Pointers in Assembly

Adalbert Gerald Soosai Raj
swap() in C
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);

    x: 1
    0x108
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;

    swap(&x, &y);
    x: 1
0x108
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
    x: 1  y: 2
    0x108 0x104
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py) {
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main() {
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;
    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py) {
    int x = *px;
    int y = *py;
    
    *px = y;
    *py = x;
}

int main() {
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py) {
    int x = *px;
    int y = *py;
    *px = y;
    *py = x;
}

int main() {
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py) {
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main() {
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py)
{
    int x = *px;
    int y = *py;

    *px = y;
    *py = x;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(&x, &y);
}
void swap(int *px, int *py) {
    int x = *px;
    int y = *py;
    *px = y;
    *py = x;
}

int main() {
    int x = 1;
    int y = 2;
    swap(&x, &y);
}

Control returns to main!
Assembly
main() function
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  $1, -4(%ebp)
movl  $2, -8(%ebp)
leal  -8(%ebp), %eax
pushl  %eax
leal  -4(%ebp), %eax
pushl  %eax
call  swap
addl  $8, %esp
leave
ret
main:

    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:

```
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret
```
main:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   $1, -4(%ebp)
  movl   $2, -8(%ebp)
  leal   -8(%ebp), %eax
  pushl  %eax
  leal   -4(%ebp), %eax
  pushl  %eax
  call   swap
  addl   $8, %esp
  leave
  ret
main:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl $1, -4(%ebp)
  movl $2, -8(%ebp)
  leal -8(%ebp), %eax
  pushl %eax
  leal -4(%ebp), %eax
  pushl %eax
  call swap
  addl $8, %esp
  leave
  ret
main:
    pushl  %ebp
    movl   %esp, %ebp
    subl   $16, %esp
    movl   $1, -4(%ebp)
    movl   $2, -8(%ebp)
    leal   -8(%ebp), %eax
    pushl  %eax
    leal   -4(%ebp), %eax
    pushl  %eax
    call   swap
    addl   $8, %esp
    leave
    ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl $1, -4(%ebp)
  movl $2, -8(%ebp)
  leal -8(%ebp), %eax
  pushl %eax
  leal -4(%ebp), %eax
  pushl %eax
  call swap
  addl $8, %esp
  leave
  ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   $1, -4(%ebp)
  movl   $2, -8(%ebp)
  leal   -8(%ebp), %eax
  pushl  %eax
  leal   -4(%ebp), %eax
  pushl  %eax
  call   swap
  addl   $8, %esp
  leave
  ret

```
caller's frame

<table>
<thead>
<tr>
<th>Saved %ebp</th>
<th>return address</th>
</tr>
</thead>
<tbody>
<tr>
<td>x: 1 0x108</td>
<td></td>
</tr>
<tr>
<td>y: 2 0x104</td>
<td></td>
</tr>
</tbody>
</table>

%ebp
%esp
```

```
x:  1 0x108
y:  2 0x104
```
main:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret

-8(%ebp) = -8 + R[%ebp] = 0x104
main:
    pushl  %ebp
    movl   %esp, %ebp
    subl   $16, %esp
    movl   $1, -4(%ebp)
    movl   $2, -8(%ebp)
    leal   -8(%ebp), %eax
    pushl  %eax
    leal   -4(%ebp), %eax
    pushl  %eax
    call   swap
    addl   $8, %esp
    leave
    ret

caller's frame

<table>
<thead>
<tr>
<th></th>
<th>x: 1</th>
<th>0x108</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>y: 2</td>
<td>0x104</td>
</tr>
</tbody>
</table>

Saved %ebp
main:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   $1, -4(%ebp)
  movl   $2, -8(%ebp)
  leal   -8(%ebp), %eax
  pushl  %eax
  leal   -4(%ebp), %eax
  pushl  %eax
  call   swap
  addl   $8, %esp
  leave
  ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
```
main:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret
```
main:
    pushl  %ebp
    movl   %esp, %ebp
    subl   $16, %esp
    movl   $1, -4(%ebp)
    movl   $2, -8(%ebp)
    leal   -8(%ebp), %eax
    pushl  %eax
    leal   -4(%ebp), %eax
    pushl  %eax
    call   swap
    addl   $8, %esp
    leave
    ret

<table>
<thead>
<tr>
<th>caller's frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>return address</td>
</tr>
<tr>
<td>Saved %ebp</td>
</tr>
<tr>
<td>%ebp</td>
</tr>
<tr>
<td>x: 1</td>
</tr>
<tr>
<td>y: 2</td>
</tr>
<tr>
<td>py: 0x104</td>
</tr>
<tr>
<td>%esp</td>
</tr>
</tbody>
</table>

