III. STRUCTURED ASSEMBLY LANGUAGE PROGRAMMING TECHNIQUES

Modular Programming

Functions: Return Value

High-level PL

subprogram:

```
int sum (int a, int b) {
    return(a + b);
}
```

subprogram call:

```
num = sum(x, y);
```

Assembly

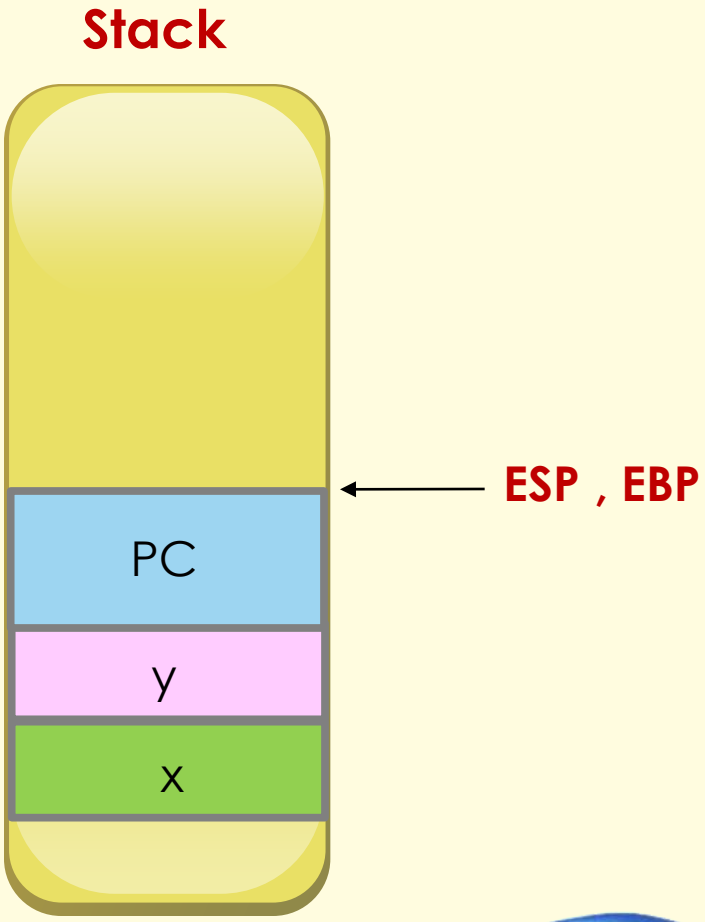
; subprogram call

```
sub esp, 2
push word [x]
push word [y]
call     sum
pop word[num]
```

Functions: Return Value

subprogram:

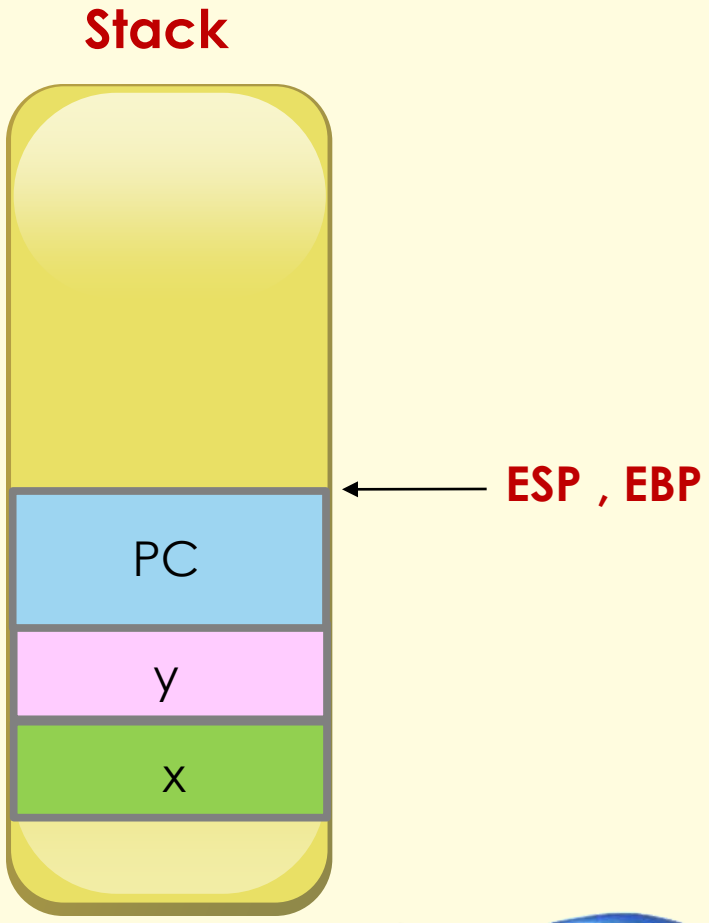
```
int sum (int a, int b) {
   return(a + b);
}

sum:
    mov  ebp, esp
```



Functions: Return Value

subprogram:

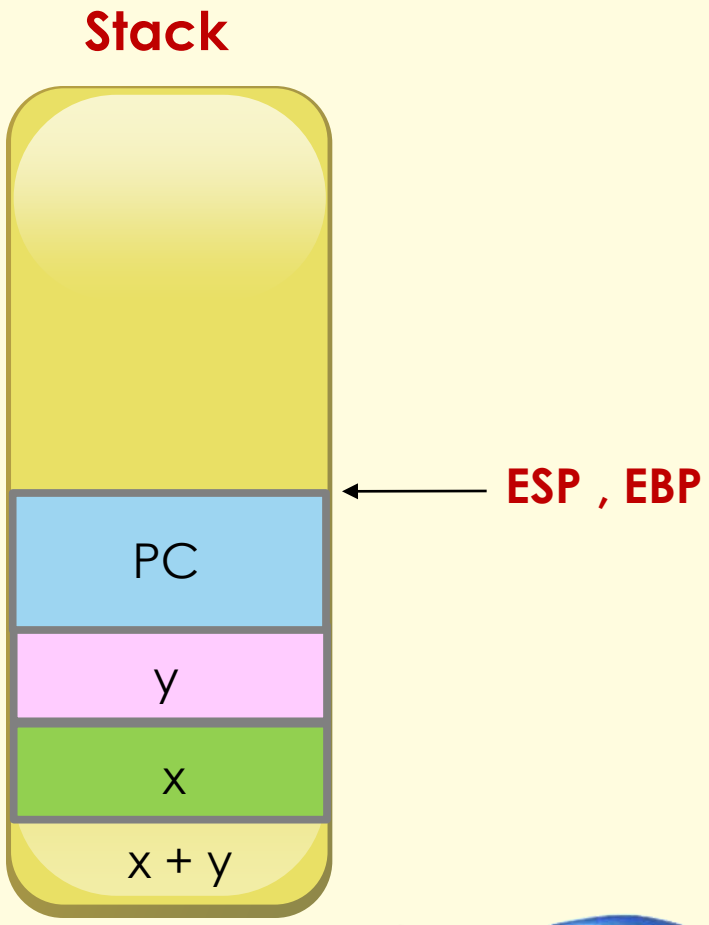
```
int sum (int a, int b) {
   return(a + b);
}

sum:
    mov  ebp, esp
    mov  ax, [ebp + 6]
    add  ax, [ebp + 4]
```



Functions: Return Value

subprogram:

