

# Curso de Assembly x86 64 – Bits

Prof. Ronaldo Luiz Alonso  
UFMT

# Instrução call

- **call** <endereço-da-função>
  - Coloca na pilha o endereço imediatamente após o endereço da função.
  - Desvia para o endereço da função (faz o ponteiro de instrução receber o valor do endereço da função).
- **ret**
  - Retira da pilha o endereço que está no topo.
  - Desvia para este endereço (faz o ponteiro de instrução receber esse endereço).

# Pilha

- A pilha possui dois ponteiros
  - SP: Aponta para o topo da pilha
  - BP: aponta para a base da pilha
- A instrução **push** coloca um valor na pilha (incrementa ou decrementa sp e coloca um valor na posição apontada por sp)
- A instrução **pop** retira um valor da pilha (incrementa ou decrementa sp e coloca um valor na posição apontada por sp)
- A instrução **call** coloca um endereço na pilha (endereço para onde a função chamada deve retornar).
- A instrução **ret** retira um endereço da pilha (endereço para onde a função chamada deve retornar).

# Convenção de chamada de funções

- Chamador empilhar os parâmetros
- Chamador chama função
- Função chamada cria as variáveis locais
- Função chamada destrói as variáveis locais
- Função chamada coloca o endereço de retorno na pilha
- Chamador retira o valor retornado da pilha colocado pela função chamada.

# Assembly gerado pelo gcc(AT&T) e correspondente no NASM (gnu)

## AT&T:

- Constantes começam com \$
- Registradores começam com %
- <instrução> fonte, destino
- Labels começam com .

## NASM:

- Constantes são definidas com equ (equate).
- Registradores não precisam de %
- <instrução> destino, fonte
- Labels são identificados com :

# Convenção de chamada em C - 64 bits

- Primeiro parâmetro em rdi  
Segundo parâmetro em rsi  
Terceiro parâmetro em rdx  
Quarto parâmetro em rcx  
Quinto parâmetro em r8  
Sexto parâmetro em r9
- Valor de retorno em rax.

## E se tiver mais de 6 parâmetros

- Os demais parâmetros são empilhados, da mesma maneira que em 32 bits.

```

int main(void) {
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

main:

.LFB0:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

subq \$16, %rsp

movl \$2, -12(%rbp)

movl \$3, -8(%rbp)

movl -8(%rbp), %edx

movl -12(%rbp), %eax

movl %edx, %esi

movl %eax, %edi

call func1

movl %eax, -4(%rbp)

movl -4(%rbp), %eax

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

main:

push rbp

mov rbp, rsp

sub rsp, 16

mov [rbp-12], 2

mov [rbp-8], 3

mov edx, [rbp-8]

mov eax, [rbp-12]

mov esi, edx

mov edi, eax

call func1

mov [rbp-4], eax

mov eax, [rbp-4]

leave

ret



main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
    pushq    %rbp  
  
    movq     %rsp, %rbp  
  
    subq     $16, %rsp  
    movl     $2, -12(%rbp)  
    movl     $3, -8(%rbp)  
    movl     -8(%rbp), %edx  
    movl     -12(%rbp), %eax  
    movl     %edx, %esi  
    movl     %eax, %edi  
    call     func1  
    movl     %eax, -4(%rbp)  
    movl     -4(%rbp), %eax  
    leave  
  
    ret
```

main:

```
    push    rbp  
    mov     rbp, rsp  
    sub     rsp, 16  
    mov     [rbp-12], 2  
    mov     [rbp-8], 3  
    mov     edx, [rbp-8]  
    mov     eax, [rbp-12]  
    mov     esi, edx  
    mov     edi, eax  
    call    func1  
    mov     [rbp-4], eax  
    mov     eax, [rbp-4]  
    leave  
    ret
```



main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

main:

```
int main(void) {  
    int a, b, c;  
    a = 2;  
    b = 3;  
    c = func1(a,b);  
    return c;  
}
```

```
pushq    %rbp  
  
movq     %rsp, %rbp  
  
subq     $16, %rsp  
movl     $2, -12(%rbp)  
movl     $3, -8(%rbp)  
movl     -8(%rbp), %edx  
movl     -12(%rbp), %eax  
movl     %edx, %esi  
movl     %eax, %edi  
call     func1  
movl     %eax, -4(%rbp)  
movl     -4(%rbp), %eax  
leave  
  
ret
```

main:

```
push     rbp  
mov      rbp, rsp  
sub      rsp, 16  
mov      [rbp-12], 2  
mov      [rbp-8], 3  
mov      edx, [rbp-8]  
mov      eax, [rbp-12]  
mov      esi, edx  
mov      edi, eax  
call     func1  
mov      [rbp-4], eax  
mov      eax, [rbp-4]  
leave  
ret
```

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax,[rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```

rsp →

Endereço para  
onde main deve  
retornar

rbp →

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

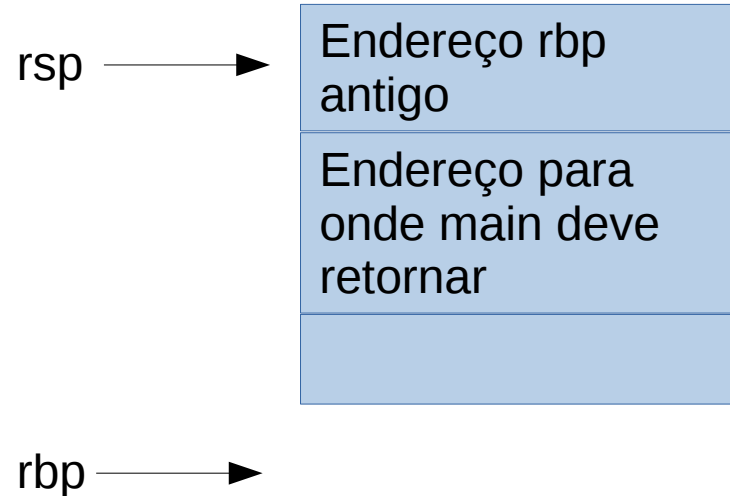
```

main:

```

push    rbp
mov     rbp, rsp
sub     rsp, 16
mov     [rbp-12], 2
mov     [rbp-8], 3
mov     edx, [rbp-8]
mov     eax, [rbp-12]
mov     esi, edx
mov     edi, eax
call    func1
mov     [rbp-4], eax
mov     eax, [rbp-4]
leave
ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

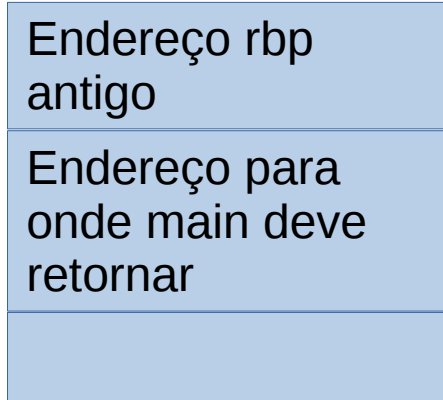
```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```

rsp →  
 rbp →



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

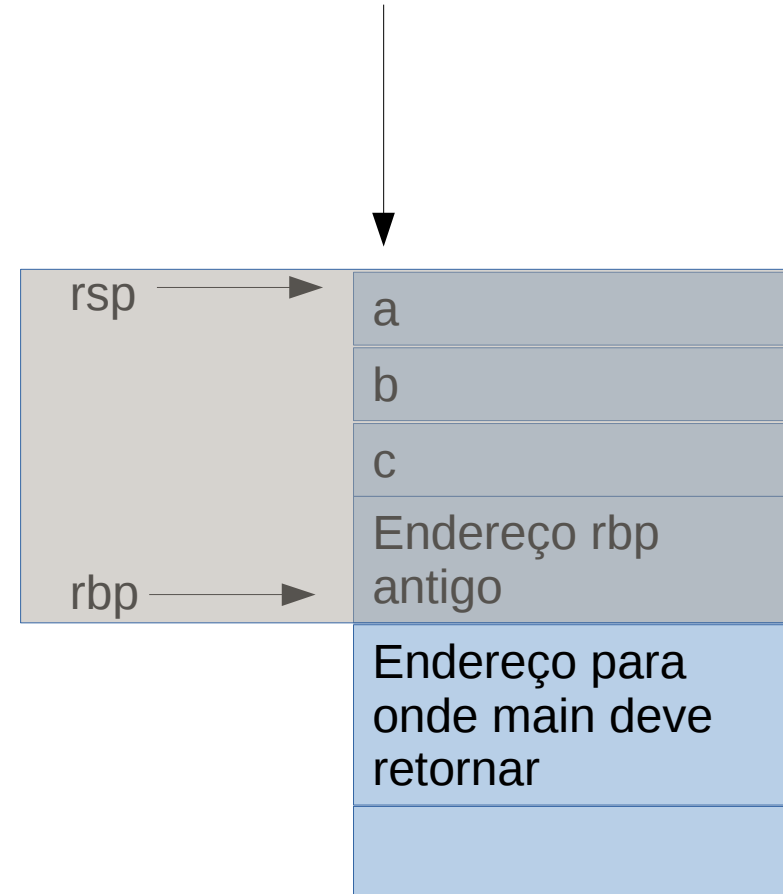
```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```

stack frame para main  
(registro de ativação  
para função main)



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

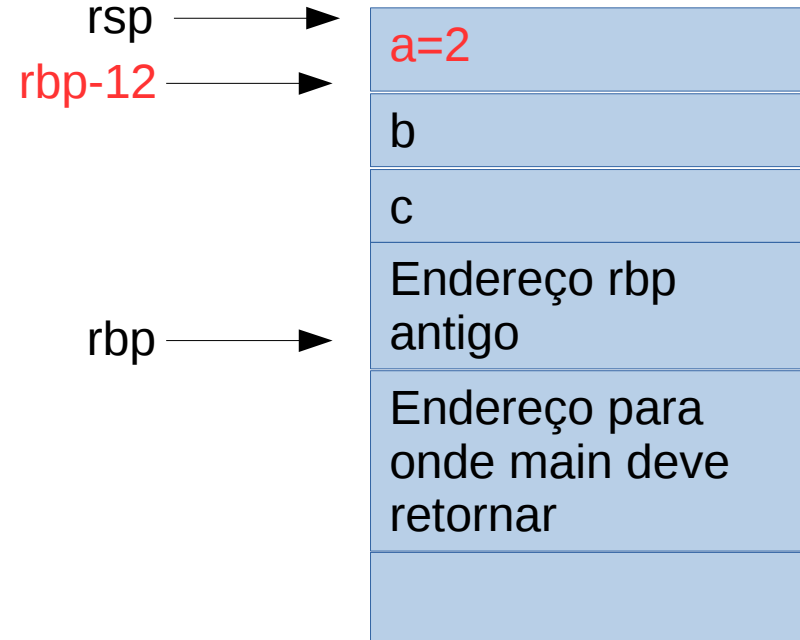
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