%eax

0x104
main:
    pushl  %ebp
    movl  %esp, %ebp
    subl  $16, %esp
    movl  $1, -4(%ebp)
    movl  $2, -8(%ebp)
    leal  -8(%ebp), %eax
    pushl  %eax
    leal  -4(%ebp), %eax
    pushl  %eax
    call  swap
    addl  $8, %esp
    leave
    ret
main:

pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    $1, -4(%ebp)
movl    $2, -8(%ebp)
leal    -8(%ebp), %eax
pushl   %eax
leal    -4(%ebp), %eax
pushl   %eax
call    swap
addl    $8, %esp
leave
ret

-le4(%ebp) = -8 + R[%ebp] = 0x108
main:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl $1, -4(%ebp)
  movl $2, -8(%ebp)
  leal -8(%ebp), %eax
  pushl %eax
  leal -4(%ebp), %eax
  pushl %eax
  call swap
  addl $8, %esp
  leave
  ret
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl $1, -4(%ebp)
  movl $2, -8(%ebp)
  leal -8(%ebp), %eax
  pushl %eax
  leal -4(%ebp), %eax
  pushl %eax
  call swap
  addl $8, %esp
  leave
  ret

caller's frame
  return address
  Saved %ebp
  %ebp
  x: 1 0x108
  y: 2
  py: 0x104
  0x108

%es%p
%eax

0x108
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret

 caller's frame

<table>
<thead>
<tr>
<th>return address</th>
</tr>
</thead>
<tbody>
<tr>
<td>%ebp</td>
</tr>
<tr>
<td>Saved %ebp</td>
</tr>
<tr>
<td>x: 1</td>
</tr>
<tr>
<td>py: 0x104</td>
</tr>
<tr>
<td>px: 0x108</td>
</tr>
<tr>
<td>y: 2</td>
</tr>
<tr>
<td>0x104</td>
</tr>
<tr>
<td>0x108</td>
</tr>
</tbody>
</table>
main:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret

<table>
<thead>
<tr>
<th>caller's frame</th>
<th>return address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saved %ebp</td>
<td></td>
</tr>
<tr>
<td>x:</td>
<td>1</td>
</tr>
<tr>
<td>y:</td>
<td>2</td>
</tr>
<tr>
<td>py:</td>
<td>0x104</td>
</tr>
<tr>
<td>px:</td>
<td>0x108</td>
</tr>
</tbody>
</table>

%ebp

%esp

%eax

0x108
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret

 caller's frame
 return address
 %ebp
 Saved %ebp
 x: 1 0x108
 y: 2 0x104
 py: 0x104
 px: 0x108
 return address
 %esp
 %eax
 0x108

%esp

%ebp

%eax

0x108

0x104

0x108

0x104

0x108

%esp

%ebp

%eax
swap() function
swap:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl 8(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -4(%ebp)
  movl 12(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -8(%ebp)
  movl 8(%ebp), %eax
  movl -8(%ebp), %edx
  movl %edx, (%eax)
  movl 12(%ebp), %eax
  movl -4(%ebp), %edx
  movl %edx, (%eax)
leave
ret
swap:
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
swap:

```
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
```
swap:

```
pushl  %ebp
movl   %esp, %ebp
subl   $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
```
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
...movl -8(%ebp), %edx
leave
ret
swap:

```assembly
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
```
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
swap:

```
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
```
swap:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl 8(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -4(%ebp)
  movl 12(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -8(%ebp)
  movl 8(%ebp), %eax
  movl -8(%ebp), %edx
  movl %edx, (%eax)
  movl 12(%ebp), %eax
  movl -4(%ebp), %edx
  movl %edx, (%eax)
leave
ret
swap:

pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp

movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
swap:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -8(%ebp)
    movl 8(%ebp), %eax
    movl -8(%ebp), %edx
    movl %edx, (%eax)
    movl 12(%ebp), %eax
    movl -4(%ebp), %edx
    movl %edx, (%eax)
    leave
    ret
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
```

\[
(%eax) = M[0x108] = 1
\]
swap:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   8(%ebp), %eax
  movl   (%eax), %eax
  movl   %eax, -4(%ebp)
  movl   12(%ebp), %eax
  movl   (%eax), %eax
  movl   %eax, -8(%ebp)
  movl   8(%ebp), %eax
  movl   -8(%ebp), %edx
  movl   %edx, (%eax)
  movl   12(%ebp), %eax
  movl   -4(%ebp), %edx
  movl   %edx, (%eax)
  leave
  ret
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:

    pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
swap:

