# Local or global

main.c

```
1 #include <stdio.h>
2
3 int x=3;
4
5 int main(){
6   int x=5;
7   printf("x = %d\n", x);
8 }
```

## 生成汇编代码

```
1 gcc -S main.c
```

```
.file "main.c"
2
       .text
3
       .globl x
4
       .data
 5
       .align 4
 6
   x:
7
       .long 3
       .def __main; .scl 2; .type 32; .endef
8
9
       .section .rdata, "dr"
   .LC0:
10
      .ascii "x = %d\12\0"
11
12
       .text
13
       .globl main
14
       .def main; .scl 2; .type 32; .endef
15
       .seh_proc main
   main:
16
17
       pushq %rbp
18
       .seh_pushreg %rbp
19
             %rsp, %rbp
       movq
       .seh_setframe %rbp, 0
20
21
       subq $48, %rsp
       .seh_stackalloc 48
22
23
       .seh_endprologue
```

```
__main
24
       call
              $5, -4(%rbp)
25
       mov1
             -4(%rbp), %eax
26
       mov1
             %eax, %edx
27
      mo∨l
28
      leaq
             .LC0(%rip), %rcx
29
      call
             printf
             $0, %eax
30
       mov1
       addq $48, %rsp
31
             %rbp
32
       popq
33
       ret
34
       .seh_endproc
       .ident "GCC: (GNU) 10.2.0"
35
       .def printf; .scl 2; .type 32; .endef
36
```

### 汇编,链接,运行

```
1 gcc -c main.s
2 gcc main.o
3 .\a.exe
```

### 得到输出

```
1 \mid x = 5
```

# 在汇编代码中可以看到, 局部变量 x 被放到了栈上

```
1 mov1 $5, -4(%rbp)
```

## 而全局变量x被放到了rdata段

```
1 x:
2 .long 3
3 .def __main; .scl 2; .type 32; .endef
4 .section .rdata,"dr"
```

## 输出的关键部分在第26行到第29行

```
1 movl -4(%rbp), %eax
2 movl %eax, %edx
3 leaq .LCO(%rip), %rcx
4 call printf
```

这里通过 rbp 取了栈上的局部变量(x), 作为 printf() 的参数, 要想让程序输出全局变量的x, 只要把这里改为取全局的x即可

```
1 movl x(%rip), %eax
2 movl %eax, %edx
3 leaq .LCO(%rip), %rcx
4 call printf
```

## 修改后的完整代码

```
1
       .file
               "main.c"
 2
       .text
 3
       .globl x
4
       .data
 5
       .align 4
6
   x:
7
       .long
               3
              __main; .scl 2; .type 32; .endef
       .def
8
       .section .rdata,"dr"
9
10
   .LC0:
       .ascii "x = %d\12\0"
11
12
       .text
13
       .globl main
       .def main; .scl 2; .type 32; .endef
14
15
       .seh_proc main
   main:
16
17
       pushq %rbp
18
       .seh_pushreg %rbp
19
       movq %rsp, %rbp
       .seh_setframe %rbp, 0
20
21
       subq
             $48, %rsp
22
       .seh_stackalloc 48
23
       .seh_endprologue
       call ___main
24
       mov1 $5, -4(%rbp)
25
              x(%rip), %eax
26
       mov1
              %eax, %edx
27
       mov1
             .LCO(%rip), %rcx
28
       leaq
29
       call
              printf
30
       mov1
              $0, %eax
              $48, %rsp
       addq
31
              %rbp
32
       popq
33
       ret
```

```
.seh_endproc
.ident "GCC: (GNU) 10.2.0"
.def printf; .scl 2; .type 32; .endef
```

# 重新汇编,链接,运行

```
1 gcc -c main.s
2 gcc main.o
3 .\a.exe
```

# 得到运行结果

```
1 \mid x = 3
```