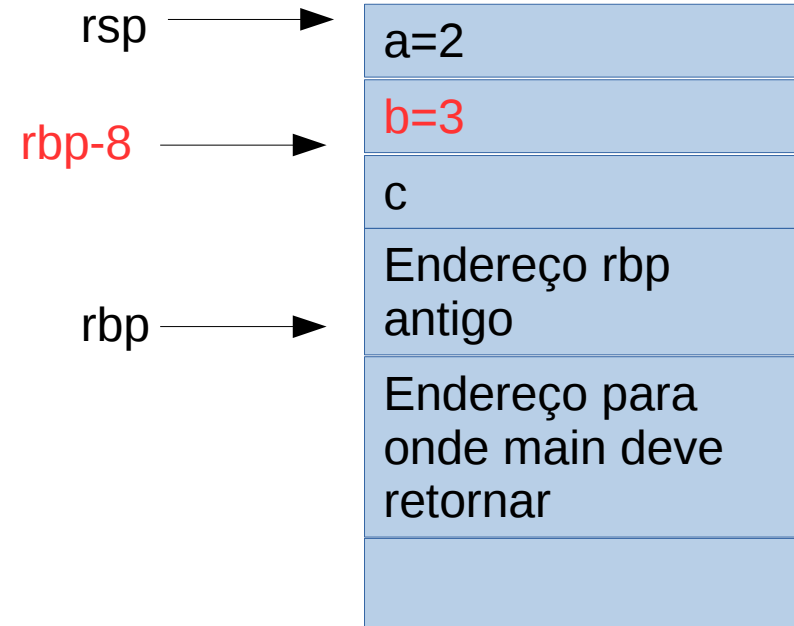
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```





```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

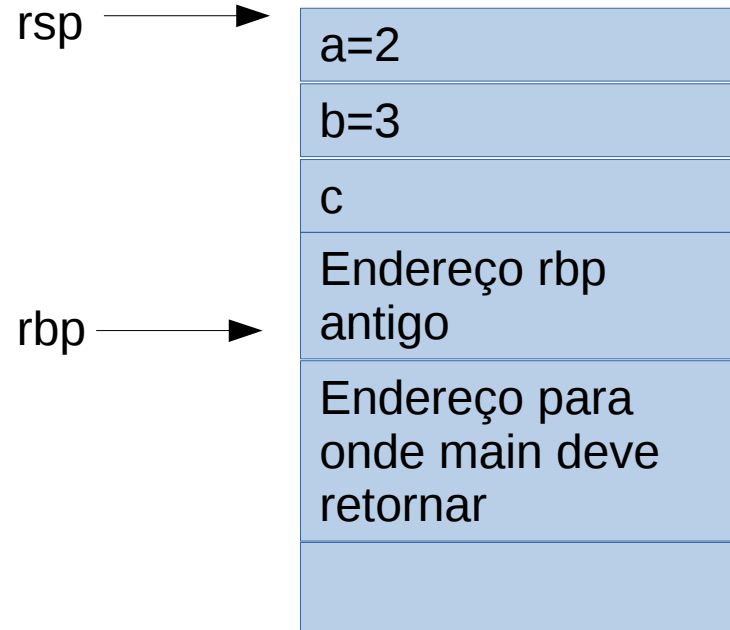
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

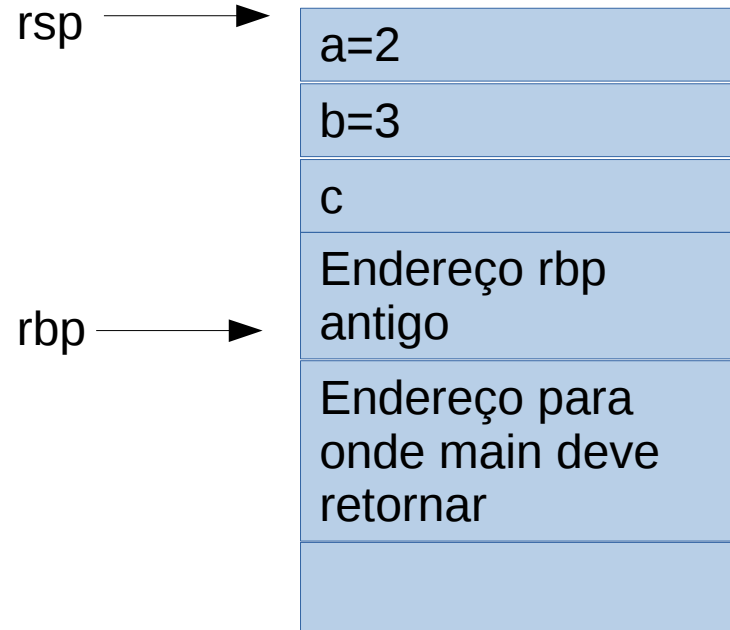
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```



```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret
```

rsp →

rbp →

Endereço para  
onde onde func1  
deve retornar

a=2

b=3

c

Endereço rbp  
antigo

Endereço para  
onde main deve  
retornar

Note que o compilador gcc colocou as instruções para passar os parâmetros fora de ordem. Ordem natural:

```
mov eax,[rbp-12]
mov edi, eax
mov edx,[rbp-8]
mov esi, edx
call func1
```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

func1:
.LFB1:
.cfi_startproc
    pushq    %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
    movq     %rsp, %rbp
.cfi_def_cfa_register 6
    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave
.cfi_def_cfa 7, 8
    ret
.cfi_endproc

```

```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq $32, %rsp
    movl %edi, -20(%rbp)
    movl %esi, -24(%rbp)
    movl -20(%rbp), %edx
    movl -24(%rbp), %eax
    addl %edx, %eax
    movl %eax, -8(%rbp)
    movl -8(%rbp), %eax
    movl %eax, %edi
    call func2
    movl %eax, -4(%rbp)
    movl -8(%rbp), %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

pushq %rbp

movq %rsp, %rbp

```
subq $32, %rsp
movl %edi, -20(%rbp)
movl %esi, -24(%rbp)
movl -20(%rbp), %edx
movl -24(%rbp), %eax
addl %edx, %eax
movl %eax, -8(%rbp)
movl -8(%rbp), %eax
movl %eax, %edi
call func2
movl %eax, -4(%rbp)
movl -8(%rbp), %edx
movl -4(%rbp), %eax
addl %edx, %eax
leave
```

ret

func1:

push rbp  
mov rbp, rsp,

```
sub     rsp,32
movl    [rbp-20],edi
movl    [rbp-24],esi
movl    edx, [rbp-20]
movl    eax, [rbp-24]
addl    eax, edx
movl    [rbp-8], eax
movl    eax,[rbp-8]
movl    edi, eax
call    func2
movl    [rbp-4], eax
movl    edx, [rbp-8]
movl    eax, [rbp-4]
addl    eax, edx
leave
```

ret

func1:

pushq %rbp

movq %rsp, %rbp

```
subq $32, %rsp
movl %edi, -20(%rbp)
movl %esi, -24(%rbp)
movl -20(%rbp), %edx
movl -24(%rbp), %eax
addl %edx, %eax
movl %eax, -8(%rbp)
movl -8(%rbp), %eax
movl %eax, %edi
call func2
movl %eax, -4(%rbp)
movl -8(%rbp), %edx
movl -4(%rbp), %eax
addl %edx, %eax
leave
```

ret

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

func1:

push rbp  
mov rbp, rsp,

```
sub rsp,32
movl [rbp-20],edi
movl [rbp-24],esi
movl edx, [rbp-20]
movl eax, [rbp-24]
addl eax, edx
movl [rbp-8], eax
movl eax,[rbp-8]
movl edi, eax
call func2
movl [rbp-4], eax
movl edx, [rbp-8]
movl eax, [rbp-4]
addl eax, edx
leave
```

ret

Espaço para cópia dos  
parâmetros e variáveis  
locais.

$32 = (6+2)*4$

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

func1:

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax, [rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```



Salva cópia dos  
parâmetros passados por  
valor na pilha.  
Não tem código  
correspondente.

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

func1:

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq $32, %rsp
    movl %edi, -20(%rbp)
    movl %esi, -24(%rbp)
    movl -20(%rbp), %edx
    movl -24(%rbp), %eax
    addl %edx, %eax
    movl %eax, -8(%rbp)
    movl -8(%rbp), %eax
    movl %eax, %edi
    call func2
    movl %eax, -4(%rbp)
    movl -8(%rbp), %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave

    ret
```

func1:

```
    push     rbp
    mov      rbp, rsp,

    sub      rsp,32
    movl     [rbp-20],edi
    movl     [rbp-24],esi
    movl     edx, [rbp-20]
    movl     eax, [rbp-24]
    addl     eax, edx
    movl     [rbp-8], eax
    movl     eax,[rbp-8]
    movl     edi, eax
    call     func2
    movl     [rbp-4], eax
    movl     edx, [rbp-8]
    movl     eax, [rbp-4]
    addl     eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave

    ret
```

func1:

```
    push     rbp
    mov      rbp, rsp,

    sub      rsp,32
    movl     [rbp-20],edi
    movl     [rbp-24],esi
    movl     edx, [rbp-20]
    movl     eax, [rbp-24]
    addl     eax, edx
    movl     [rbp-8], eax
    movl     eax,[rbp-8]
    movl     edi, eax
    call     func2
    movl     [rbp-4], eax
    movl     edx, [rbp-8]
    movl     eax, [rbp-4]
    addl     eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq     $32, %rsp
    movl     %edi, -20(%rbp)
    movl     %esi, -24(%rbp)
    movl     -20(%rbp), %edx
    movl     -24(%rbp), %eax
    addl     %edx, %eax
    movl     %eax, -8(%rbp)
    movl     -8(%rbp), %eax
    movl     %eax, %edi
    call     func2
    movl     %eax, -4(%rbp)
    movl     -8(%rbp), %edx
    movl     -4(%rbp), %eax
    addl     %edx, %eax
    leave

    ret
```

func1:

```
    push     rbp
    mov      rbp, rsp,

    sub      rsp,32
    movl     [rbp-20],edi
    movl     [rbp-24],esi
    movl     edx, [rbp-20]
    movl     eax, [rbp-24]
    addl     eax, edx
    movl     [rbp-8], eax
    movl     eax,[rbp-8]
    movl     edi, eax
    call     func2
    movl     [rbp-4], eax
    movl     edx, [rbp-8]
    movl     eax, [rbp-4]
    addl     eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq $32, %rsp
    movl %edi, -20(%rbp)
    movl %esi, -24(%rbp)
    movl -20(%rbp), %edx
    movl -24(%rbp), %eax
    addl %edx, %eax
    movl %eax, -8(%rbp)
    movl -8(%rbp), %eax
    movl %eax, %edi
    call func2
    movl %eax, -4(%rbp)
    movl -8(%rbp), %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq $32, %rsp
    movl %edi, -20(%rbp)
    movl %esi, -24(%rbp)
    movl -20(%rbp), %edx
    movl -24(%rbp), %eax
    addl %edx, %eax
    movl %eax, -8(%rbp)
    movl -8(%rbp), %eax
    movl %eax, %edi
    call func2
    movl %eax, -4(%rbp)
    movl -8(%rbp), %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```

func1:

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
    pushq    %rbp

    movq     %rsp, %rbp

    subq $32, %rsp
    movl %edi, -20(%rbp)
    movl %esi, -24(%rbp)
    movl -20(%rbp), %edx
    movl -24(%rbp), %eax
    addl %edx, %eax
    movl %eax, -8(%rbp)
    movl -8(%rbp), %eax
    movl %eax, %edi
    call func2
    movl %eax, -4(%rbp)
    movl -8(%rbp), %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave

    ret
```

func1:

```
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret
```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