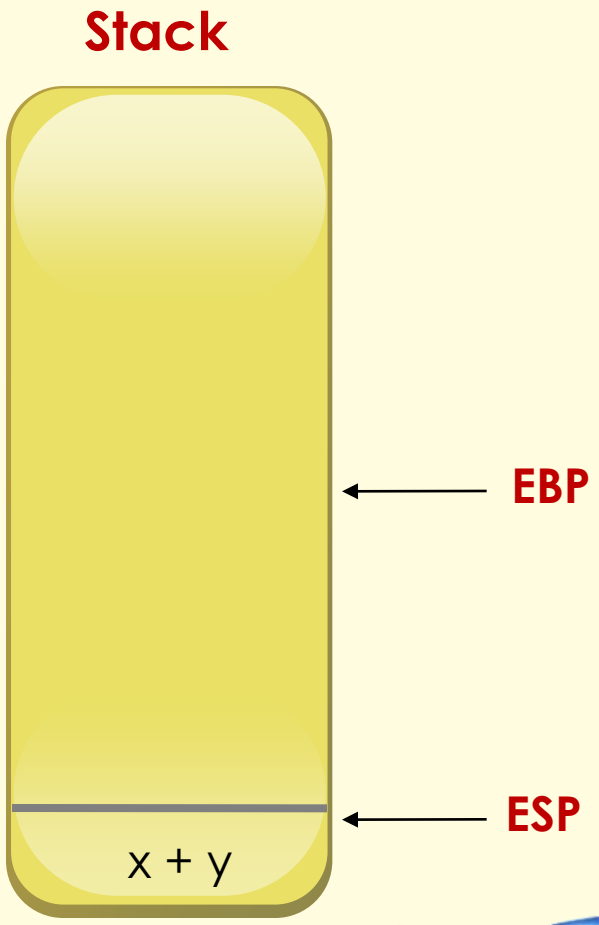
```
int sum (int a, int b) {
   return(a + b);
}

sum:
    mov  ebp, esp
    mov  ax, [ebp + 6]
    add  ax, [ebp + 4]
    mov  [ebp + 8], ax
```



Functions: Return Value

subprogram:

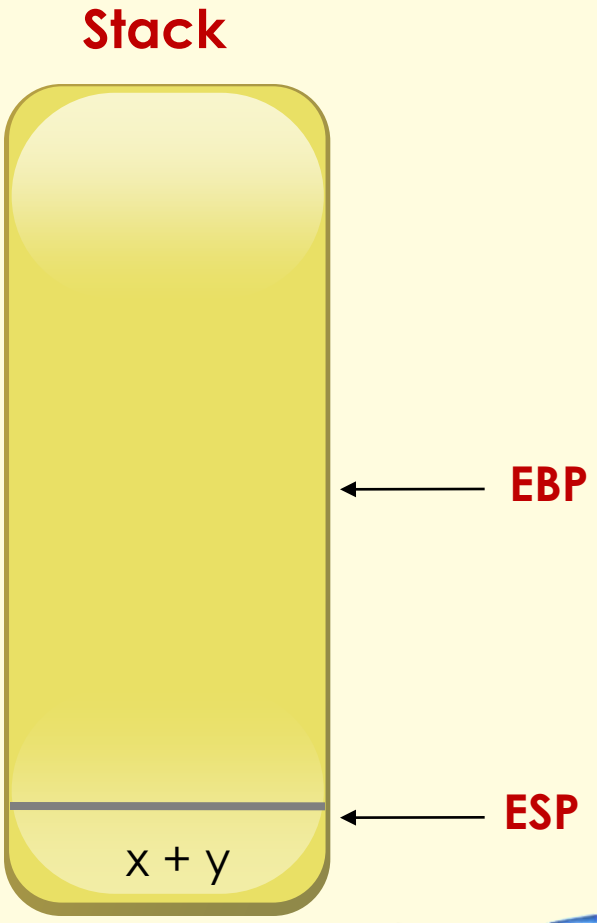
```
int sum (int a, int b) {
   return(a + b);
}

sum:
    mov  ebp, esp
    mov  ax, [ebp + 6]
    add  ax, [ebp + 4]
    mov  [ebp + 8], ax
    ret    4
```



Functions: Return Value

; subprogram call

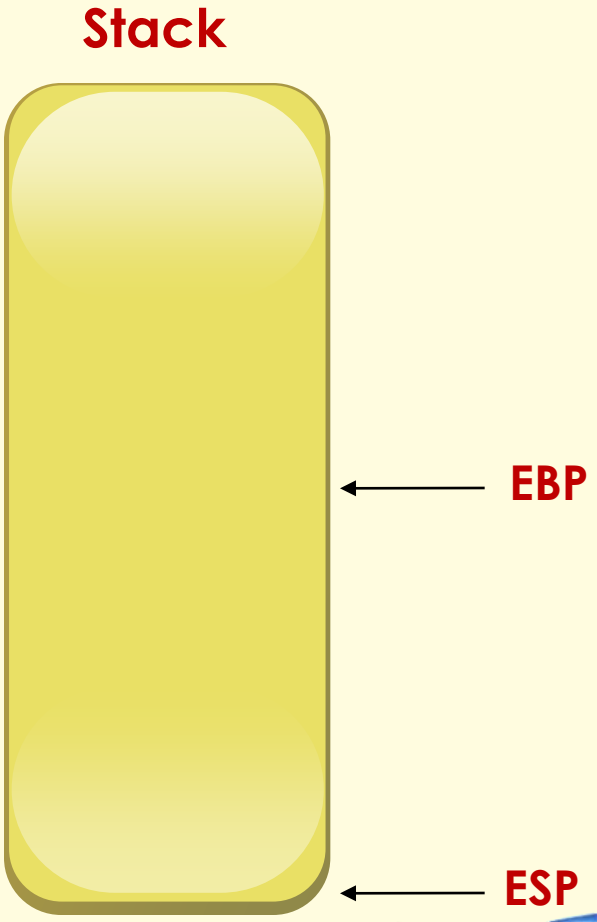
```
sub esp, 2
push word [x]
push word [y]
call    sum
pop word [num]
```



Functions: Return Value

; subprogram call

```
sub esp, 2
push word [x]
push word [y]
call    sum
pop word [num]
```



Functions: Return Value

```
sum:
    mov  ebp, esp              ; create stack frame
    mov  ax, [ebp + 6]         ; retrieve parameter a
    add  ax, [ebp + 4]         ; retrieve parameter b
    mov  [ebp + 8], ax         ; return a + b
    ret  4                     ; return to caller and
                                   clear stack
```