```
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
```
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl -8(%ebp), %edx
    movl 12(%ebp), %eax
    movl -4(%ebp), %edx
    movl %edx, (%eax)
leave
ret
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl %edx, (%eax)
leave
ret

(%eax) = M[0x104] = 2
swap:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl %eax, -4(%ebp)
movl 8(%ebp), %eax
movl %eax, -8(%ebp)
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret

Saved %ebp
0x104
0x108
return address
py: 0x104
px: 0x108

in swap()
swap:
  pushl  %ebp
  movl  %esp, %ebp
  subl  $16, %esp
  movl  8(%ebp), %eax
  movl  (%eax), %eax
  movl  %eax, -4(%ebp)
  movl  12(%ebp), %eax
  movl  (%eax), %eax
  movl  %eax, -8(%ebp)
  movl  8(%ebp), %eax
  movl  %eax, -8(%ebp)
  movl  8(%ebp), %eax
  movl  -8(%ebp), %edx
  movl  %edx, (%eax)
  movl  12(%ebp), %eax
  movl  -4(%ebp), %edx
  movl  %edx, (%eax)
  leave
  ret
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -8(%ebp)
    movl 8(%ebp), %eax
    movl -8(%ebp), %edx
    movl %edx, (%eax)
    movl 12(%ebp), %eax
    movl -4(%ebp), %edx
    movl %edx, (%eax)
    leave
    ret
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   8(%ebp), %eax
  movl   (%eax), %eax
  movl   %eax, -4(%ebp)
  movl   12(%ebp), %eax
  movl   (%eax), %eax
  movl   %eax, -8(%ebp)
  movl   8(%ebp), %eax
  movl   -8(%ebp), %edx
  movl   %edx, (%eax)
  movl   12(%ebp), %eax
  movl   -4(%ebp), %edx
  movl   %edx, (%eax)
  leave
  ret

Saved %ebp

0x104
0x108

return address

py: 0x104
px: 0x108

%ebp
%esp
%eax

2 1
x: 1 0x108
y: 2 0x104

0x104
0x108

%ebp
%esp
%eax

2

%ebp
%esp
%eax
swap:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %edx
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret

return address
Saved %ebp
%ebp
%esp
x: 1 0x108
y: 2 0x104
px: 0x108
py: 0x104

%ebp
%esp
%eax
2
swap:

```
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl %eax, -8(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %edx, (%eax)
lea
ret
```
swap:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -8(%ebp)
    movl 8(%ebp), %eax
    movl -8(%ebp), %edx
    movl %edx, (%eax)
    movl 12(%ebp), %eax
    movl -4(%ebp), %edx
    movl %edx, (%eax)
    leave
    ret

\[
x: 1 \quad y: 2 \quad px: 0x108 \\
y: 2 \quad px: 0x108 \quad py: 0x104
\]
swap:
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
swap:

    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -8(%ebp)
    movl 8(%ebp), %eax
    movl -8(%ebp), %edx
    movl %edx, (%eax)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %edx, (%eax)
    leave
    ret
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```

(\%eax) = M[0x108]
swap:
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
swap:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl 8(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -4(%ebp)
  movl 12(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -8(%ebp)
  movl 8(%ebp), %eax
  movl %eax, -8(%ebp), %edx
  movl %edx, (%eax)
  movl 12(%ebp), %eax
  movl (%eax), %eax
  movl %edx, (%eax)
leave
ret
swap:
  pushl  %ebp
  movl  %esp, %ebp
  subl  $16, %esp
  movl  8(%ebp), %eax
  movl    (%eax), %eax
  movl    %eax, -4(%ebp)
  movl  12(%ebp), %eax
  movl    (%eax), %eax
  movl    %eax, -8(%ebp)
  movl  8(%ebp), %eax
  movl  -8(%ebp), %edx
  movl    %edx, (%eax)
  movl  12(%ebp), %eax
  movl  -4(%ebp), %edx
  movl    %edx, (%eax)
  leave
  ret
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
swap:

pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%eax), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
swap:
  pushl %ebp
  movl %esp, %ebp
  subl $16, %esp
  movl 8(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -4(%ebp)
  movl 12(%ebp), %eax
  movl (%eax), %eax
  movl %eax, -8(%ebp)
  movl 8(%ebp), %eax
  movl %eax, -8(%ebp)
  movl 8(%ebp), %eax
  movl -8(%ebp), %edx
  movl %edx, (%eax)
  movl 12(%ebp), %eax
  movl -4(%ebp), %edx
  movl %edx, (%eax)
leave
ret
swap:

    pushl  %ebp
    movl   %esp, %ebp
    subl   $16, %esp
    movl   8(%ebp), %eax
    movl   (%eax), %eax
    movl   %eax, -4(%ebp)
    movl   12(%ebp), %eax
    movl   (%eax), %eax
    movl   %eax, -8(%ebp)
    movl   8(%ebp), %eax
    movl   -8(%ebp), %edx
    movl   %edx, (%eax)
    movl   12(%ebp), %eax
    movl   -4(%ebp), %edx
    movl   %edx, (%eax)
    leave
    ret

return address

Saved %ebp

%ebp

%esp

%eax

%edx

0x104

0x108
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl 8(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -4(%ebp)
    movl 12(%ebp), %eax
    movl (%eax), %eax
    movl %eax, -8(%ebp)
    movl 8(%ebp), %eax
    movl -8(%ebp), %edx
    movl %edx, (%eax)
    movl 12(%ebp), %eax
    movl -4(%ebp), %edx
    movl %edx, (%eax)
    leave
    ret
swap:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl %edx, (%eax)
leave
ret

y: 1
x: 2
py: 0x104
px: 0x108
return address
Saved %ebp
%ebp
%eax
%edx
0x104
0x108
swap:

