# Recursive Example

## factorial

```
0 \times 08048067 <+0>:
                            push
                                     %ebp
0 \times 08048068 < +1>:
                                     %esp,%ebp
                            mov
                                     0x8(%ebp),%eax
0 \times 0804806a <+3>:
                            mov
0 \times 0804806d <+6>:
                                     $0x1,%eax
                            \mathsf{cmp}
0 \times 08048070 < +9 > 1
                                     0x804807f <end_factorial>
                            jе
0 \times 08048072 < +11>:
                            dec
                                     %eax
0 \times 08048073 < +12>:
                            push
                                     %eax
                                     0x8048067 <factorial>
0 \times 08048074 < +13 > :
                            call
                                     0x8(%ebp),%ebx
0 \times 08048079 < +18 > :
                            mov
0 \times 0804807c < +21>:
                            imul
                                     %ebx,%eax
```

# \_start

```
0 \times 08048054 <+0>:
                           push
                                     $0x5
0 \times 08048056 <+2>:
                                     0x8048067 <factorial>
                           call
                                     $0x4,%esp
0 \times 0804805b < +7>:
                           add
0 \times 0804805e < +10>:
                                    %eax,%ebx
                           mov
0 \times 08048060 < +12 > :
                                    $0x1,%eax
                           mov
                                     $0x80
0 \times 08048065 < +17>:
                           int
```

# just before the call to factorial

#### start

0x08048054 <+0>: push \$0x5 0x8048067 <factorial>  $0 \times 08048056 <+2>$ : call  $0 \times 0804805b <+7>$ : add \$0x4,%esp  $0 \times 0804805e < +10>$ : %eax,%ebx mov  $0 \times 08048060 < +12>$ : \$0x1,%eax mov  $0 \times 08048065 < +17>$ : \$0x80 int

### Stack

0xffffd66c

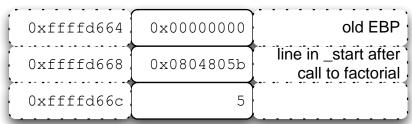
# just after the call to factorial(5)

#### factorial

```
0 \times 08048067 < +0>:
                          push
                                   %ebp
0 \times 08048068 < +1>:
                                   %esp,%ebp
                          mov
                                   0x8(%ebp),%eax
0x0804806a <+3>:
                          mov
0 \times 0804806d <+6>:
                                   $0x1,%eax
                          cmp
                                   0x804807f <end factorial>
0 \times 08048070 < +9>:
                          jе
0 \times 08048072 < +11>:
                          dec
                                   %eax
0 \times 08048073 < +12>:
                          push
                                   %eax
                          call
                                   0x8048067 <factorial>
0 \times 08048074 < +13>:
0 \times 08048079 < +18 > :
                                   0x8(%ebp),%ebx
                          mov
0 \times 0804807c < +21>:
                                   %ebx,%eax
                          imul
```

#### start

0×08048054		push	\$0x5	
0x08048056		call	0x8048067	<factorial></factorial>
0x0804805b	<+7> <b>:</b>	add	\$0x4,%esp	
0x0804805e	<+10>:	mov	%eax,%ebx	
0×08048060	<+12> <b>:</b>	mov	\$0x1,%eax	
0x08048065	<+17> <b>:</b>	int	\$0×80	



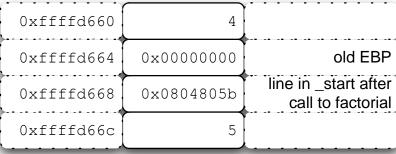
# about to call factorial(4)

#### factorial

```
0 \times 08048067 < +0>:
                          push
                                   %ebp
                                   %esp,%ebp
0 \times 08048068 < +1>:
                          mov
                                   0x8(%ebp),%eax
0x0804806a <+3>:
                          mov
0 \times 0804806d <+6>:
                                   $0x1,%eax
                          cmp
0 \times 08048070 < +9>:
                                   0x804807f <end factorial>
                          jе
0 \times 08048072 < +11>:
                          dec
                                   %eax
0 \times 08048073 < +12>:
                          push
                                   %eax
                          call
                                   0x8048067 <factorial>
0 \times 08048074 < +13>:
0 \times 08048079 < +18 > :
                                   0x8(%ebp),%ebx
                          mov
0 \times 0804807c < +21>:
                                   %ebx,%eax
                          imul
```