More Examples

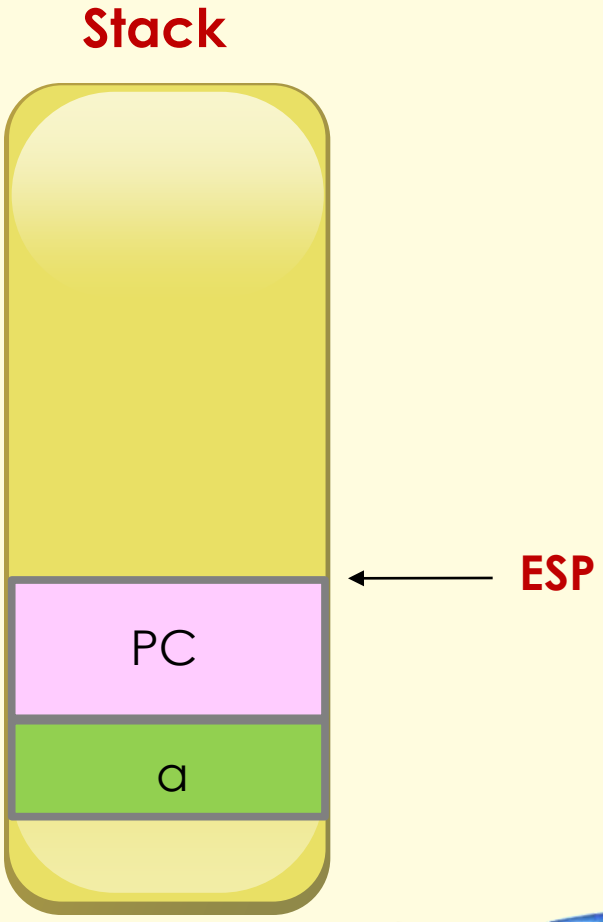
```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}

r = abc (a);
```

```
;subprogram call

sub esp, 2
push word [a]
call abc
pop word [r]
```

Stack

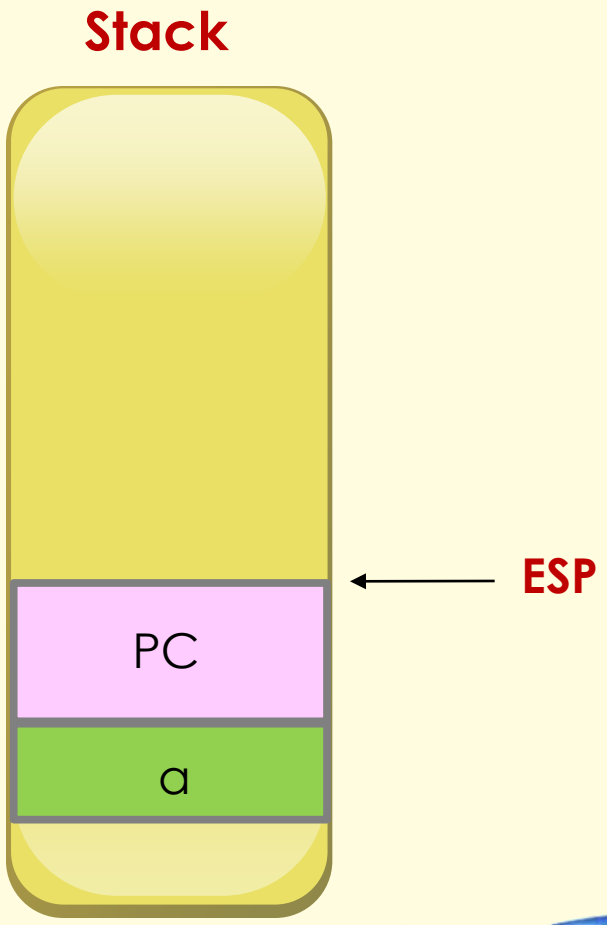
subprogram call:

```
sub esp, 2
push word [a]
call abc
pop word [r]
```



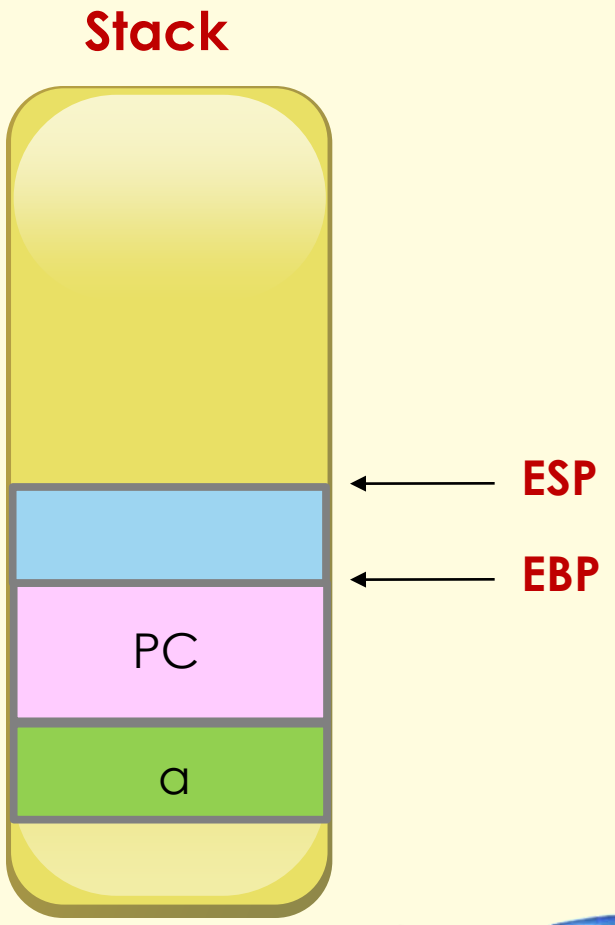
More Examples

```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}
```



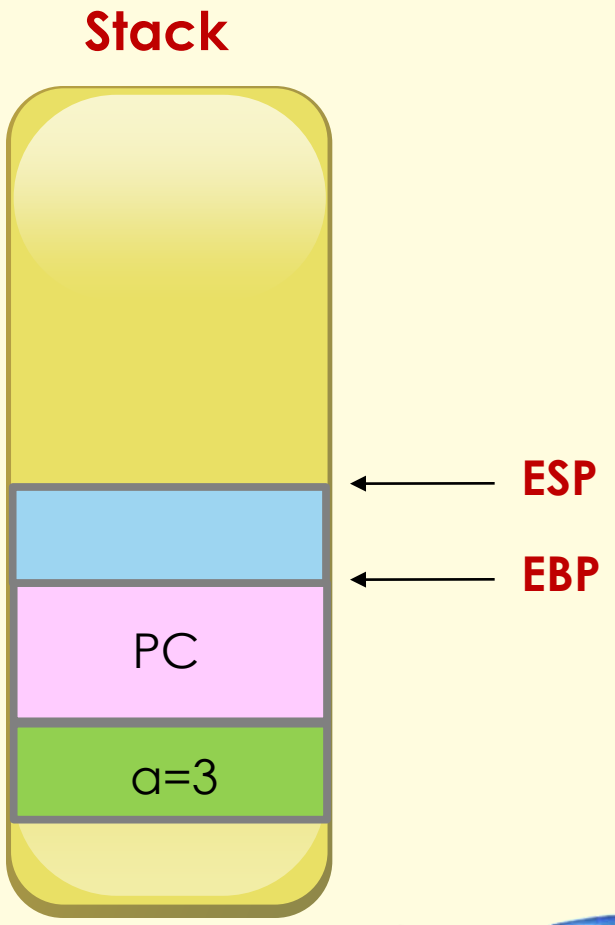
More Examples

```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}

;subprogram
  abc:
    mov ebp, esp
    sub esp, 2
```