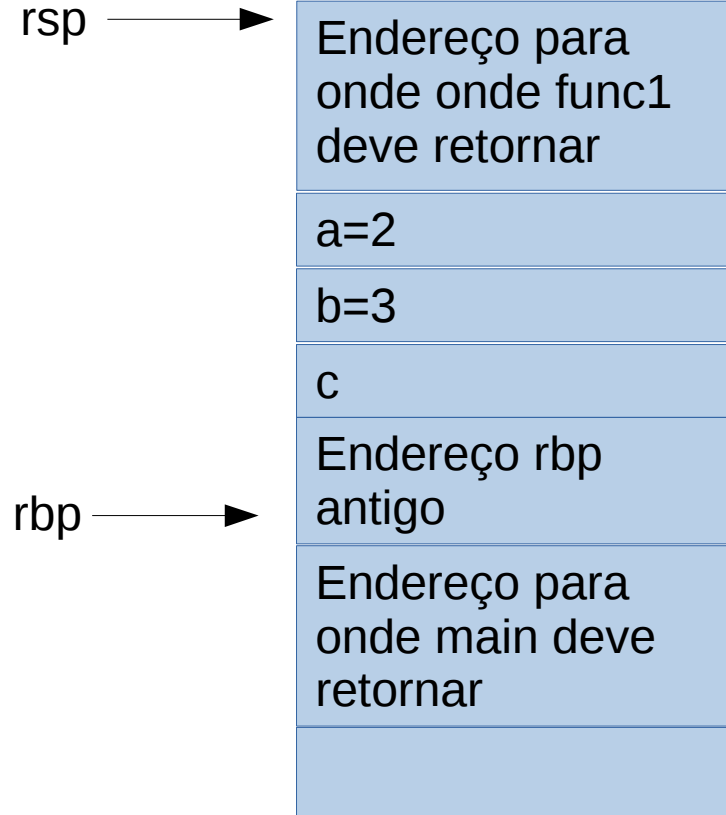
```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```

rsp →

Endereço rbp  
antigo

Endereço para  
onde onde func1  
deve retornar

a=2

b=3

c

rbp →

Endereço rbp  
antigo

Endereço para  
onde main deve  
retornar

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```

rsp →

rbp →

Endereço rbp antigo
Endereço para onde onde func1 deve retornar
a=2
b=3
c
Endereço rbp antigo
Endereço para onde main deve retornar

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

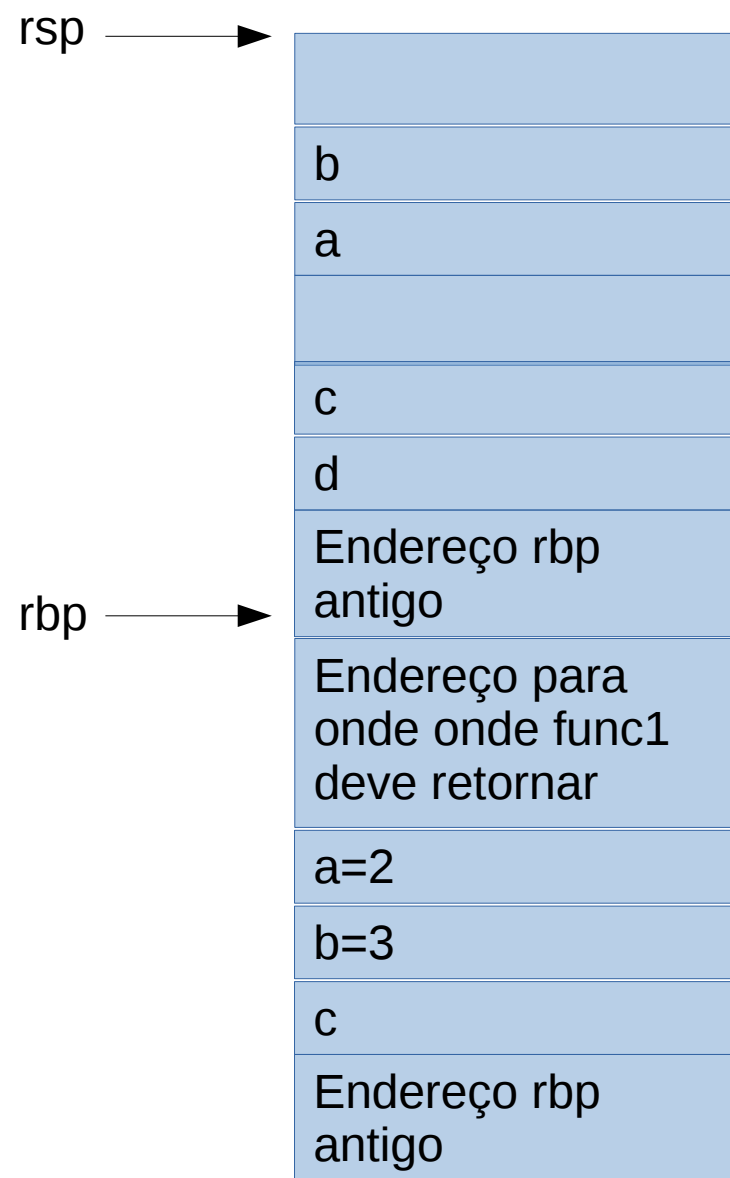
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
           sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```



```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

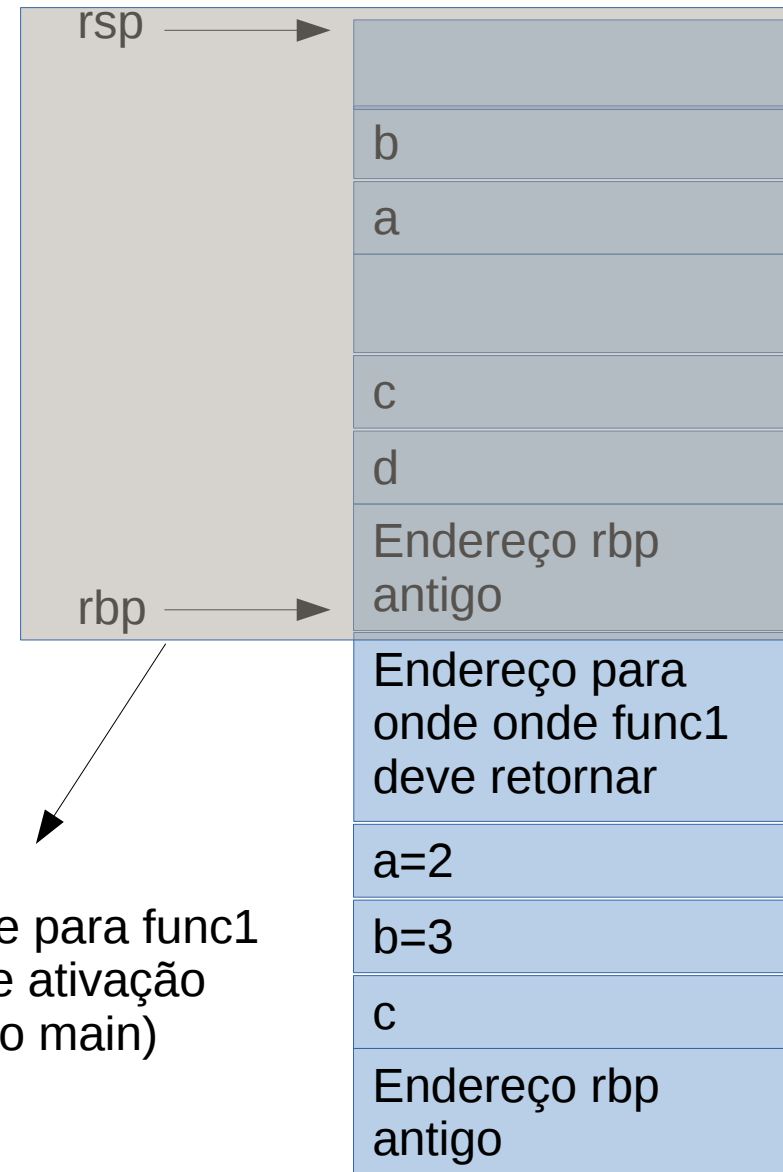
```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
func1:
    push    rbp
    mov     rbp, rsp,
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
```

```
ret
```

stack frame para func1  
(registro de ativação  
para função main)



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

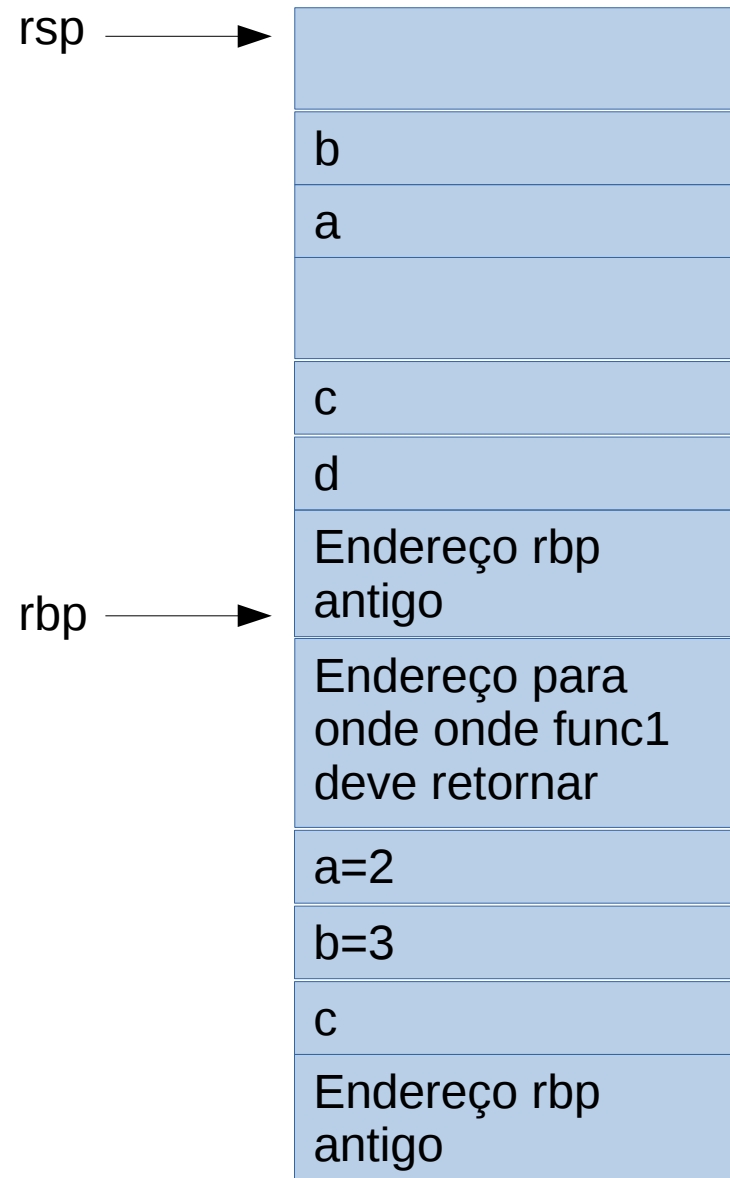
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

