CSCI 331: Introduction to Computer Security

Lecture 13: How C passes arguments

Instructor: Dan Barowy

Williams

Topics

Solution to Lab 4

How C passes arguments

Announcements

1. TA applications due tomorrow.
   Please consider "giving back."
2. Sandia National Labs
   Internships in Cybersecurity R&D

   https://www.sandia.gov/careers/career-possibilities/students-and-postdocs/internships-co-ops/institute-programs/titans-technical-internships-to-advance-national-security/

   https://cg.sandia.gov

   (American citizens only—sorry!)

Your to-dos

1. Project part 2, **due Sunday 10/22**.
2. Read and take notes (Wang) **for Thur 11/2**.
3. Lab 5, **due ~~Sunday 10/29~~**.
   Sunday 11/5

Lab 5 walkthrough

Required Readings

Assembly Level Debugging with GDB

Finding a Return Address on the Stack (video)

Creating a Shellcode File

Disabling security features

GDB

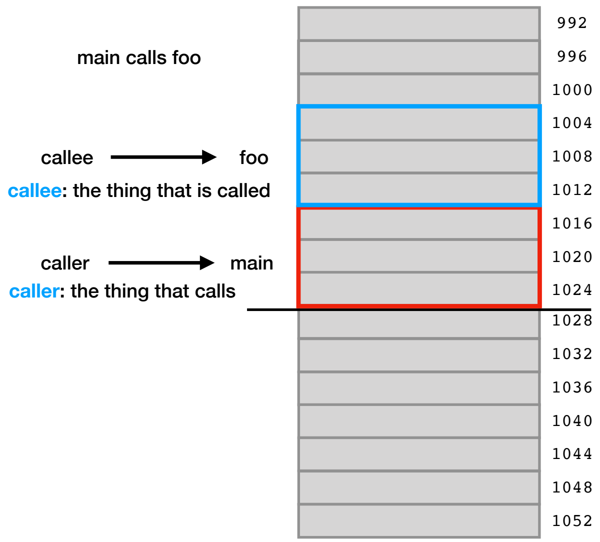
Paper discussion

The program you examined in lab 4

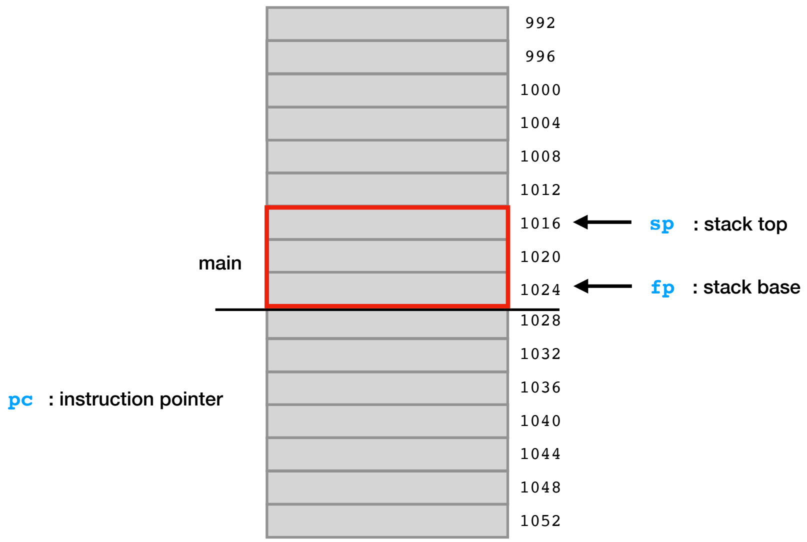
```
void foo() {}

int main() {
  foo();
}
```