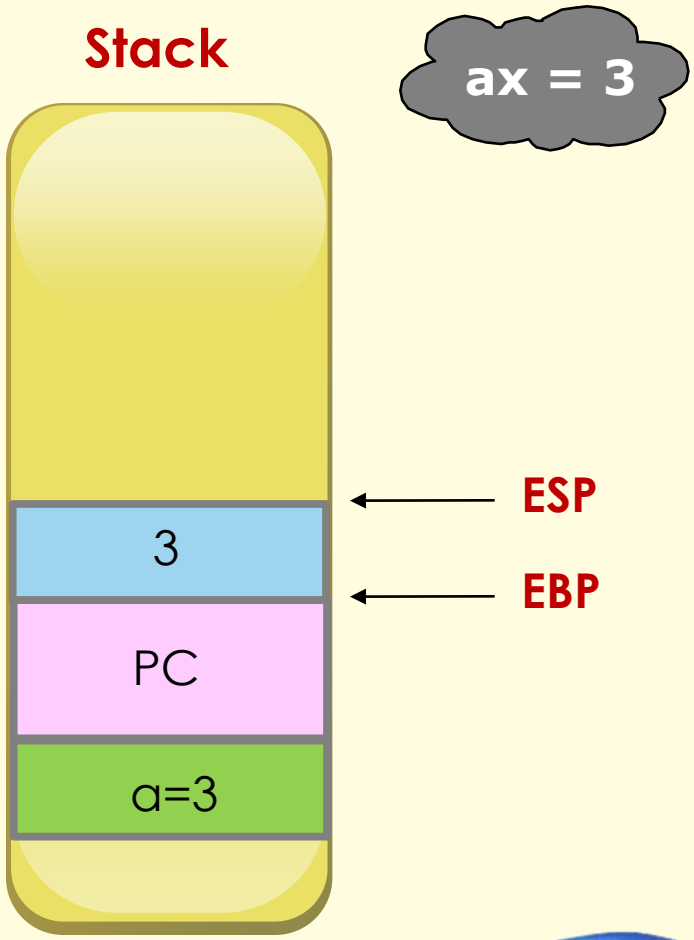
More Examples

```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}

;subprogram
  abc:
    mov ebp, esp
    sub esp, 2
```



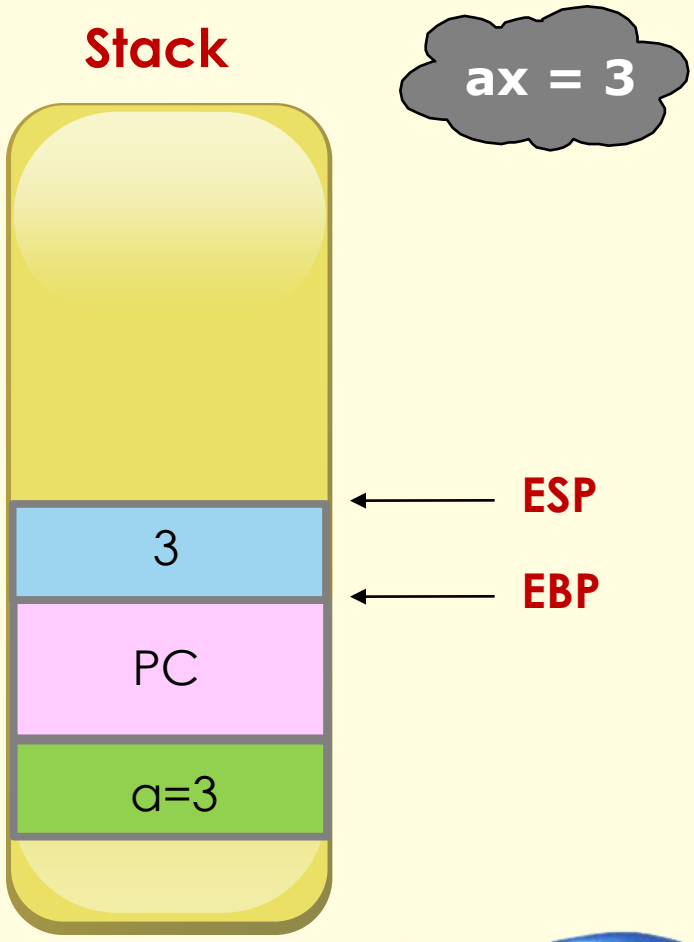
More Examples

```
int abc (int n) {
    int result=n;
    . . .
    return result;
}

;subprogram
  abc:
    mov ebp, esp
    sub esp, 2
    mov ax, [ebp+4]
    mov word[ebp-2], ax
```



More Examples

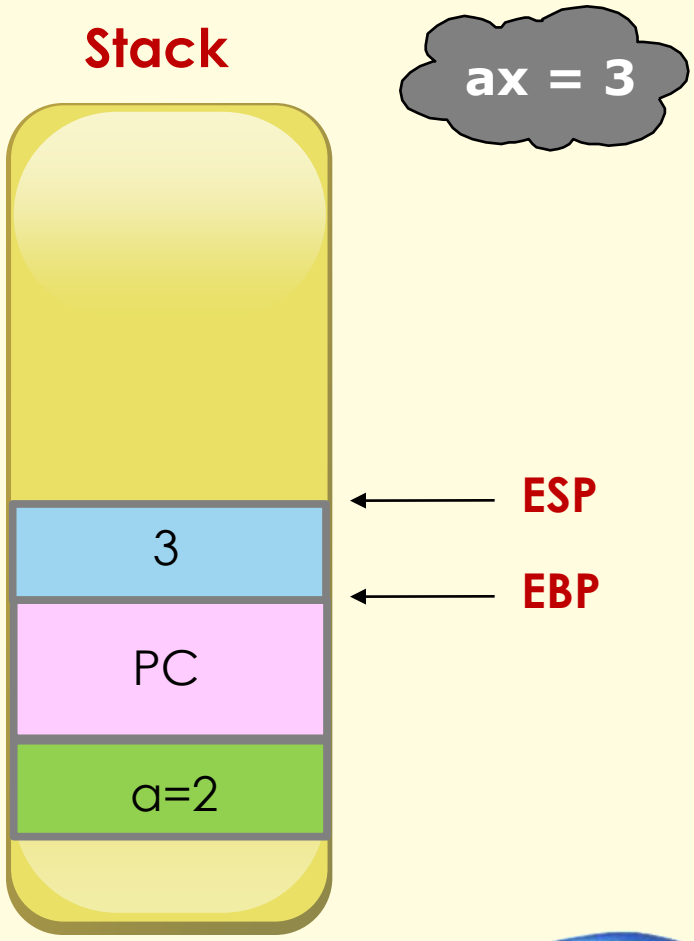
```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}
```



More Examples

```
. . .
while(n>1) {
        n--;
        result=result*n;
}
. . .
```

