```
endbr64
push rbp
mov rbp rsp
mov [ rbp - 0x10 ] 0x2a
mov edx [ rbp - 0x10 ]
mov eax [ rbp - 0x10 ]
imul eax edx
sub eax 0x5
mov [ rbp - 0xc ] eax
mov edx [ rbp - 0x10 ]
mov eax [ rbp - 0xc ]
and eax edx
mov [ rbp - 0x8 ] eax
mov eax [ rbp - 0x8 ]
add eax 0x1
mov [ rbp - 0x4 ] eax
mov eax 0x0
pop rbp
ret
```

1

```
endbr64
push rbp
mov rbp rsp

mov [ rbp - 0x10 ] 0x2a

mov edx [ rbp - 0x10 ]
mov eax [ rbp - 0x10 ]
imul eax edx
sub eax 0x5
mov [ rbp - 0xc ] eax

mov edx [ rbp - 0x10 ]
mov eax [ rbp - 0xc ]
and eax edx
mov [ rbp - 0x8 ] eax

mov eax [ rbp - 0x8 ]
add eax 0x1
mov [ rbp - 0x4 ] eax

mov eax 0x0
pop rbp
ret
```

2

```
VAR = NUM ;
VAR = ( ( VAR * VAR ) - NUM ) ;
VAR = ( VAR & VAR ) ;
VAR = ( VAR + NUM ) ;
```

3

```
int main(){
int x0 , x1 , x2 , x3 ;
x0 = 42 ;
x1 = ( ( x0 * x0 ) - 5 ) ;
x2 = ( x0 & x1 ) ;
x3 = ( x2 + 1 ) ;
return 0;}
```