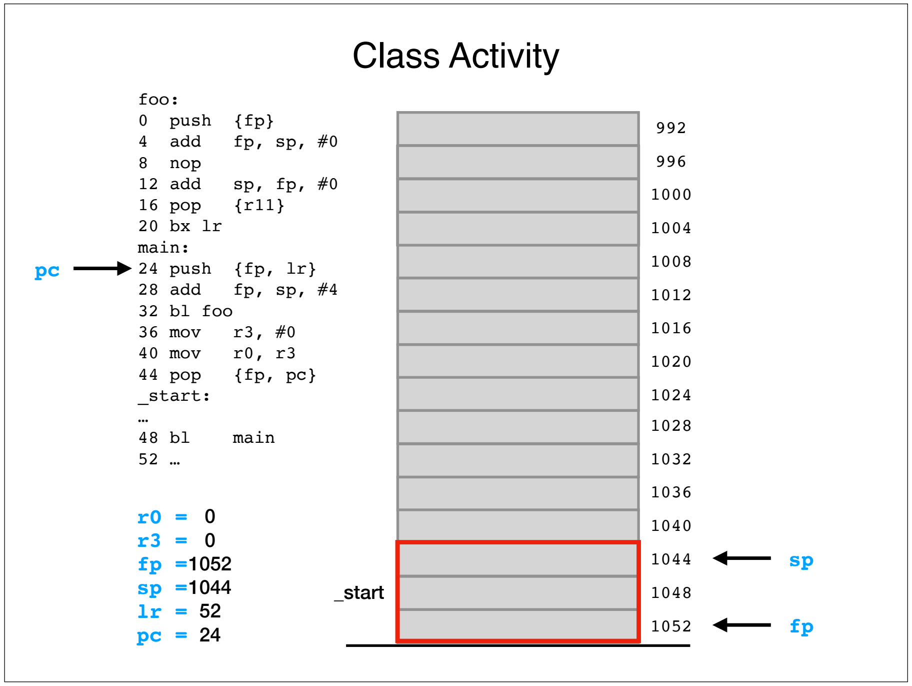
What does it do?

Caller vs callee



ARM Calling Convention

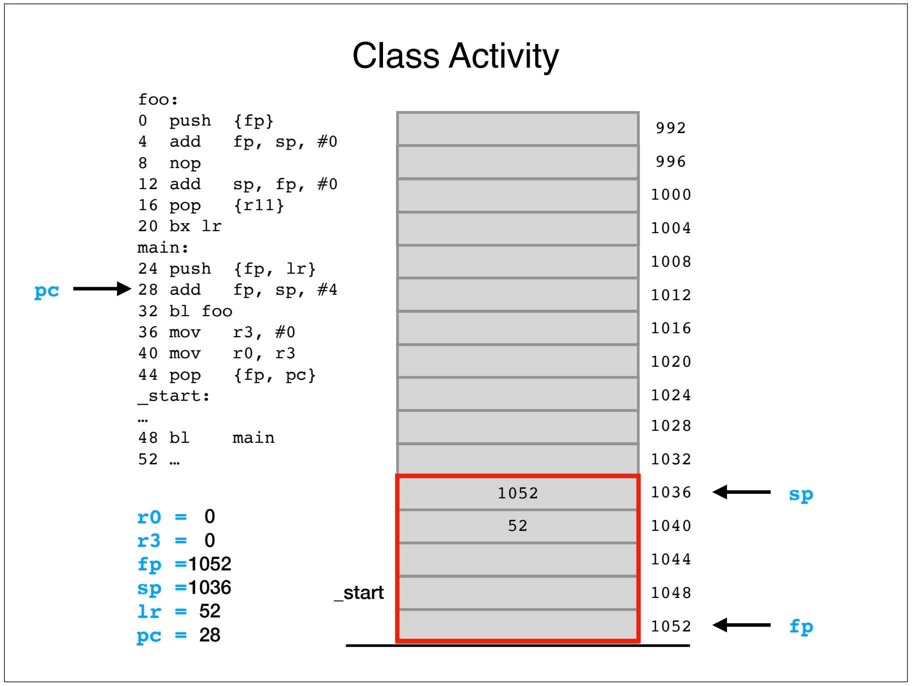




Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
pc → 24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