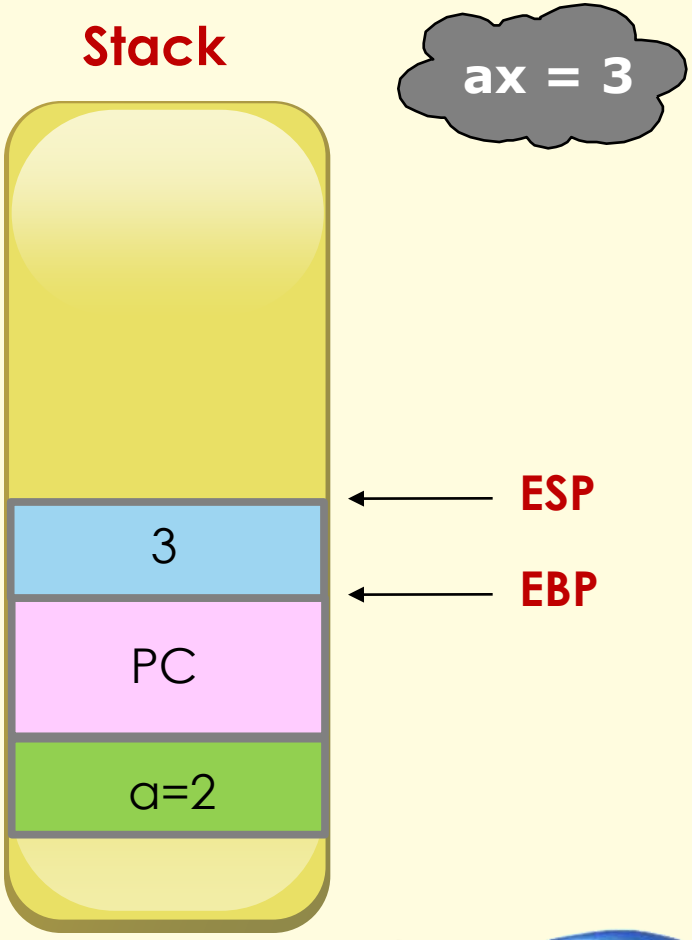
```
;subprogram
while:
        cmp word[ebp+4], 1
        jng exit
        dec word[ebp+4]
```



More Examples

```
. . .
        result=result*n;
. . .
```

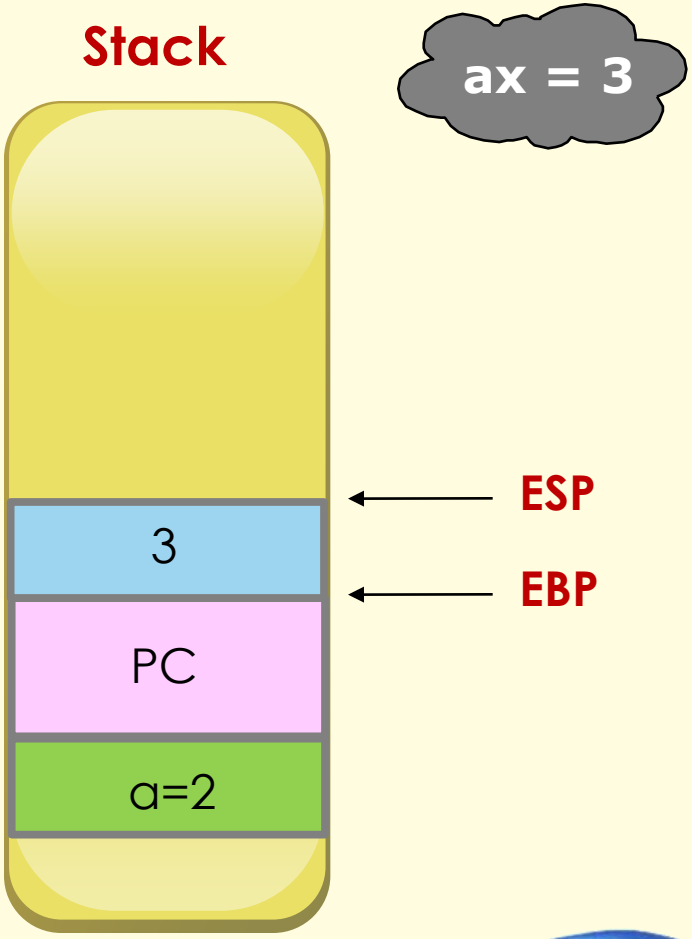
```
;subprogram
while:
      cmp word[ebp+4], 1
      jng exit
      dec word[ebp+4]
      mov ax, [ebp-2]
```



More Examples

```
. . .
        result=result*n;
. . .
```

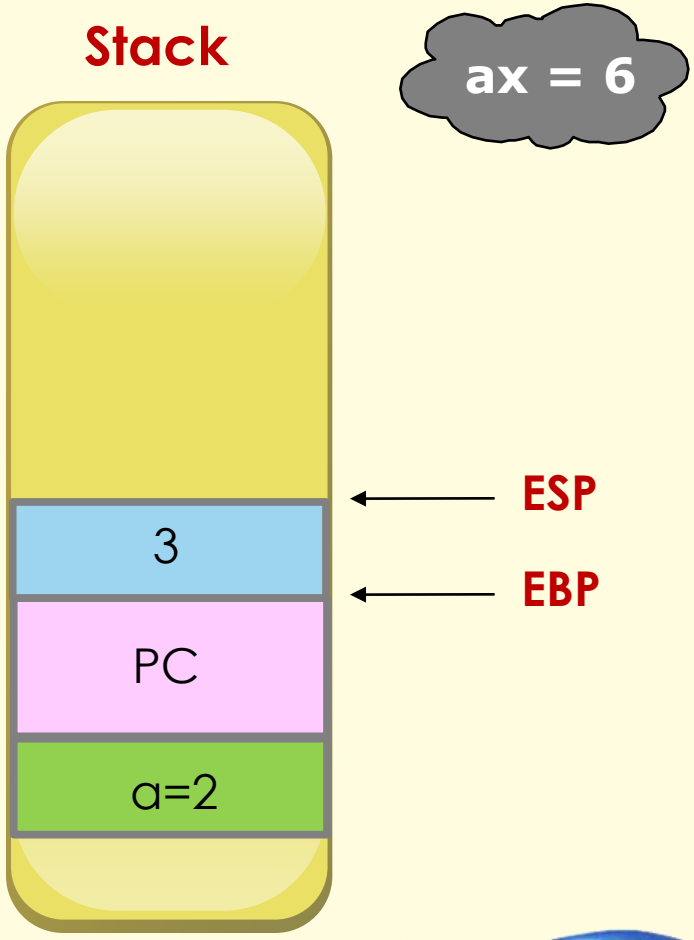
```
;subprogram
while:
      cmp word[ebp+4], 1
      jng exit
      dec word[ebp+4]
     mov ax, [ebp-2]
      mul word[ebp+4]
```



More Examples

```
. . .
        result=result*n;
. . .
```

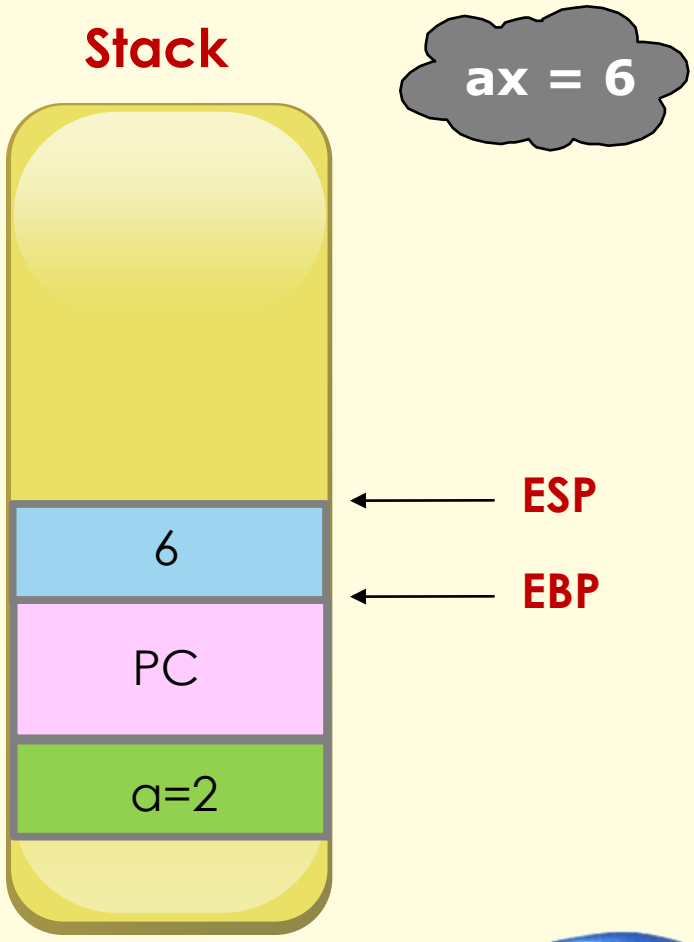
```
;subprogram
while:
      cmp word[ebp+4], 1
      jng exit
      dec word[ebp+4]
     mov ax, [ebp-2]
      mul word[ebp+4]
```