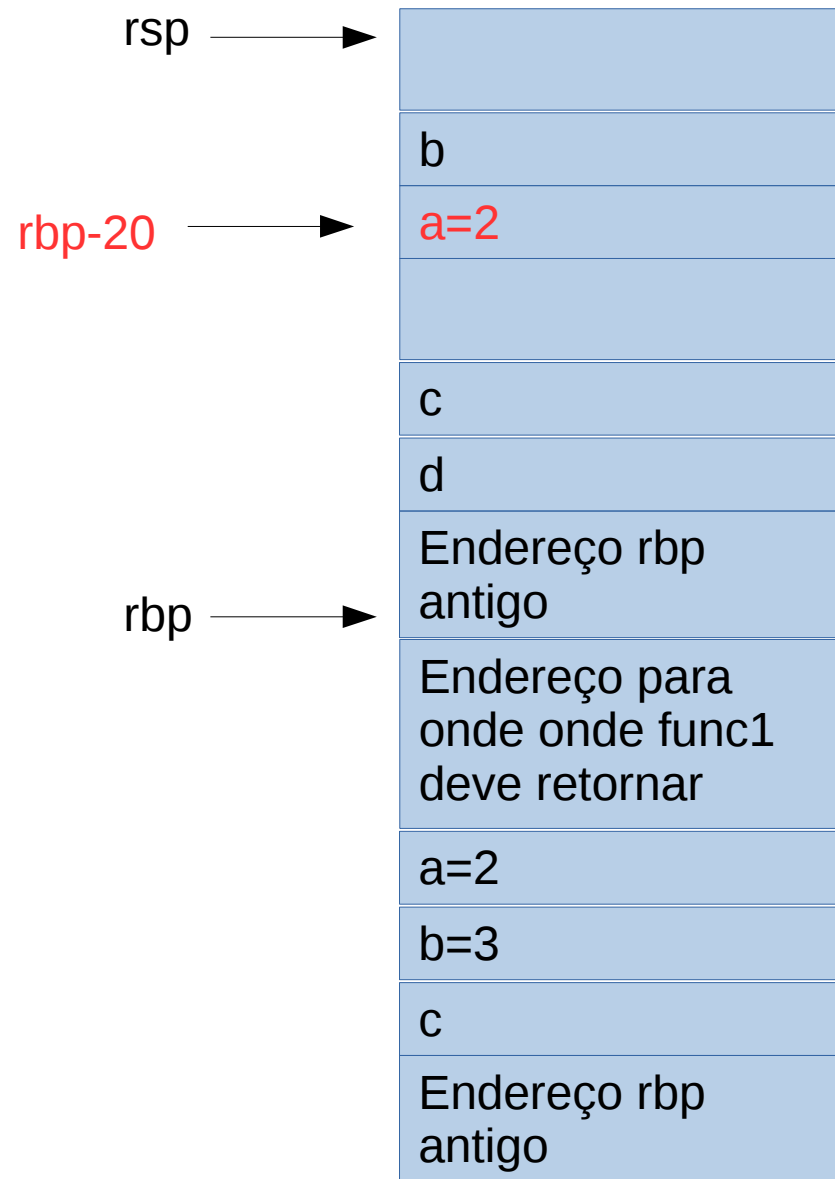
```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

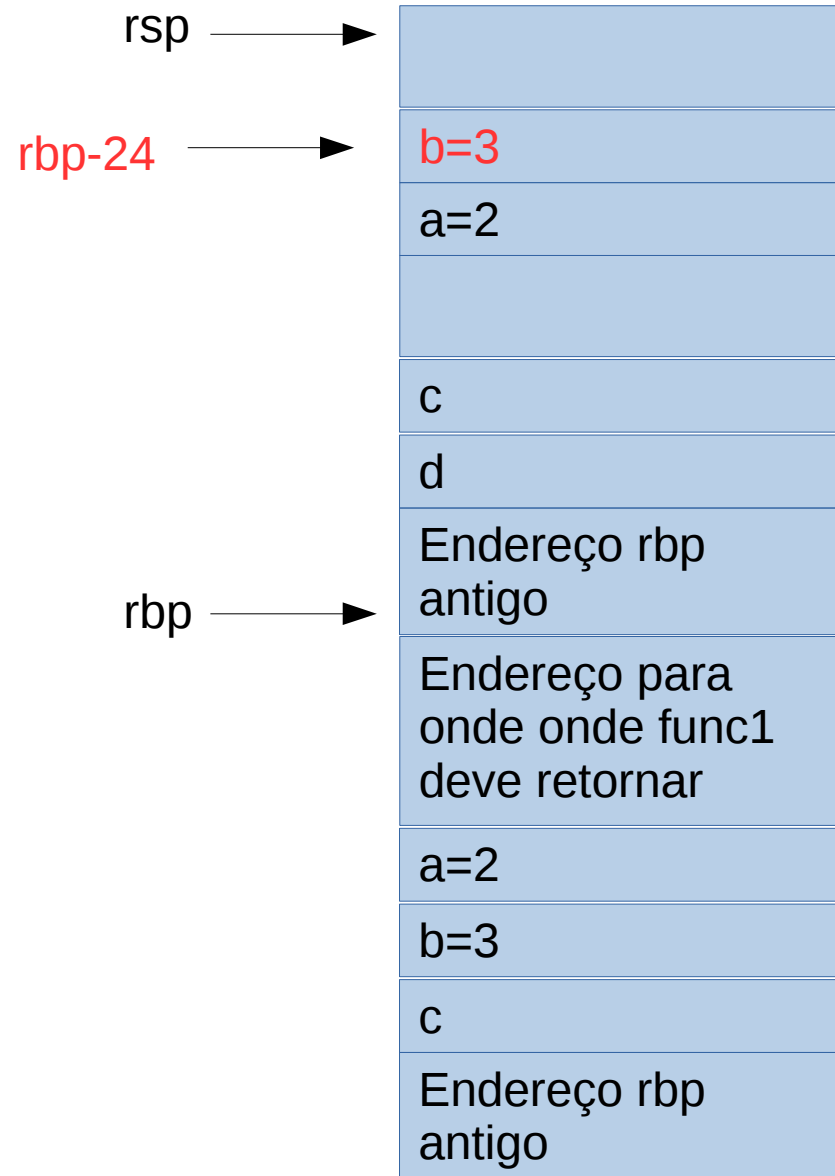
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
        sub    rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```





```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

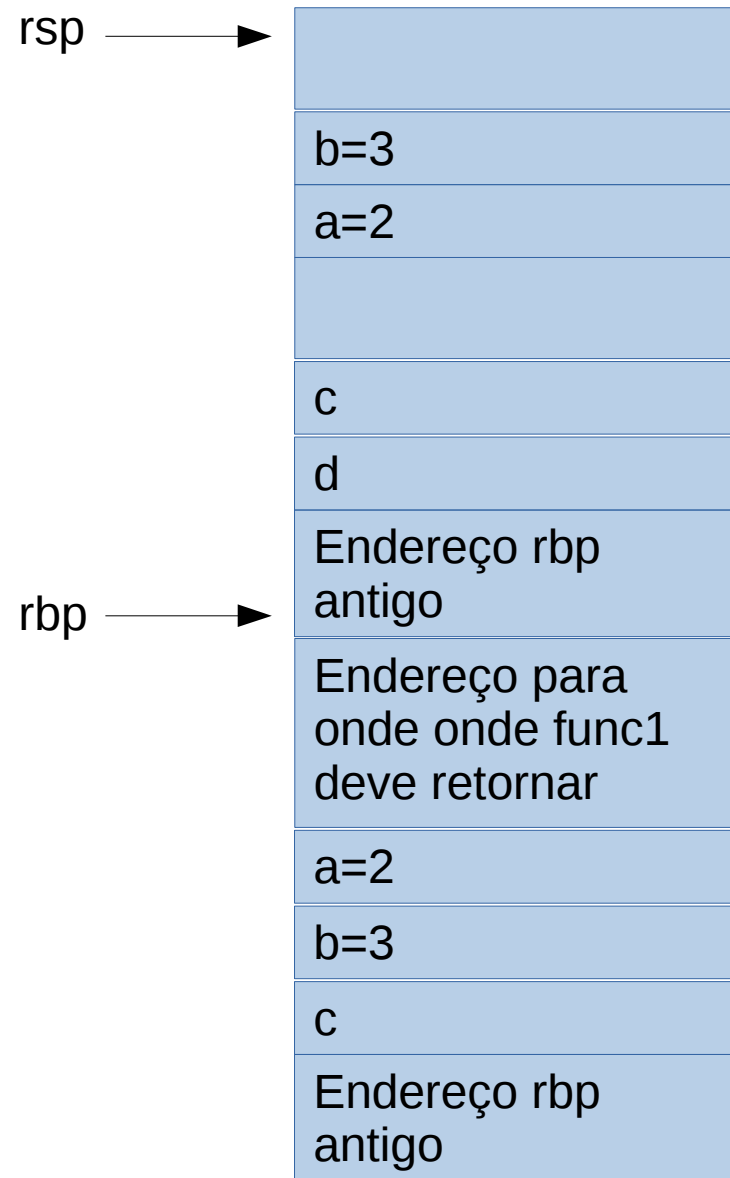
```

func1:
    push    rbp
    mov     rbp, rsp,
        4

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

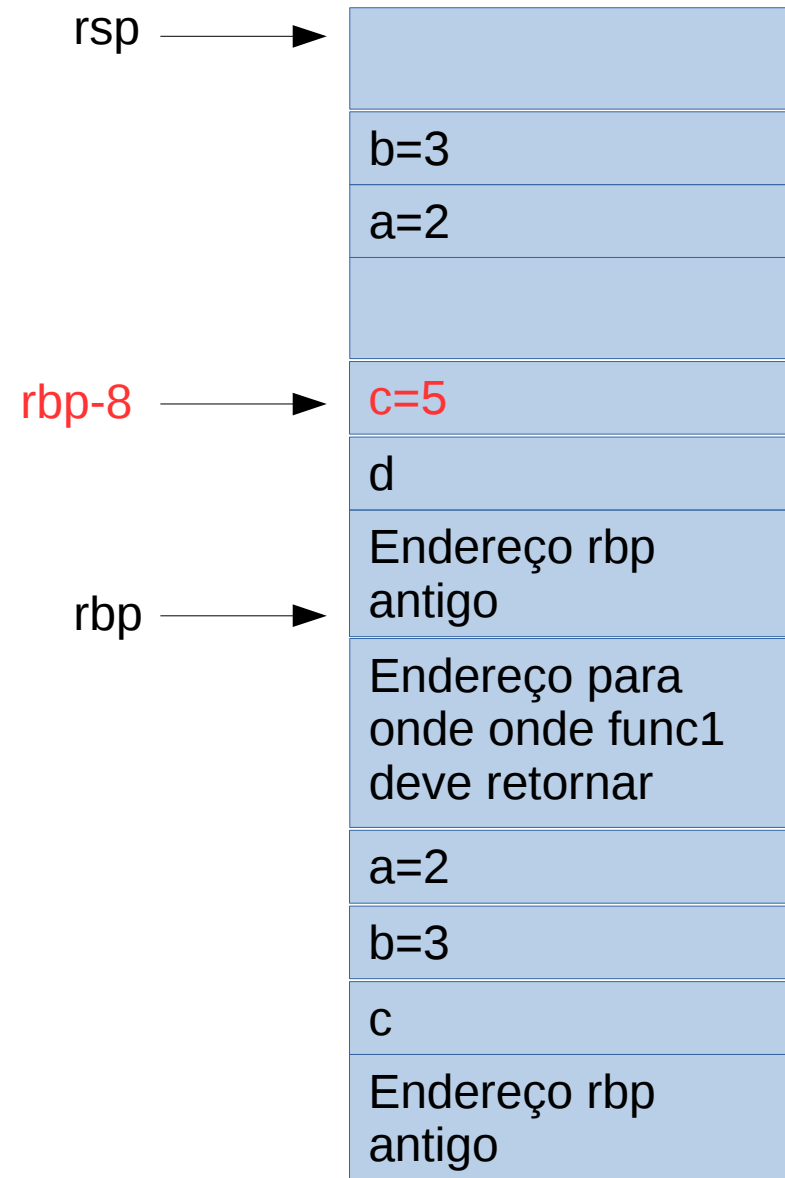
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
    
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

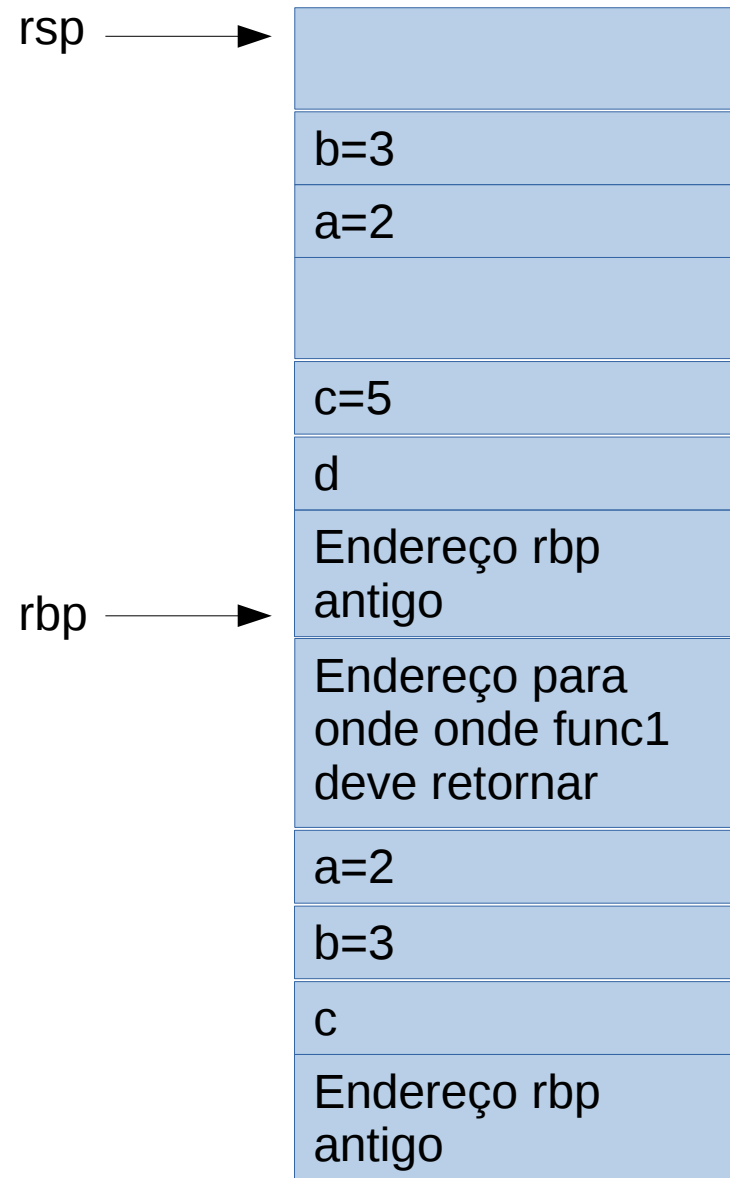
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
        sub    rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

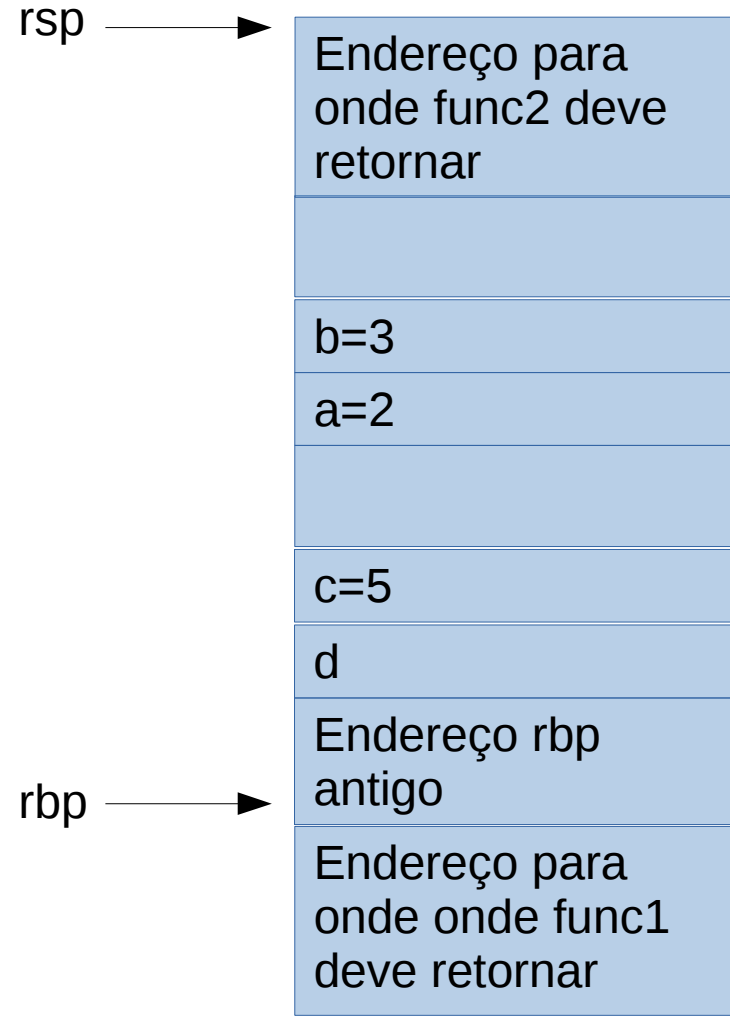
```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    eax,[rbp-8]
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