```
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
```
swap:

```
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)

leave
ret
```
swap:

```assembly
pushl   %ebp
movl    %esp, %ebp
subl    $16, %esp
movl    8(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -4(%ebp)
movl    12(%ebp), %eax
movl    (%eax), %eax
movl    %eax, -8(%ebp)
movl    8(%ebp), %eax
movl    -8(%ebp), %edx
movl    %edx, (%eax)
movl    12(%ebp), %eax
movl    -4(%ebp), %edx
movl    %edx, (%eax)
leave
ret
```
swap:

```
pushl  %ebp
movl  %esp, %ebp
subl  $16, %esp
movl  8(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -4(%ebp)
movl  12(%ebp), %eax
movl  (%eax), %eax
movl  %eax, -8(%ebp)
movl  8(%ebp), %eax
movl  -8(%ebp), %edx
movl  %edx, (%eax)
movl  12(%ebp), %eax
movl  -4(%ebp), %edx
movl  %edx, (%eax)
leave
ret
```
swap:

```assembly
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl 8(%ebp), %eax
movl (%eax), %eax
movl %eax, -4(%ebp)
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl 12(%ebp), %eax
movl (%eax), %eax
movl %eax, -8(%ebp)
movl 8(%ebp), %eax
movl -8(%ebp), %edx
movl %edx, (%eax)
movl 12(%ebp), %eax
movl -4(%ebp), %edx
movl %edx, (%eax)
leave
ret
```
return to main()
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret

Remember?

caller's frame

<table>
<thead>
<tr>
<th>return address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saved %ebp</td>
</tr>
</tbody>
</table>

|  x: 1 |
|  y: 2 |
|  px: &x |
|  py: &y |
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
  pushl  %ebp
  movl   %esp, %ebp
  subl   $16, %esp
  movl   $1, -4(%ebp)
  movl   $2, -8(%ebp)
  leal   -8(%ebp), %eax
  pushl  %eax
  leal   -4(%ebp), %eax
  pushl  %eax
  call   swap
  addl   $8, %esp
  leave
  ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
    addl $8, %esp
    leave
    ret
main:
    pushl  %ebp
    movl  %esp, %ebp
    subl  $16, %esp
    movl  $1, -4(%ebp)
    movl  $2, -8(%ebp)
    leal  -8(%ebp), %eax
    pushl  %eax
    leal  -4(%ebp), %eax
    pushl  %eax
    call  swap
    addl  $8, %esp
leave
ret
main:
  pushl  %ebp
  movl    %esp, %ebp
  subl    $16, %esp
  movl    $1, -4(%ebp)
  movl    $2, -8(%ebp)
  leal    -8(%ebp), %eax
  pushl   %eax
  leal    -4(%ebp), %eax
  pushl   %eax
  call    swap
addl    $8, %esp
leave
ret
main:
    pushl %ebp
    movl %esp, %ebp
    subl $16, %esp
    movl $1, -4(%ebp)
    movl $2, -8(%ebp)
    leal -8(%ebp), %eax
    pushl %eax
    leal -4(%ebp), %eax
    pushl %eax
    call swap
 leave
    ret
main:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $16, %esp
    movl    $1, -4(%ebp)
    movl    $2, -8(%ebp)
    leal    -8(%ebp), %eax
    pushl   %eax
    leal    -4(%ebp), %eax
    pushl   %eax
    call    swap
    addl    $8, %esp
    leave
    ret
main:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -4(%ebp)
movl $2, -8(%ebp)
leal -8(%ebp), %eax
pushl %eax
leal -4(%ebp), %eax
pushl %eax
call swap
addl $8, %esp
leave
ret
finish
ret
Questions?