More Examples

```
. . .
        result=result*n;
. . .
```

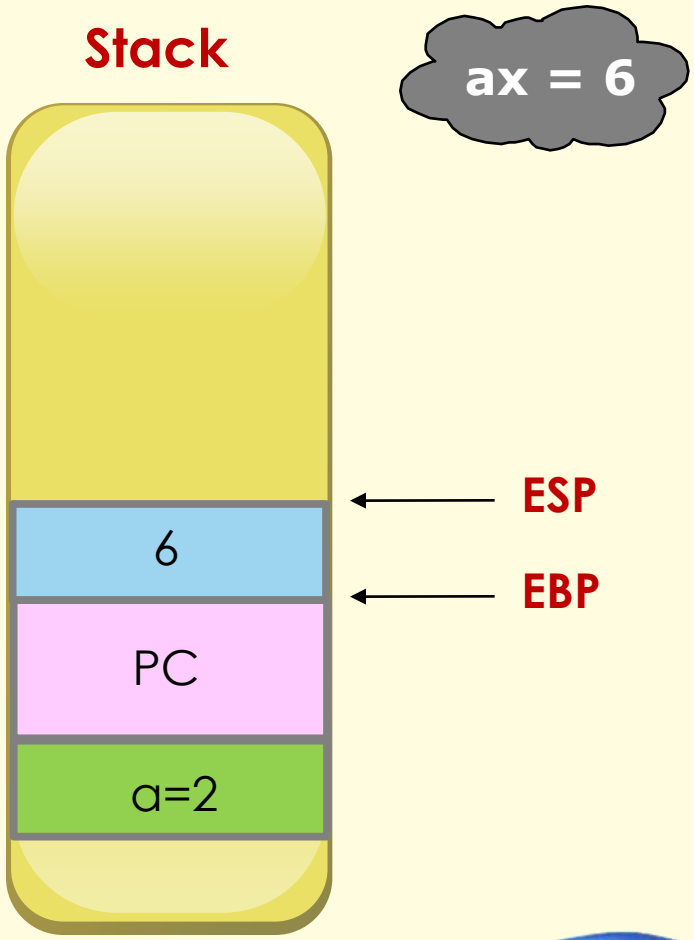
```
;subprogram
while:
        cmp word[ebp+4], 1
        jng exit
        dec word[ebp+4]
        mov ax, [ebp-2]
        mul word[ebp+4]
        mov word[ebp-2], ax
```



More Examples

```
. . .
        result=result*n;
. . .
```

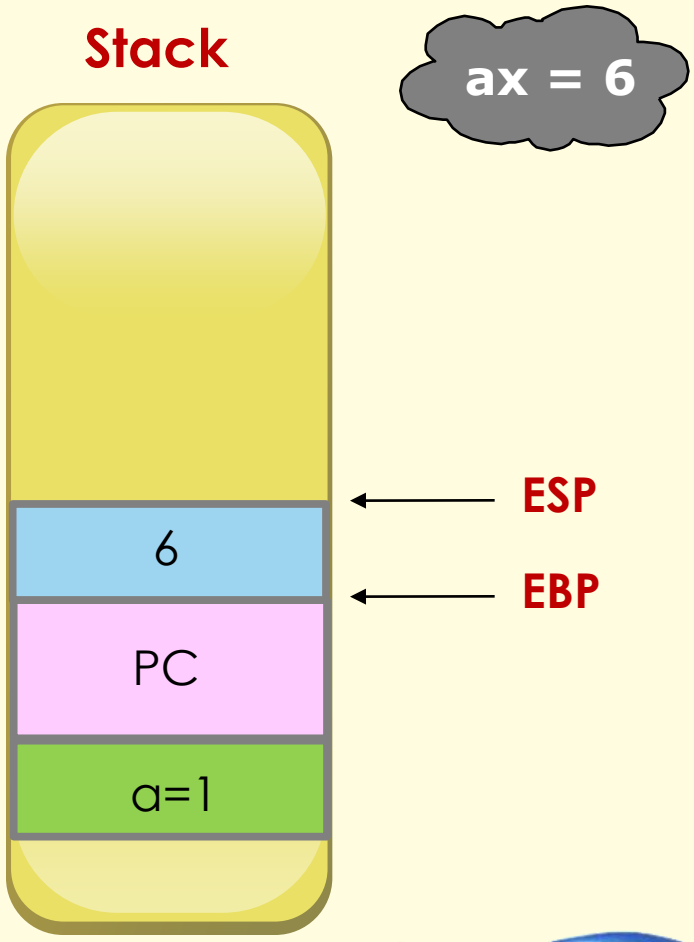
```
;subprogram
while:
        cmp word[ebp+4], 1
        jng exit
        dec word[ebp+4]
        mov ax, [ebp-2]
        mul word[ebp+4]
        mov word[ebp-2], ax
        jmp while
```



More Examples

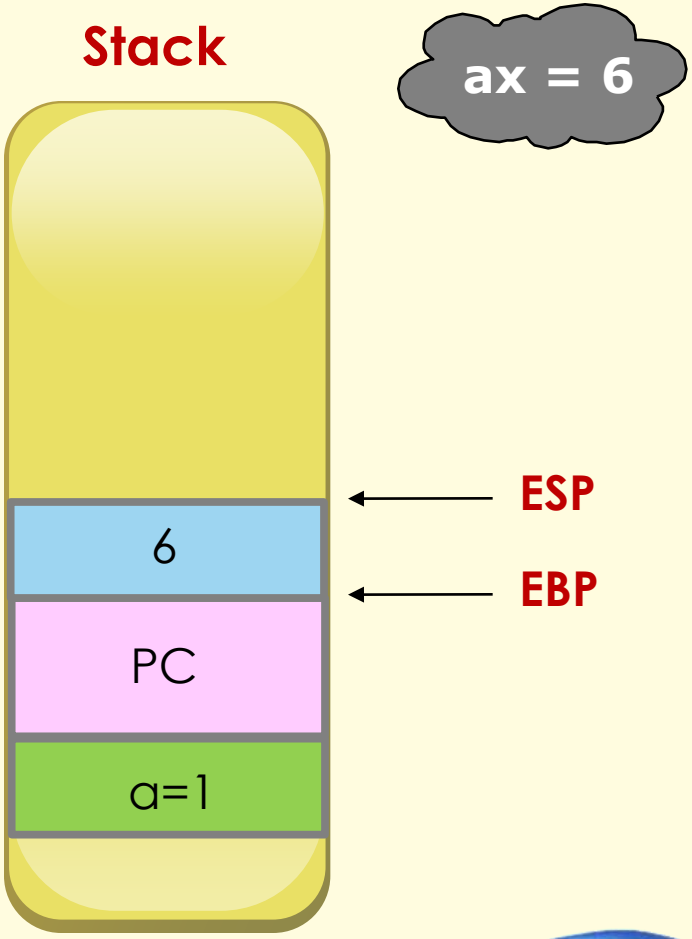
```
. . .
        result=result*n;
. . .
```

```
;subprogram
while:
        cmp word[ebp+4], 1
        jng exit
        dec word[ebp+4]
        mov ax, [ebp-2]
        mul word[ebp+4]
        mov word[ebp-2], ax
        jmp while
```