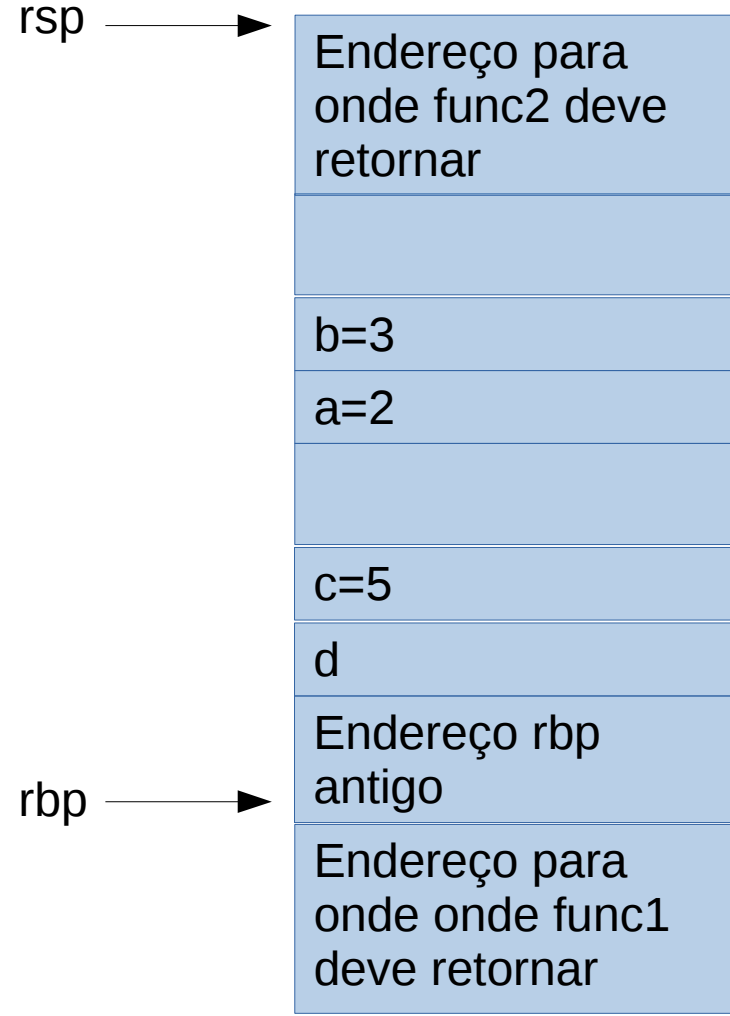
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

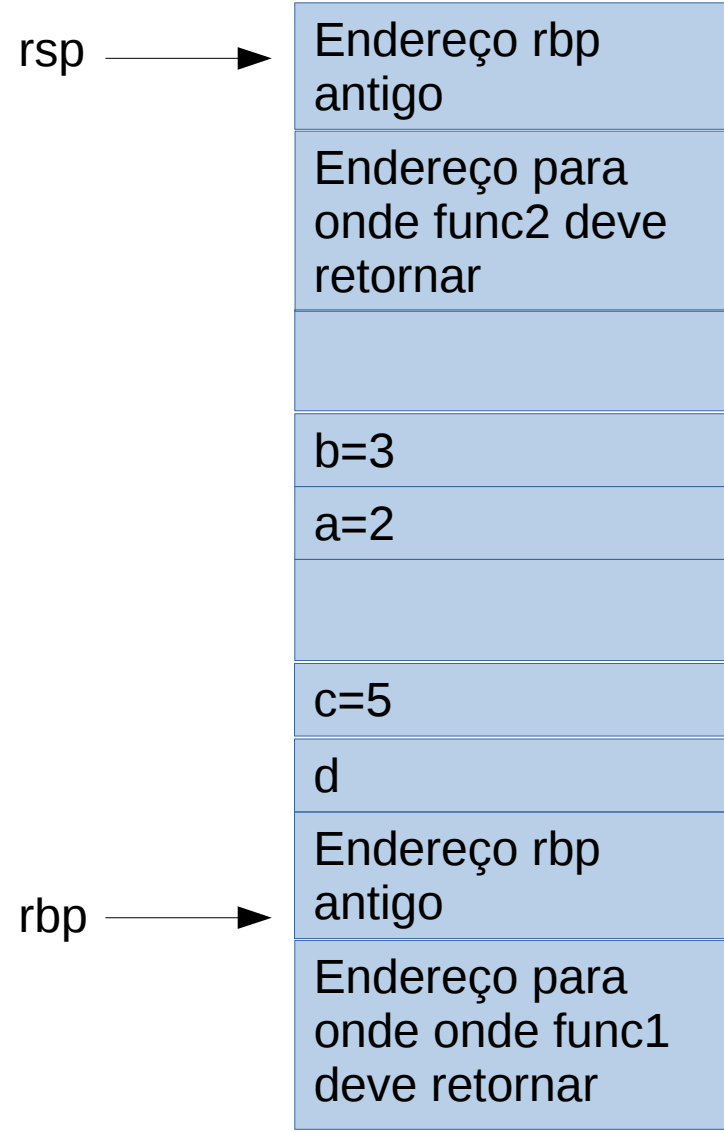
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```

rsp →  
 rbp →

Endereço rbp  
antigo

Endereço para  
onde func2 deve  
retornar

b=3

a=2

c=5

d

Endereço rbp  
antigo

Endereço para  
onde onde func1  
deve retornar

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

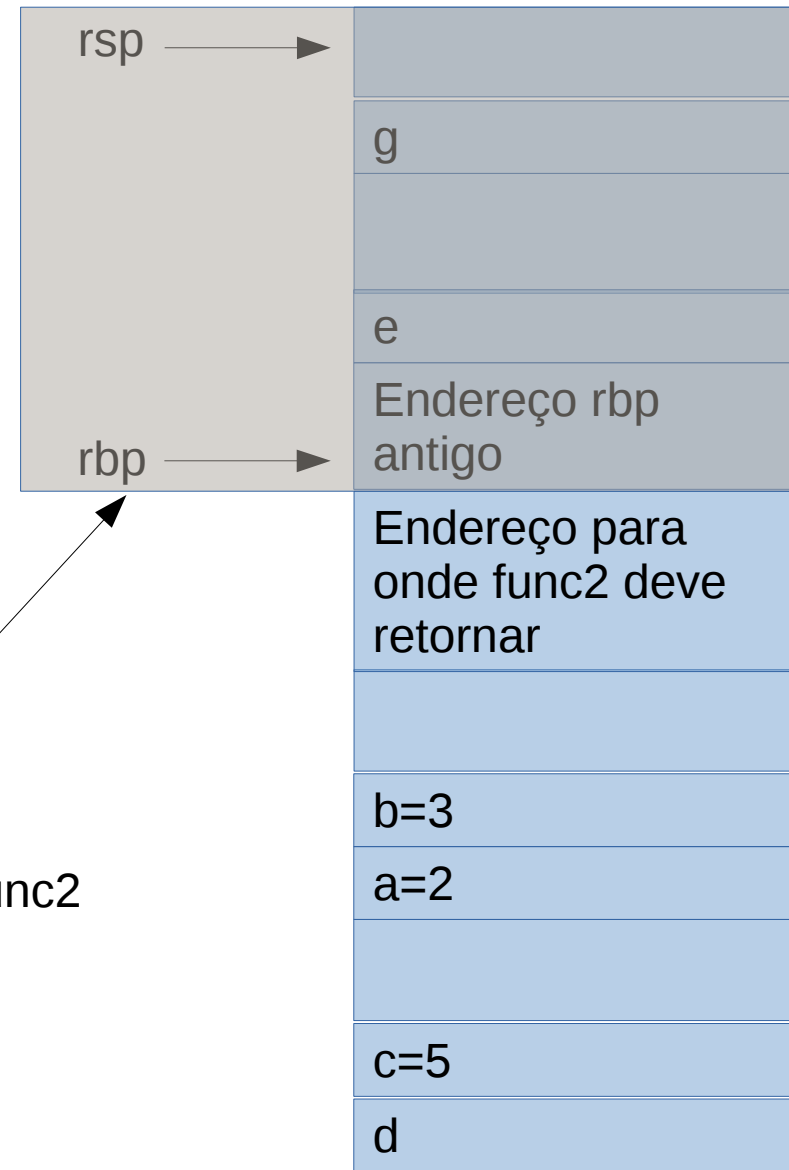
```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax, 2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```

Stack frame para func2