#### start

0x08048054	<+0>:	push	\$0x5	
0x08048056	<+2> <b>:</b>	call	0x8048067	<factorial></factorial>
0x0804805b	<+7> <b>:</b>	add	\$0x4,%esp	
0x0804805e	<+10>:	mov	%eax,%ebx	
0x08048060	<+12> <b>:</b>	mov	\$0x1,%eax	
0x08048065	<+17>:	int	\$0x80	



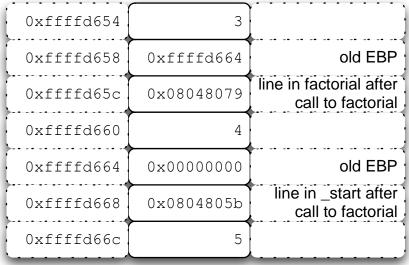
# about to call factorial(3)

### factorial

```
0×08048067 <+0>:
                          push
                                  %ebp
0 \times 08048068 < +1>:
                                  %esp,%ebp
                          mov
                                  0x8(%ebp),%eax
0 \times 0804806a < +3>:
                          mov
0 \times 0804806d <+6>:
                                   $0x1,%eax
                          cmp
                                  0x804807f <end factorial>
0 \times 08048070 < +9>:
                          jе
0 \times 08048072 < +11>:
                          dec
                                   %eax
0 \times 08048073 < +12>:
                          push
                                  %eax
                          call
                                  0x8048067 <factorial>
0 \times 08048074 < +13>:
0x08048079 <+18>:
                                  0x8(%ebp),%ebx
                          mov
0 \times 0804807c < +21>:
                                  %ebx,%eax
                          imul
```

#### start

0x08048054	<+0>:	push	\$0×5
0×08048056	<+2> <b>:</b>	call	0x8048067 <factorial></factorial>
0x0804805b	<+7> <b>:</b>	add	\$0x4,%esp
0x0804805e	<+10>:	mov	%eax,%ebx
0×08048060	<+12> <b>:</b>	mov	\$0x1,%eax
0x08048065	<+17>:	int	\$0×80



# about to call factorial(2)

#### factorial

```
0 \times 08048067 < +0>:
                           push
                                    %ebp
0 \times 08048068 < +1>:
                                   %esp,%ebp
                          mov
0 \times 0804806a < +3>:
                                    0x8(%ebp),%eax
                           mov
0 \times 0804806d <+6>:
                                    $0x1,%eax
                           cmp
                                    0x804807f <end factorial>
0 \times 08048070 < +9>:
                           jе
0 \times 08048072 < +11>:
                          dec
                                    %eax
0 \times 08048073 < +12>:
                           push
                                   %eax
0 \times 08048074 < +13>:
                           call
                                   0x8048067 <factorial>
0 \times 08048079 < +18 > :
                                   0x8(%ebp),%ebx
                           mov
0 \times 0804807c < +21>:
                                   %ebx,%eax
                           imul
```

### \_start

0x08048054	<+0>:	push	\$0x5	
0x08048056	<+2> <b>:</b>	call	0x8048067	<factorial></factorial>
0x0804805b	<+7> <b>:</b>	add	\$0x4,%esp	
0x0804805e	<+10>:	mov	%eax,%ebx	
0x08048060	<+12> <b>:</b>	mov	\$0x1,%eax	
0x08048065	<+17>:	int	\$0×80	

	2	0xffffd648
old EBP	0xffffd658	0xffffd64c
line in factorial after call to factorial	0x08048079	0xffffd650
	3	0xffffd654
old EBP	0xffffd664	0xffffd658
line in factorial after call to factorial	0x08048079	0xffffd65c
	4	0xffffd660
old EBP	0x00000000	0xffffd664
line in _start after call to factorial	0x0804805b	0xffffd668
	5	0xffffd66c