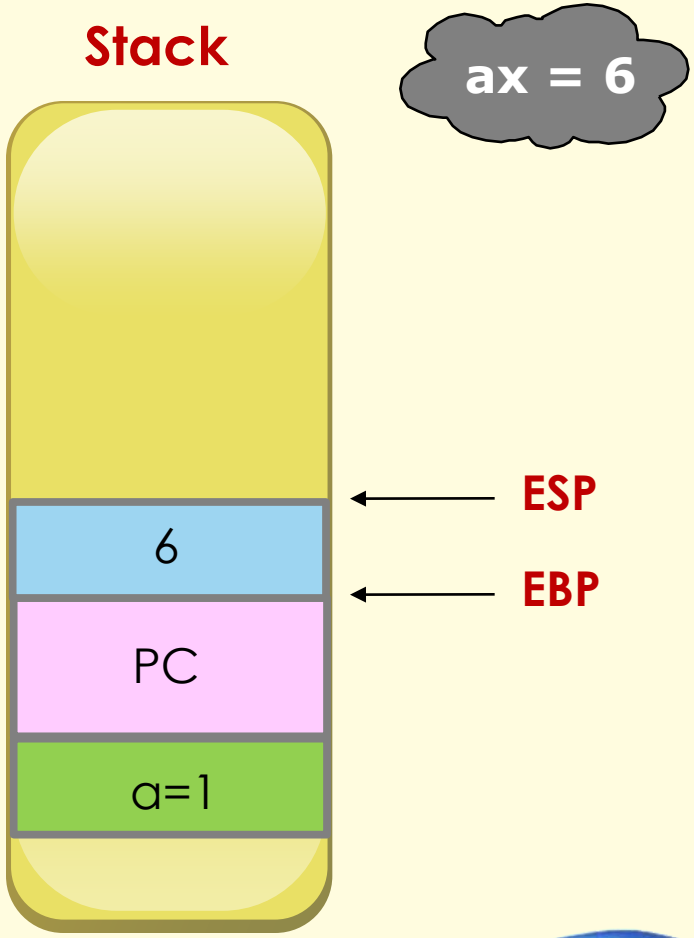
More Examples

```
int abc (int n) {
    int result=n;
    while(n>1) {
        n--;
        result=result*n;
    }
    return result;
}
```



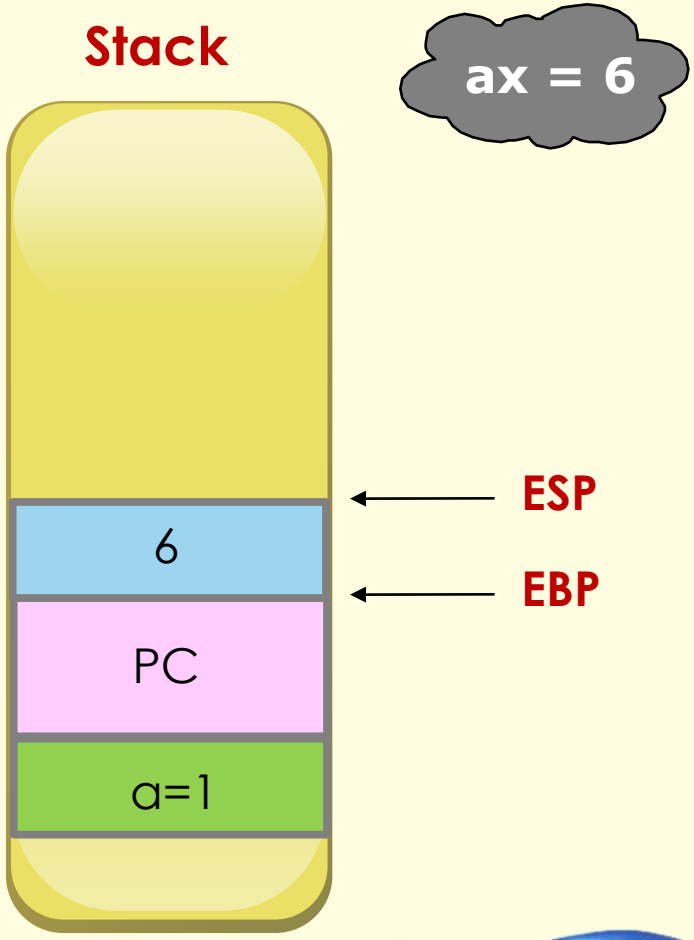
More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
     cmp word[ebp+4], 1
     jng exit
     dec word[ebp+4]
     mov ax, [ebp-2]
     mul word[ebp+4]
     mov word[ebp-2], ax
     jmp while
```



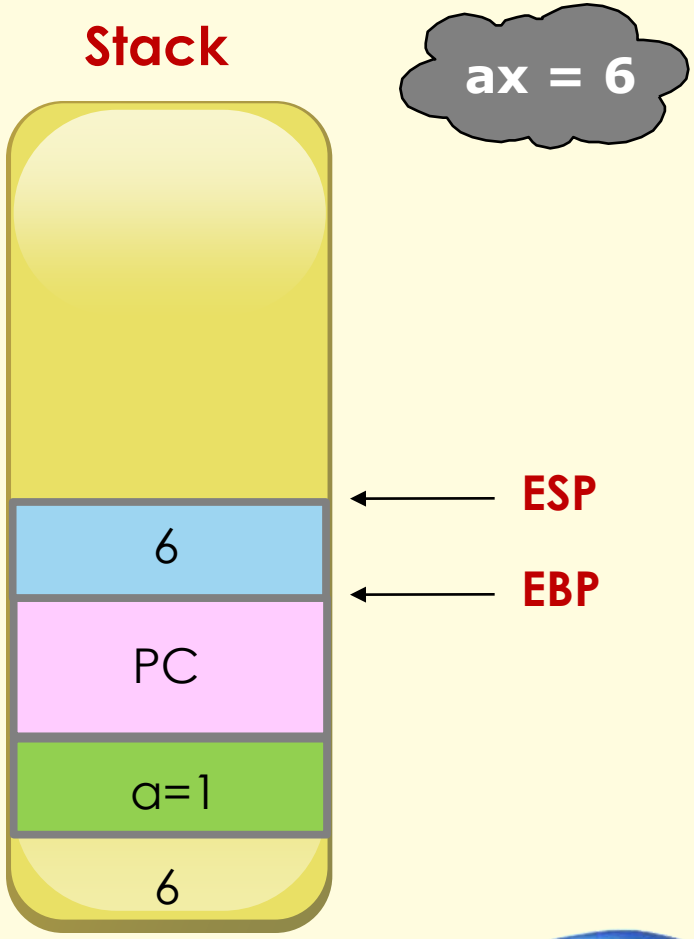
More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
     . . .
     jmp while
    exit:
```