```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

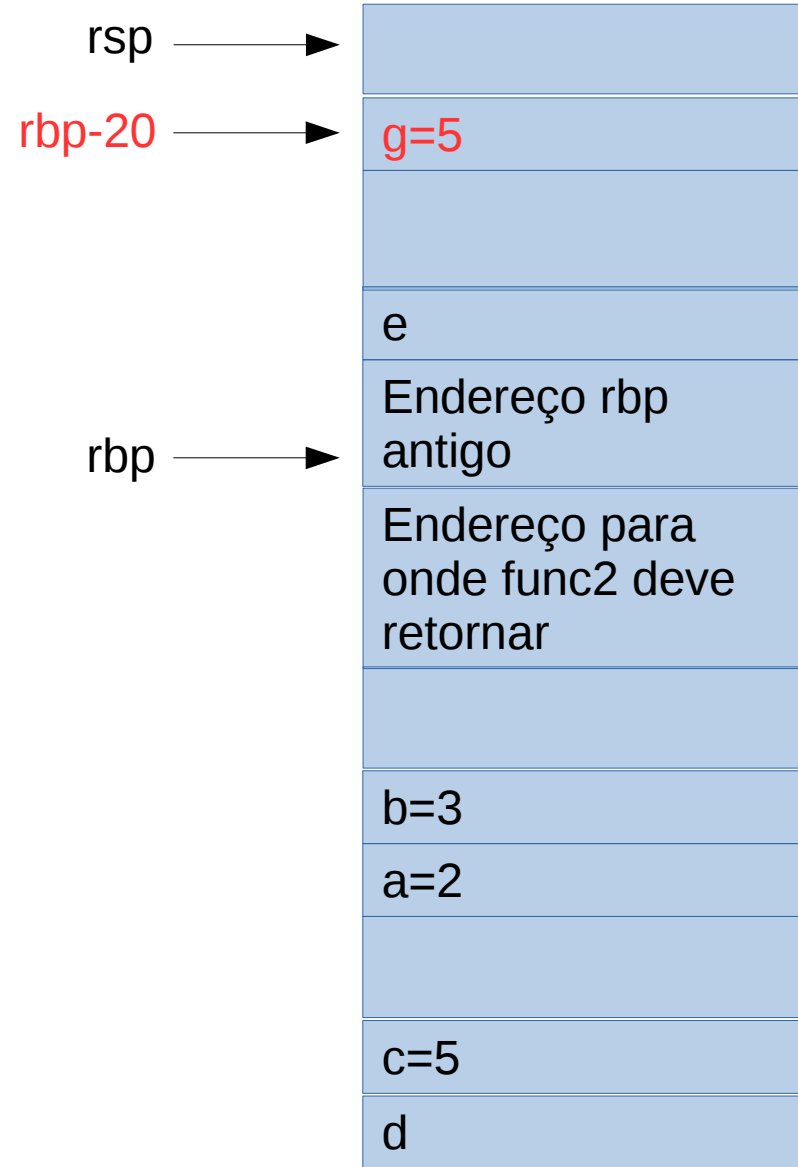
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

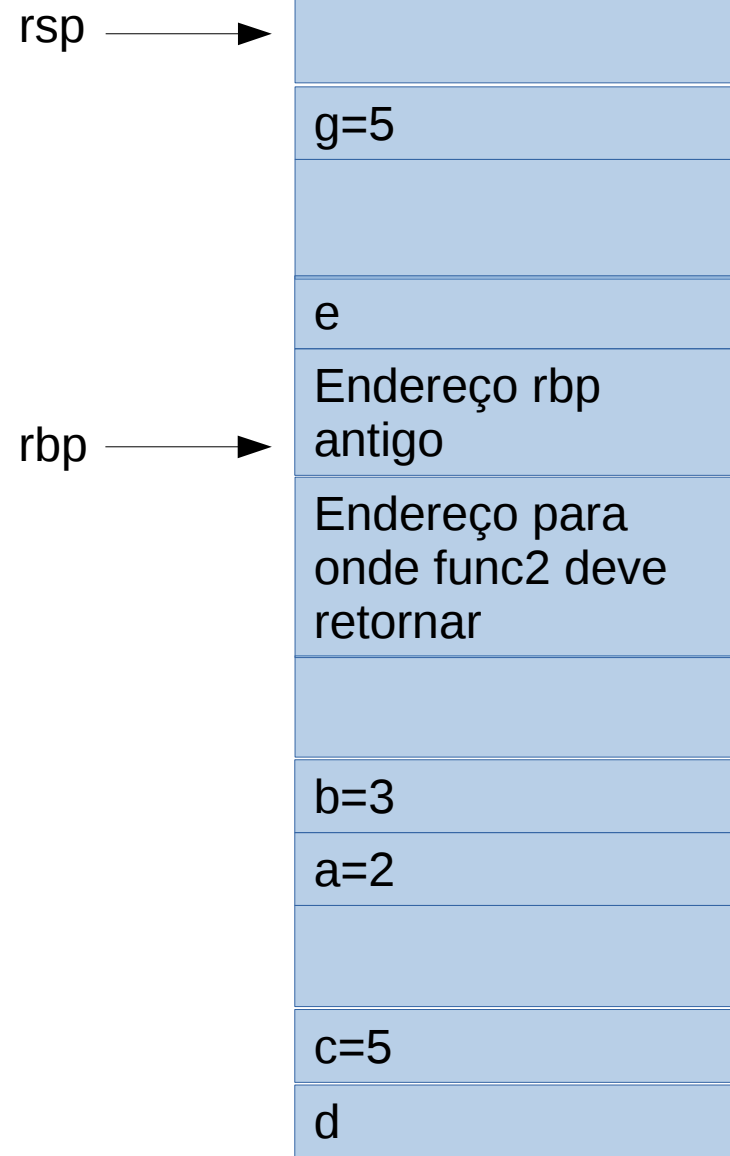
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax, 2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

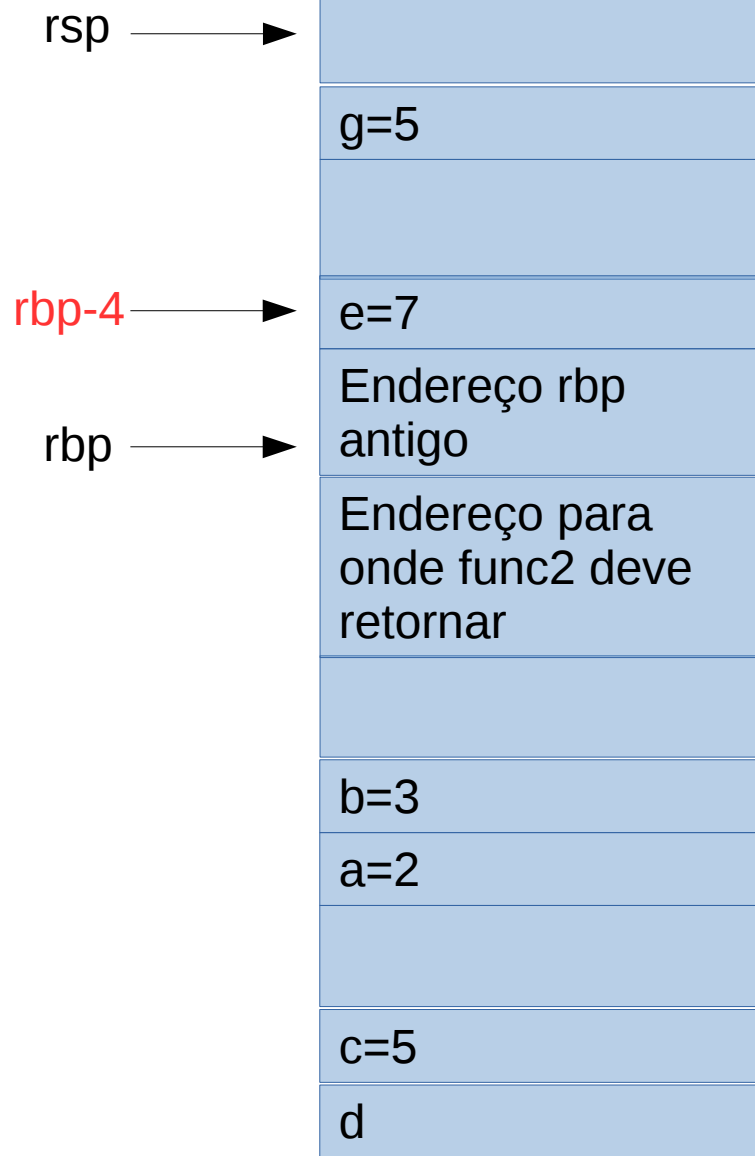
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

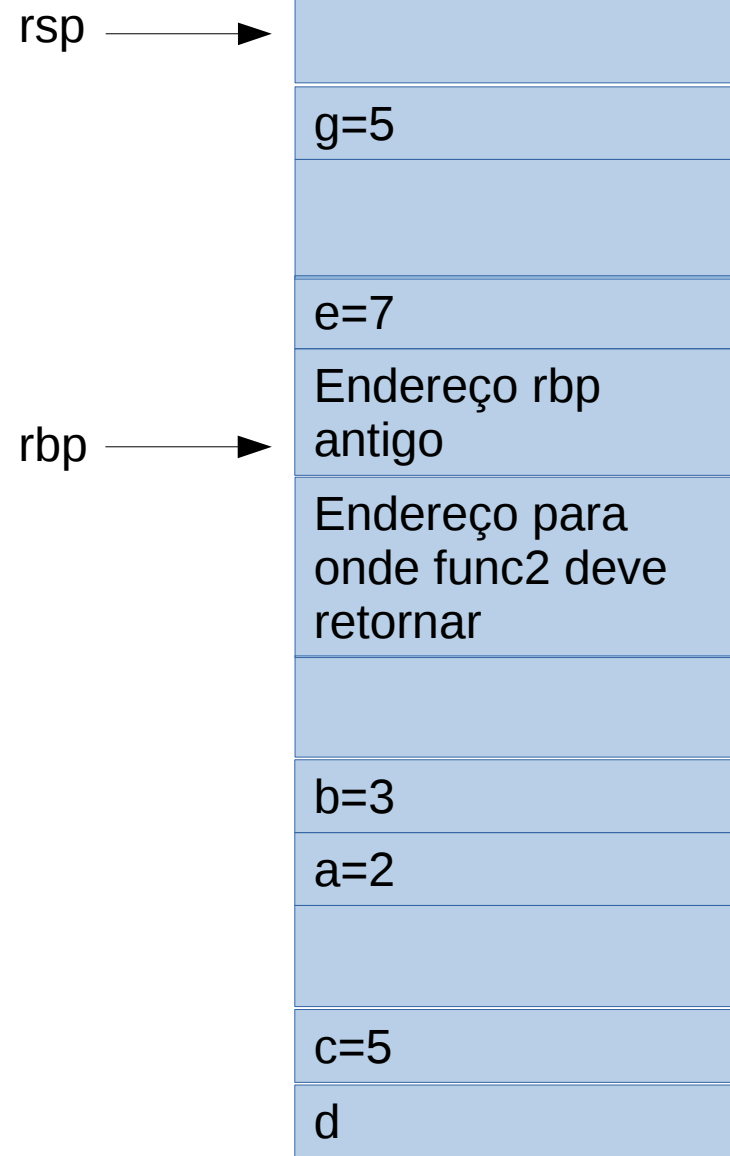
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret