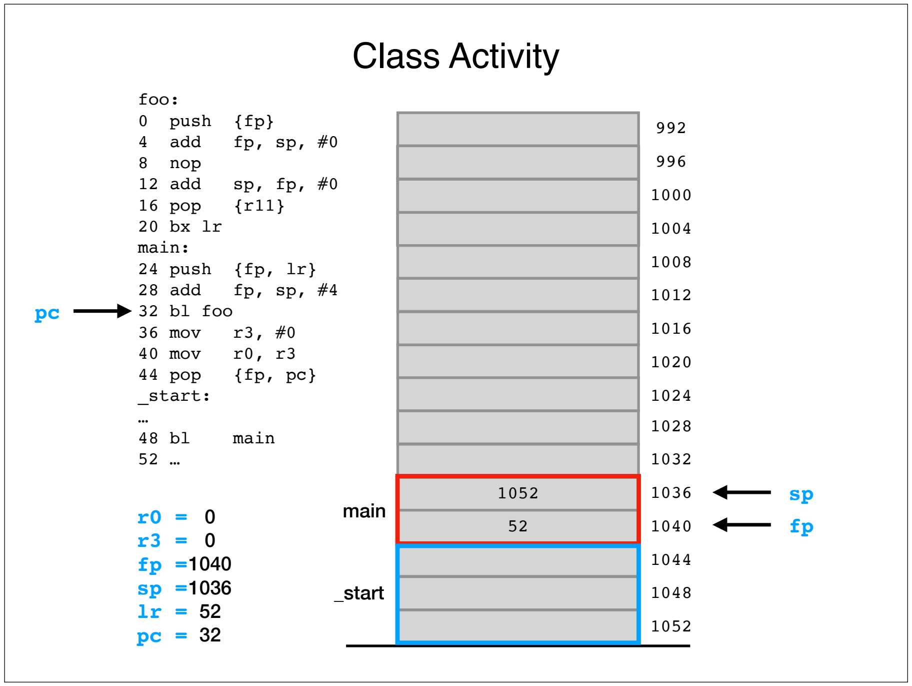
r0 = 0
r3 = 0
fp =1052
sp =1044
lr = 52
pc = 24



Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
pc → 28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

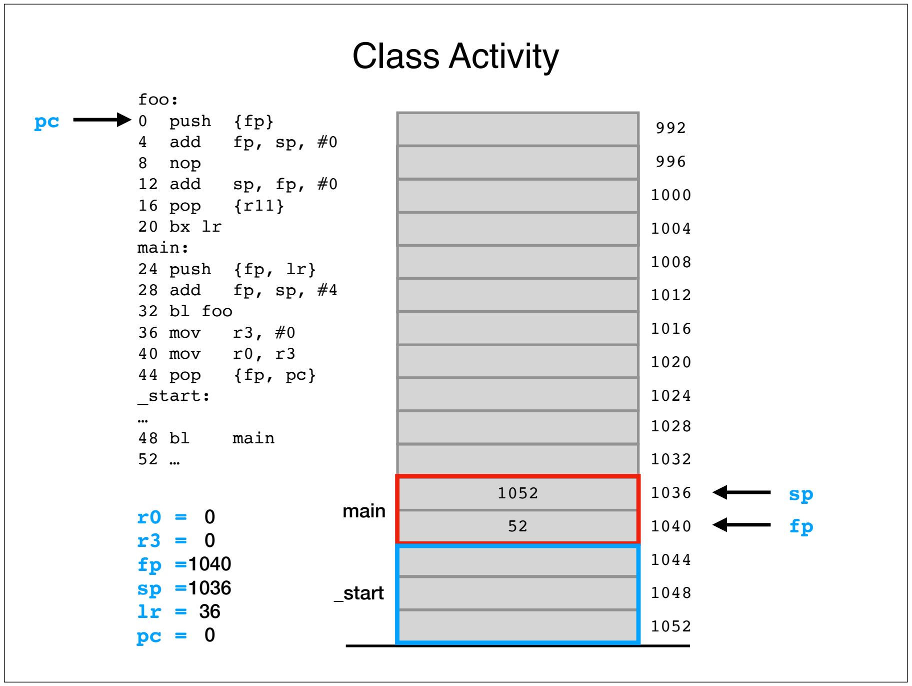
r0 = 0
r3 = 0
fp =1052
sp =1036
lr = 52
pc = 28



Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
pc → 32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 52
pc = 32



