples:
```
%x = (1 <= 2);%x = true%x = (1 <= 1);%x = true%x = (2 <= 1);%x =
false
```

Operator: >=

Name: less than or equal to conditional
Description: Returns true if the value on the left is greater than or equal to the value on the right; otherwise returns false.
Examples:
```
%x = (1 >= 1);%x = true%x = (2 >= 1);%x = true%x = (1 >= 2);%x =
false
```

Operator: &&

Name: logical AND
Description: Returns true if both the value on the left side evaluates to true and the value on the right side evaluates to true; otherwise returns false.
Examples:
```
%x = (1 = 1 && 2 = 2);%x = true%x = (1 = 1 && 1 = 2);%x = false%x =
(1 = 2 && 1 = 1);%x = false%x = (1 = 2 && 2 = 3);%x = false
```

Operator: ||

Name: logical OR
Description: Returns true if either the value on the left side evaluates to true or the value on the right

side evaluates to true; otherwise returns false.

Examples:

```
%x = (1 = 1 || 2 = 2);%x = true%x = (1 = 1 || 1 = 2);%x = true%x = (1
= 2 || 1 = 1);%x = true%x = (1 = 2 || 2 = 3);%x = false
```