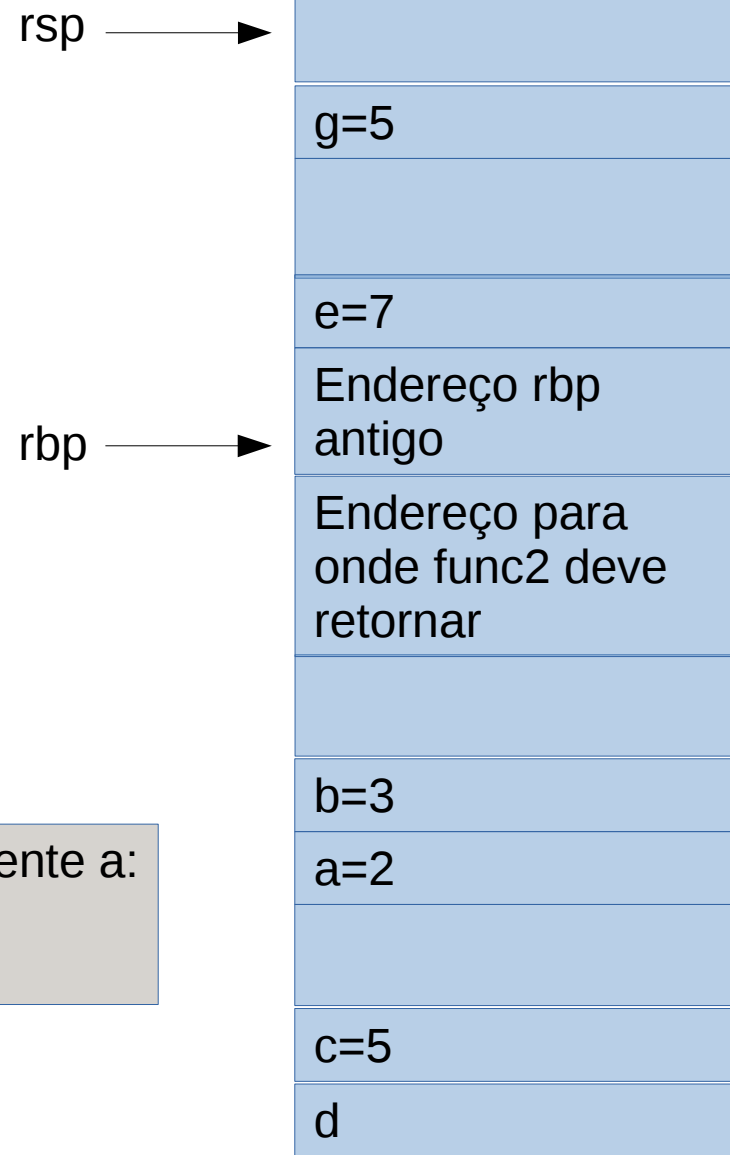
```

Instrução leave é equivalente a:

```

mov rsp,rbp
pop rbp

```



```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret
```

rsp →  
rbp →

Endereço rbp  
antigo

Endereço para  
onde func2 deve  
retornar

b=3

a=2

c=5

d

Instrução leave é equivalente a:  
**mov rsp,rbp**  
pop rbp

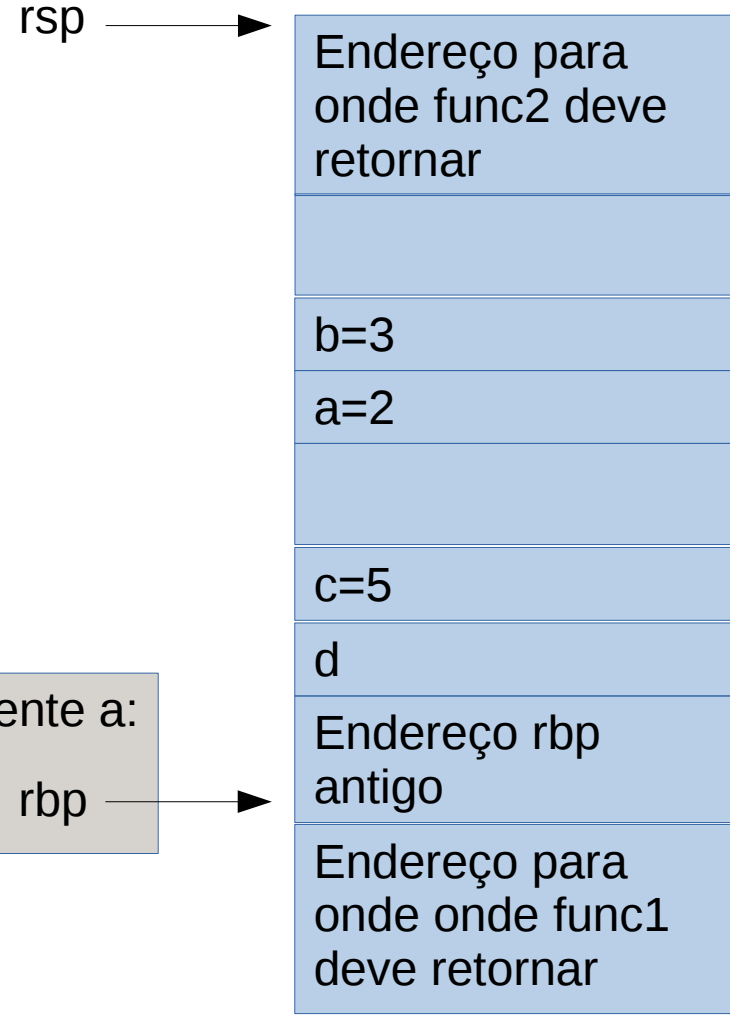
```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret
```

Instrução leave é equivalente a:  
 mov rsp,rbp  
 pop rbp



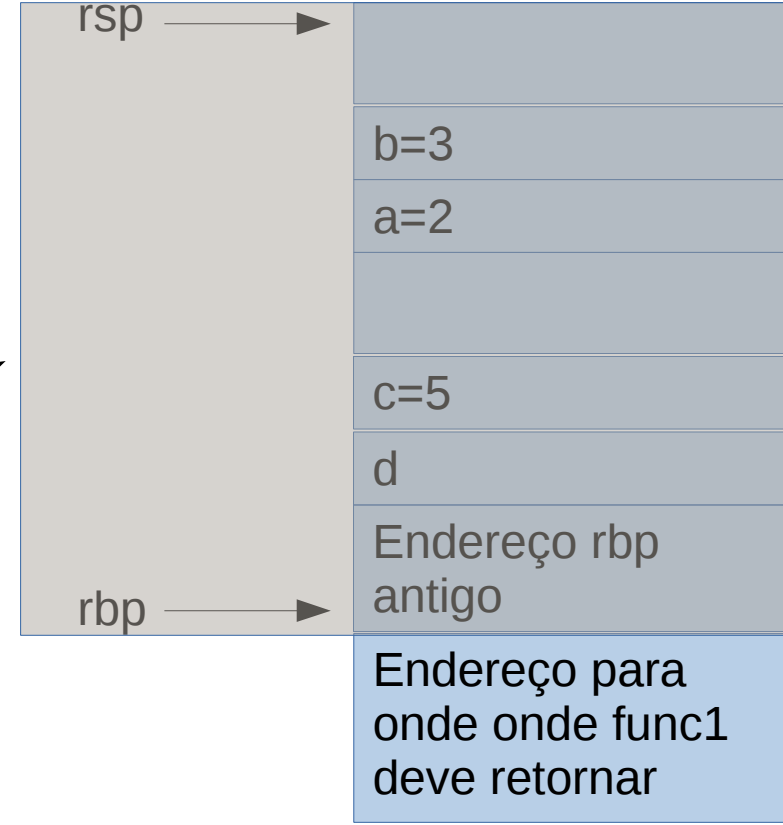
```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
func2:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 28
    mov     [rbp-20],edi
    mov     eax, [rbp-20]
    add     eax,2
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    imull   eax, [rbp-4]
    leave
    ret
```

Com a instrução ret de func2  
o stack frame de func1 é  
restaurado.





```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

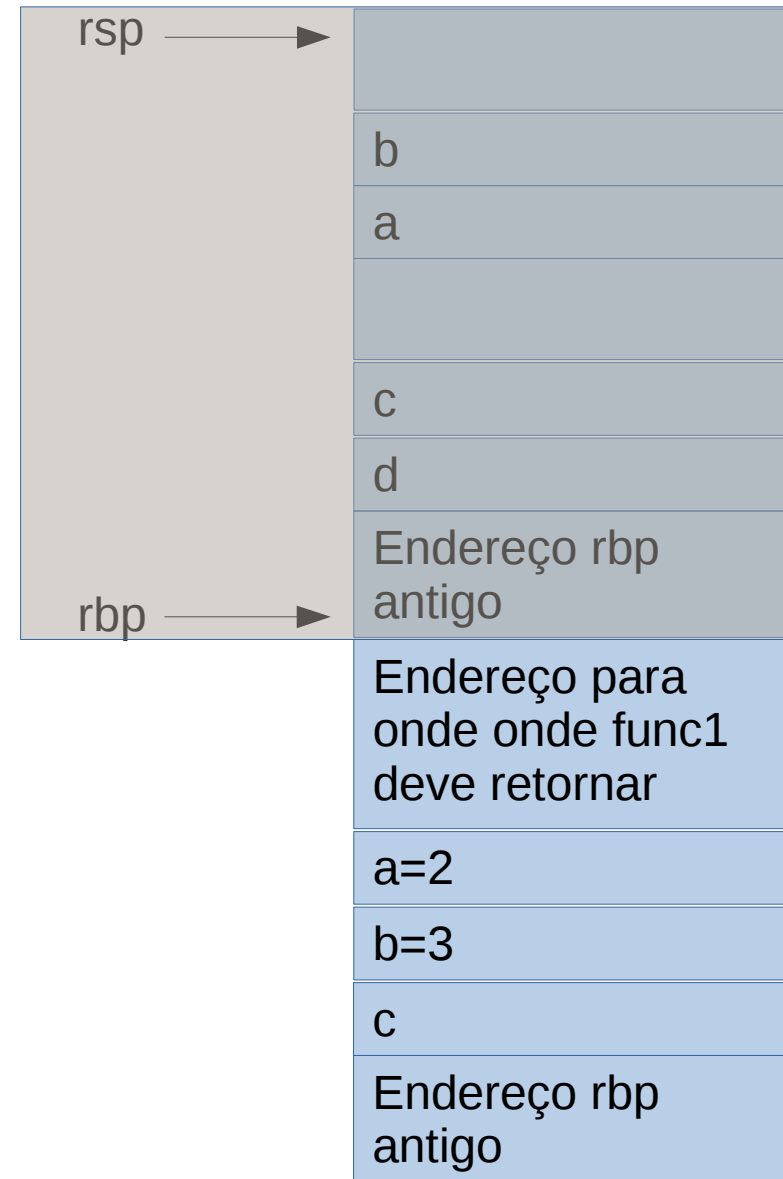
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
    