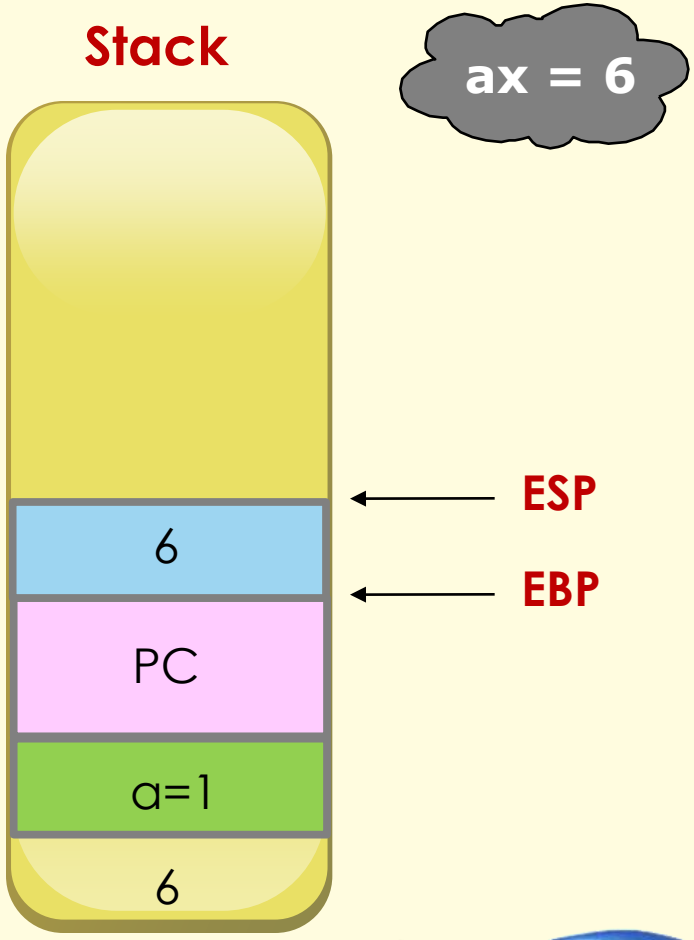
More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
      . . .
      jmp while
    exit:
```



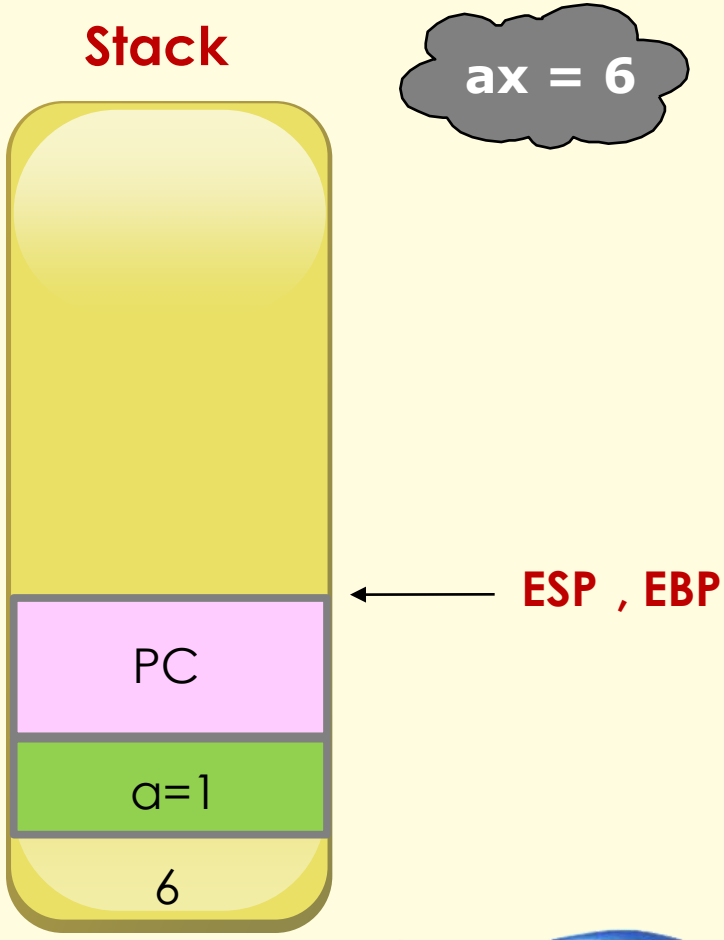
More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
     . . .
     jmp while
    exit:
     mov ax, [ebp-2]
     mov word[ebp+6], ax
```



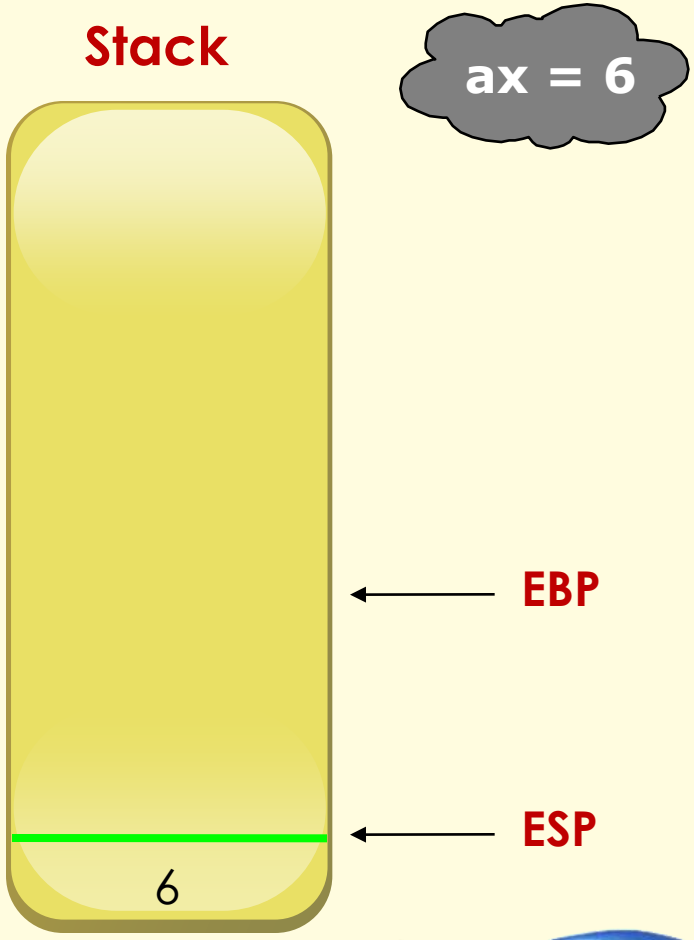
More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
     . . .
     jmp while
    exit:
     mov ax, [ebp-2]
     mov word[ebp+6], ax
     add esp, 2
```



More Examples

```
;subprogram
abc:
    ...
    mov word[ebp-2], ax
    while:
     . . .
     jmp while
    exit:
     mov ax, [ebp-2]
     mov word[ebp+6], ax
     add esp, 2
     ret 2
```



More Examples

;subprogram call

sub sp, 2
push word [a]
call abc
pop word [r]