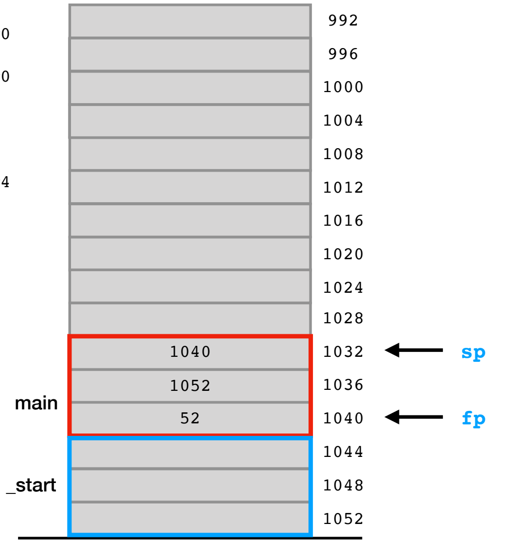
Class Activity

```
pc → foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 36
pc = 0

Class Activity

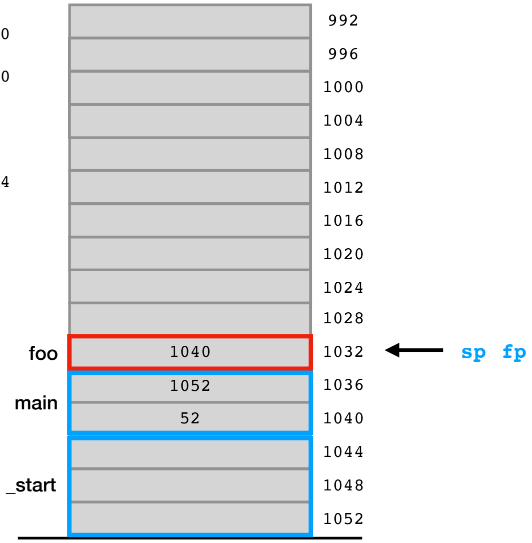
```
       foo:
       0  push  {fp}
pc →   4  add   fp, sp, #0
       8  nop
       12 add   sp, fp, #0
       16 pop   {r11}
       20 bx lr
       main:
       24 push  {fp, lr}
       28 add   fp, sp, #4
       32 bl foo
       36 mov   r3, #0
       40 mov   r0, r3
       44 pop   {fp, pc}
       _start:
       …
       48 bl    main
       52 …
```



r0 = 0
r3 = 0
fp =1040
sp =1032
lr = 36
pc = 4

Class Activity

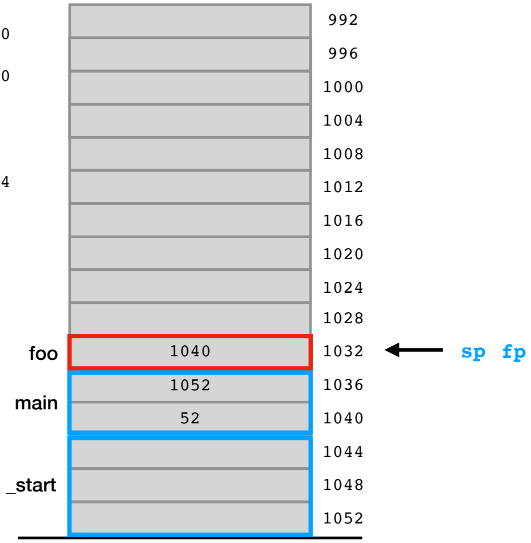
```
       foo:
       0  push  {fp}
       4  add   fp, sp, #0
pc →   8  nop
       12 add   sp, fp, #0
       16 pop   {r11}
       20 bx lr
       main:
       24 push  {fp, lr}
       28 add   fp, sp, #4
       32 bl foo
       36 mov   r3, #0
       40 mov   r0, r3
       44 pop   {fp, pc}
       _start:
       …
       48 bl    main
       52 …
```



r0 = 0
r3 = 0
fp =1032
sp =1032
lr = 36
pc = 8

Class Activity

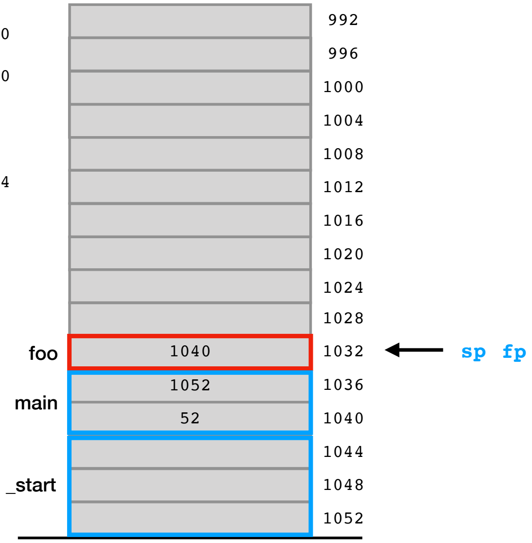
```
       foo:
       0  push  {fp}
       4  add   fp, sp, #0
       8  nop
pc →   12 add   sp, fp, #0
       16 pop   {r11}
       20 bx lr
       main:
       24 push  {fp, lr}
       28 add   fp, sp, #4
       32 bl foo
       36 mov   r3, #0
       40 mov   r0, r3
       44 pop   {fp, pc}
       _start:
       …
       48 bl    main
       52 …
```



r0 = 0
r3 = 0
fp =1032
sp =1032
lr = 36
pc = 12

Class Activity

