```
input  /asking user for input
store a /storing user's first input into variable named a
input /asking user for operand
store o  /storing user's operand into variable named o
input /asking user for input
store b /storing user's first input into variable named b

load addSign /load the default hex code for addition, which is 43
subt o /take the user's operand's hex code and subtract it from 43
Skipcond 400 / if 43-o is = 0, then do addition.
jump multiplication /if its not 0, then jump to the multiplication procedure
load a /load value a into AC
add b / add value b into AC
store r / store a + b into variable r
load r /load r into AC
output /output r into display
halt /terminate the program

subtraction, skipcond 000
jump multiplication
load a
subt b
store r
load r
output
halt

multiplication, Skipcond 800 /jump to this procedure if 43-0 is not 0, if the 43-0 is more
than 0, than multiply the two variables
jump subtraction /if 43-0 is less than 0, then jump to the subtraction procedure
load b /load value b into AC
Skipcond 000 /if value of b is less than 0, than continue with multiplication
jump nonneg /if its a postitive number, then jump to nonneg procedure
subt b / subtract value b
subt b  / subtract value b
store b /store new value into b
clear /clear all
add one /add number 1
store negflag /store value in negflag
clear /clear all
jump loop /jump to the loop for positive number multiplication

nonneg, clear /if both numbers are positive, clear the AC and follow this procedure
store negflag /store the negflag value into AC
load b /load value b into AC
Skipcond 400 /if value equals to 0, then skip the loop
jump loop /skips the loop if value equals to 0
```

```
jump halt / go to halt procedure, simple display output and store inside the final variable

loop, load r /for positive number multiplication, load value r into AC
add a /add user's first input
store r /store new value into the r variable
load b /load the value b into the AC
subt one /subtract value from 1
store b /store new value into variable b
Skipcond 400 /if the new value equals to 0, then continue with negative number
multiplication
jump loop /if not, jump to the loop procedure

load negflag /procedure for multiplication with negative numbers
Skipcond 800          /if result is more than 0, then skip the halt procedure, and continue
with multiplication
jump halt /jump to display result if the result is not more than 0

load r /load r into AC
subt r /subtract AC from value r
subt r /subtract AC from value r
store r /store new value into variable r

halt, load r     /halt procedure, used to display and store the result from the arithmetic
procedures. Load the value r into AC
output /display output
halt /terminate program

a, dec 0 /used for user's first value
b, dec 0 /used for user's second value
c, dec 0  /used for holding values
o, hex 0  /used for holding user's operand
r,dec 0  /used for holding result
n, dec 0 /used for holding values
m, dec 0 /used for holding values
negflag, DEC 0 /used for multiplication for negative numbers
one, dec 1 /used for subtracting 1 from the given values
addSign, dec 43 /used for holding value of ascii codes for addition
subSign, dec 45 /used for holding value of ascii codes for subtraction
mulSign, dec 42 /used for holding value of ascii codes for multiplication
```