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    
    ret

```

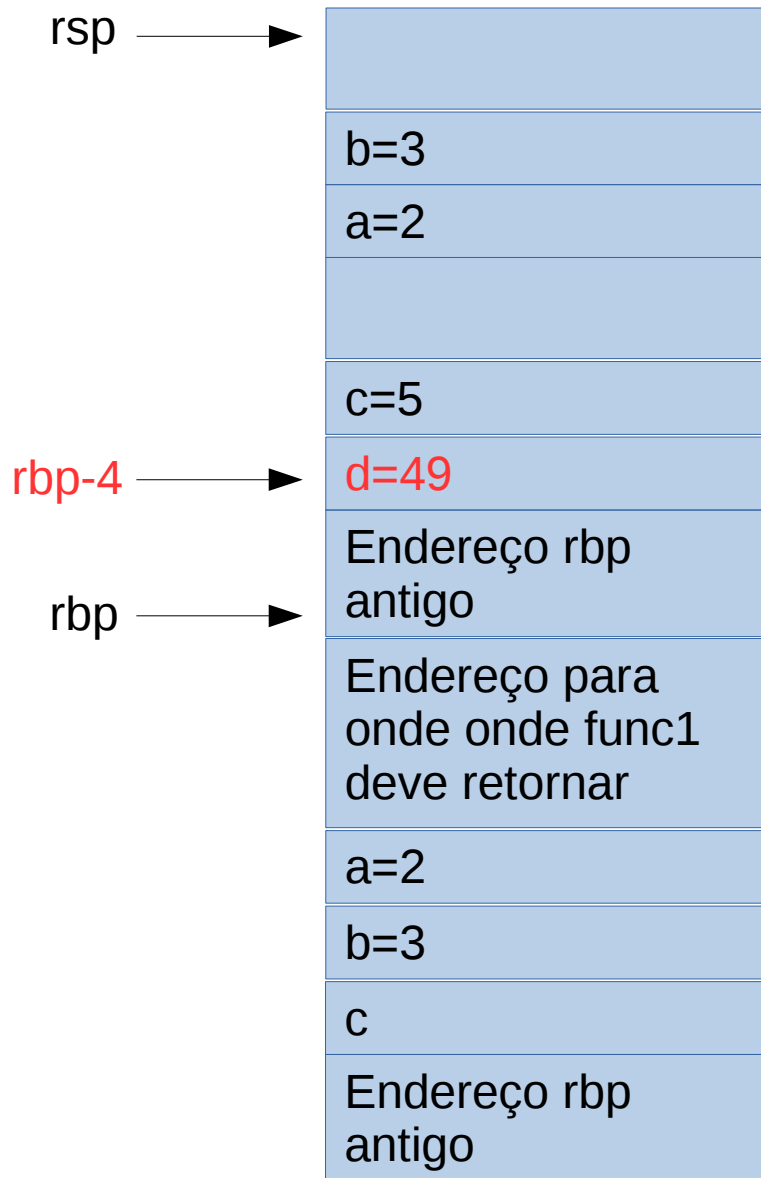


```
int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}
```

```
int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}
```

```
int func2(int g)
{
    int e = g+2;
    return e*e;
}
```

```
func1:
    push    rbp
    mov     rbp, rsp,
    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret
```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

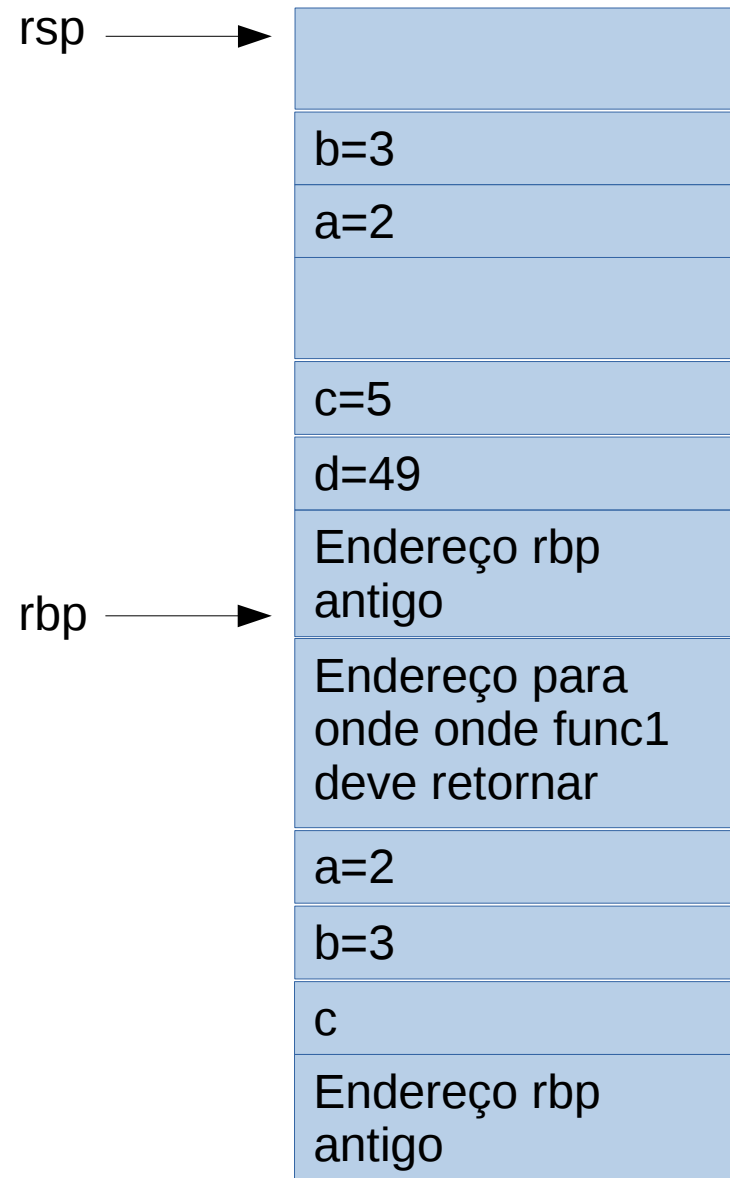
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,
        sub    rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```

rsp →  
rbp →

Endereço rbp antigo
Endereço para onde func1 deve retornar
a=2
b=3
c
Endereço rbp antigo

```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

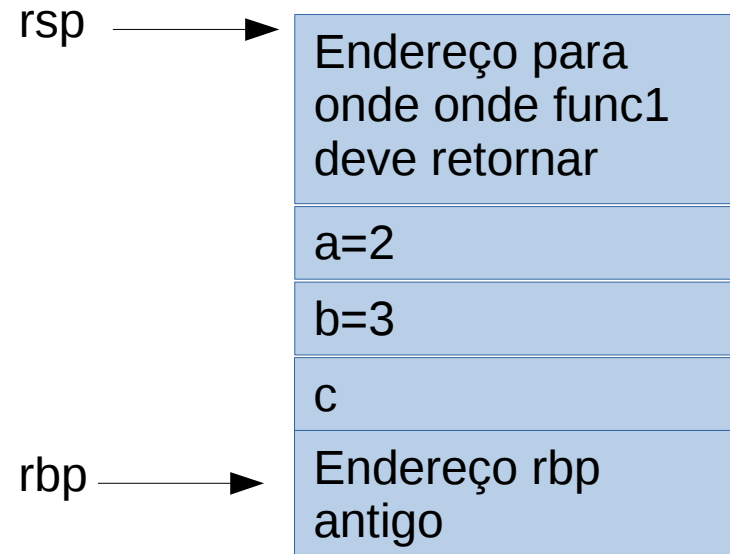
```

```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

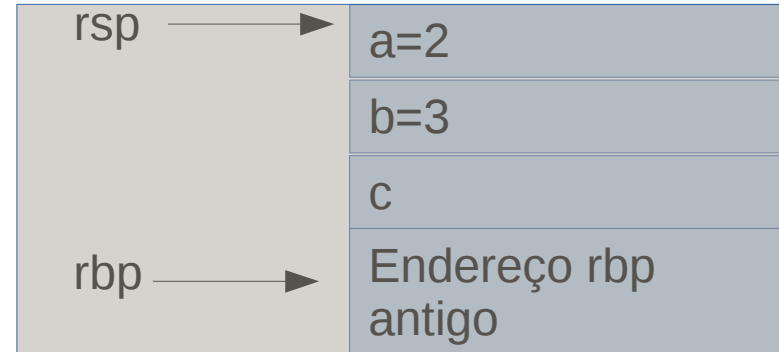
```

func1:
    push    rbp
    mov     rbp, rsp,

    sub     rsp,32
    movl    [rbp-20],edi
    movl    [rbp-24],esi
    movl    edx, [rbp-20]
    movl    eax, [rbp-24]
    addl    eax, edx
    movl    [rbp-8], eax
    movl    [rbp-8], eax
    movl    edi, eax
    call    func2
    movl    [rbp-4], eax
    movl    edx, [rbp-8]
    movl    eax, [rbp-4]
    addl    eax, edx
    leave

    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

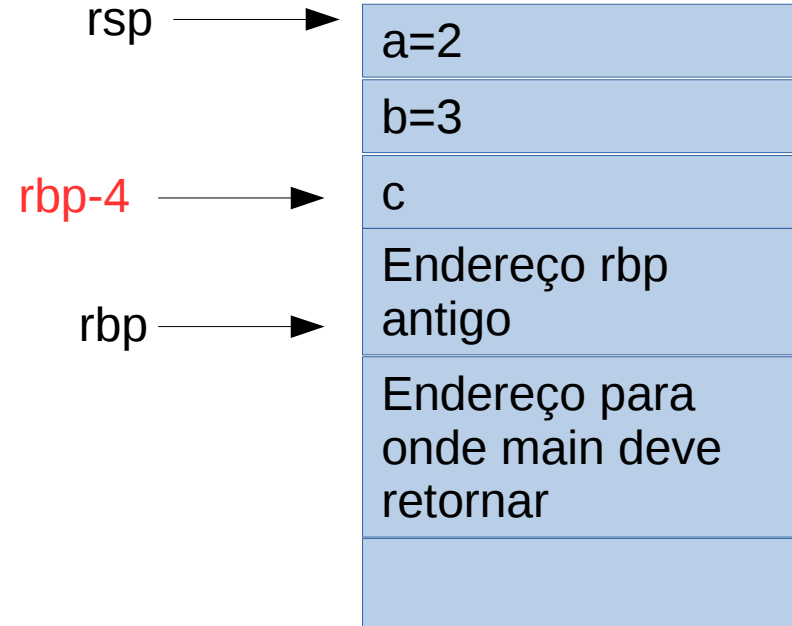
int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```



```

int main(void)
{
    int a, b, c;
    a = 2;
    b = 3;
    c = func1(a,b);
    return c;
}

```

```

int func1(int a, int b)
{
    int c,d;
    c = a+b;
    d = func2(c);
    return c+d;
}

```

```

int func2(int g)
{
    int e = g+2;
    return e*e;
}

```

```

main:
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     [rbp-12], 2
    mov     [rbp-8], 3
    mov     edx, [rbp-8]
    mov     eax, [rbp-12]
    mov     esi, edx
    mov     edi, eax
    call    func1
    mov     [rbp-4], eax
    mov     eax, [rbp-4]
    leave
    ret

```