```
       foo:
       0  push  {fp}
       4  add   fp, sp, #0
       8  nop
       12 add   sp, fp, #0
pc →   16 pop   {r11}
       20 bx lr
       main:
       24 push  {fp, lr}
       28 add   fp, sp, #4
       32 bl foo
       36 mov   r3, #0
       40 mov   r0, r3
       44 pop   {fp, pc}
       _start:
       …
       48 bl    main
       52 …
```

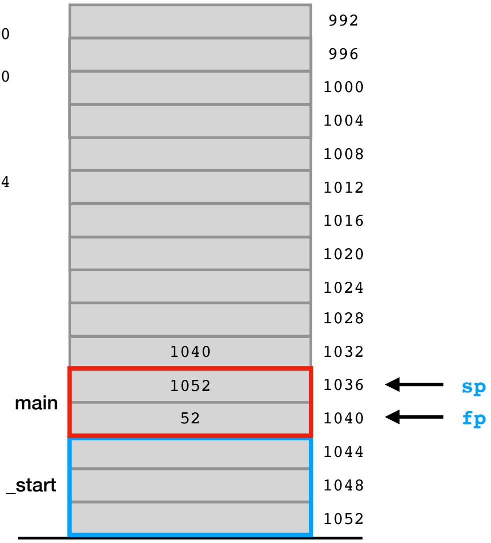


r0 = 0
r3 = 0
fp =1032
sp =1032
lr = 36
pc = 16

Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
pc → 20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 36
pc = 20

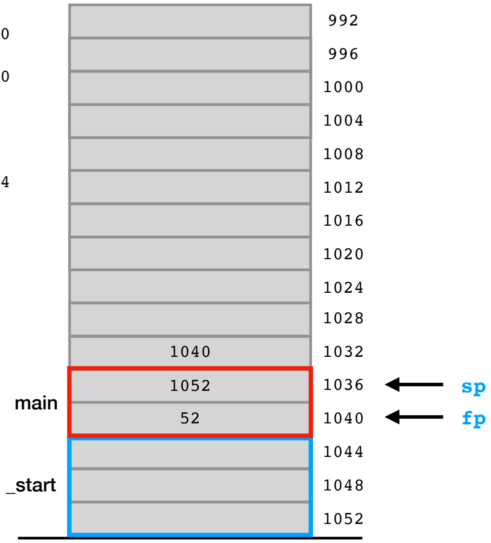


```
                992
                996
                1000
                1004
                1008
                1012
                1016
                1020
                1024
                1028
        1040    1032
main    1052    1036  ← sp
        52      1040  ← fp
                1044
_start          1048
                1052
```

Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
pc → 36 mov   r3, #0
     40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 36
pc = 36

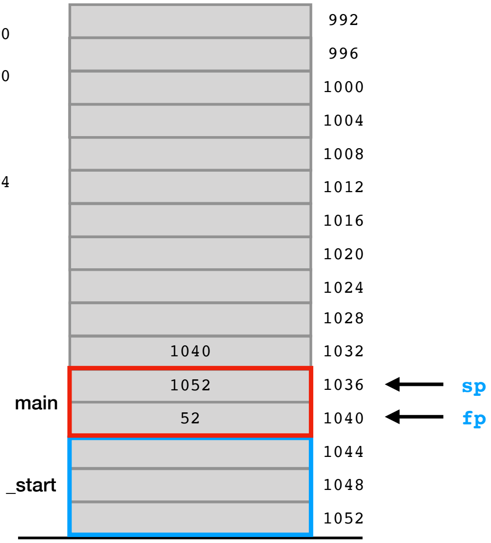