# about to call factorial(1)

#### factorial

```
0 \times 08048067 < +0>:
                           push
                                    %ebp
                                    %esp,%ebp
0 \times 08048068 < +1>:
                           mov
0 \times 0804806a < +3>:
                                    0x8(%ebp),%eax
                           mov
0 \times 0804806d <+6>:
                                    $0x1,%eax
                           cmp
                                    0x804807f <end factorial>
0 \times 08048070 < +9>:
                           jе
0 \times 08048072 < +11>:
                           dec
                                    %eax
0 \times 08048073 < +12>:
                           push
                                    %eax
0 \times 08048074 < +13 > :
                           call
                                    0x8048067 <factorial>
0 \times 08048079 < +18 > :
                                    0x8(%ebp),%ebx
                           mov
0 \times 0804807c < +21>:
                                   %ebx,%eax
                           imul
```

#### start

0x08048054	<+0>:	push	\$0x5	
0x08048056	<+2> <b>:</b>	call	0x8048067	<factorial></factorial>
0x0804805b	<+7> <b>:</b>	add	\$0x4,%esp	
0x0804805e	<+10>:	mov	%eax,%ebx	
0x08048060	<+12> <b>:</b>	mov	\$0x1,%eax	
0×08048065	<+17>:	int	\$0x80	

	1	0xffffd63c
old EBP	0xffffd64c	0xffffd640
line in factorial after call to factorial	0x08048079	0xffffd644
	2	0xffffd648
old EBP	0xffffd658	0xffffd64c
line in factorial after call to factorial	0x08048079	0xffffd650
	3	0xffffd654
old EBP	0xffffd664	0xffffd658
line in factorial after call to factorial	0x08048079	0xffffd65c
	4	0xffffd660
old EBP	0x00000000	0xffffd664
line in _start after call to factorial	0x0804805b	0xffffd668
	5	0xffffd66c

# in factorial(1)

### factorial

```
0 \times 08048067 < +0>:
                           push
                                    %ebp
0 \times 08048068 < +1>:
                                    %esp,%ebp
                           mov
0 \times 0804806a <+3>:
                                    0x8(%ebp),%eax
                           mov
0 \times 0804806d <+6>:
                                    $0x1,%eax
                           cmp
                                    0x804807f <end_factorial>
0 \times 08048070 < +9>:
                           jе
0 \times 08048072 < +11>:
                           dec
                                    %eax
0 \times 08048073 < +12>:
                           push
                                    %eax
0 \times 08048074 < +13 > :
                           call
                                    0x8048067 <factorial>
0 \times 08048079 < +18 > :
                           mov
                                    0x8(%ebp),%ebx
0 \times 0804807c < +21>:
                           imul
                                    %ebx,%eax
```

#### start

0x08048054	<+0>:	push	\$0x5	
0x08048056	<+2> <b>:</b>	call	0×8048067	<factorial></factorial>
0x0804805b	<+7>:	add	\$0x4,%esp	
0x0804805e	<+10>:	mov	%eax,%ebx	
0x08048060	<+12> <b>:</b>	mov	\$0x1,%eax	
0×08048065	<+17>:	int	\$0×80	

old EBP	0xffffd640	0xffffd634
line in factorial after call to factorial	0x08048079	0xffffd638
	1	0xffffd63c
old EBP	0xffffd64c	0xffffd640
line in factorial after call to factorial	0x08048079	0xffffd644
	2	0xffffd648
old EBP	0xffffd658	0xffffd64c
line in factorial after call to factorial	0x08048079	0xffffd650
	3	0xffffd654
old EBP	0xffffd664	0xffffd658
line in factorial after call to factorial	0x08048079	0xffffd65c
	4	0xffffd660
old EBP	0x00000000	0xffffd664
line in _start after call to factorial	0x0804805b	0xffffd668
	5	0xffffd66c

## How did we get these values? GDB.

- to get the code in assembly:
  - -disas label e.g.,
    - disas start
    - disas factorial
- to read the value stored in register %ebp
  - -p \$ebp
- to read a series of 20 values stored on the stack starting with the value stored in %esp:
  - -x/20x \$esp