```
                992
                996
                1000
                1004
                1008
                1012
                1016
                1020
                1024
                1028
        1040    1032
main    1052    1036  ← sp
        52      1040  ← fp
                1044
_start          1048
                1052
```

Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
pc → 40 mov   r0, r3
     44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 36
pc = 40

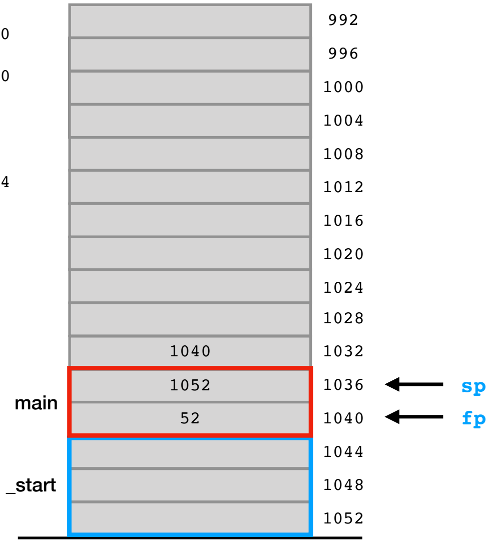


```
                992
                996
                1000
                1004
                1008
                1012
                1016
                1020
                1024
                1028
        1040    1032
main    1052    1036  ← sp
        52      1040  ← fp
                1044
_start          1048
                1052
```

Class Activity

```
     foo:
     0  push  {fp}
     4  add   fp, sp, #0
     8  nop
     12 add   sp, fp, #0
     16 pop   {r11}
     20 bx lr
     main:
     24 push  {fp, lr}
     28 add   fp, sp, #4
     32 bl foo
     36 mov   r3, #0
     40 mov   r0, r3
pc → 44 pop   {fp, pc}
     _start:
     …
     48 bl    main
     52 …
```

r0 = 0
r3 = 0
fp =1040
sp =1036
lr = 36
pc = 44



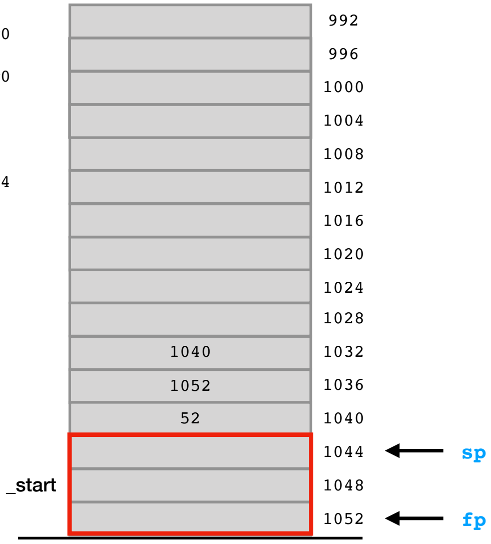
```
                992
                996
                1000
                1004
                1008
                1012
                1016
                1020
                1024
                1028
        1040    1032
main    1052    1036  ← sp
        52      1040  ← fp
                1044
_start          1048
                1052
```

Class Activity

Everything is **back to where it started** except pc, which is advanced to 52.

```
foo:
0  push  {fp}
4  add   fp, sp, #0
8  nop
12 add   sp, fp, #0
16 pop   {r11}
20 bx lr
main:
24 push  {fp, lr}
28 add   fp, sp, #4
32 bl foo
36 mov   r3, #0
40 mov   r0, r3
44 pop   {fp, pc}
_start:
…
48 bl    main
pc ⟶ 52 …
```

```
r0 =  0
r3 =  0
fp =1052
sp =1044
lr = 36
pc = 52
```



	992	
	996	
	1000	
	1004	
	1008	
	1012	
	1016	
	1020	
	1024	
	1028	
1040	1032	
1052	1036	
52	1040	
	1044	← sp
	1048	
	1052	← fp

Recap & Next Class

Today we learned:

How argument passing works

Next class:

globalthermonuclearwar

and other string vulnerabilities