

CS356: C Programming Language Review

W. Michael Petullo

University of Wisconsin–La Crosse

As of April 5, 2022





C: For loop

```
1 #include <stdio.h>

3 int main(void)
4 {
5     for (int i = 0; i < 10; i++) {
6         printf("%d\n", i);
7     }
8 }
```



C: For loop

Address	Machine Code	Assembly	Meaning
401126	55	push %rbp	Push FP
401127	48 89 e5	mov %rsp,%rbp	Set FP
40112a	48 83 ec 10	sub \$0x10,%rsp	Grow stack
40112e	c7 45 fc 00 00 00 00	movl \$0x0,-0x4(%rbp)	Initialize i
401135	eb 18	jmp 40114f <main+0x29>	Jump to cmpl
401137	8b 45 fc	mov -0x4(%rbp),%eax	Move i to eax
40113a	89 c6	mov %eax,%esi	2nd printf arg.
40113c	bf 10 20 40 00	mov \$0x402010,%edi	1st printf arg.
401141	b8 00 00 00 00	mov \$0x0,%eax	Clear ret. reg.
401146	e8 e5 fe ff ff	callq 401030 <printf@plt>	Call printf
40114b	83 45 fc 01	addl \$0x1,-0x4(%rbp)	Increment i
40114f	83 7d fc 09	cmpl \$0x9,-0x4(%rbp)	Compare i to 9
401153	7e e2	jle 401137 <main+0x11>	Jump if \leq
401155	b8 00 00 00 00	mov \$0x0,%eax	Set return value
40115a	c9	leaveq	Restore FP
40115b	c3	retq	Return



C: An undergrad's best friend

```
1 #include <stdio.h>

3 int main(void)
4 {
5     printf("%lx\n", *(long *) 0x0);
6 }
```

```
1 #include <stdio.h>

3 int main(void)
4 {
5     const char *str = "Hello , world!";

7     printf("%f %d %s\n", str, str, str);
8 }
```

```
1 #include <stdio.h>

3 int main(int argc, char *argv[])
4 {
5     for (int i = 0; i < argc; i++) {
6         printf("%s\n", argv[i]);
7     }
8 }
```

```
1 #include <stdio.h>

3 int main(void)
4 {
5     char c = 0x01;
6     short s = 0x01;
7     int i = 0x01;
8     long l = 0x01;
9     size_t z = 0x01;
10    char *str = "Hello ,_world!";
11    char array [] = "Hello ,_world!";

13    printf("%d _%d _%d _%d _%d _%d _%d\n" ,
14           sizeof c, sizeof s,
15           sizeof i, sizeof l,
16           sizeof z, sizeof str,
17           sizeof array);
18 }
```

C: Type games

```
1 #include <stdio.h>
2 #include <string.h>

4 struct foo {
5     int x;
6     int y;
7 };

9 int main(int c, char *a[])
10 {
11     size_t s;
12     struct foo f;

14     s = sizeof f > strlen(a[0]) ? strlen(a[0]) : sizeof f;
15     memcpy(&f, a[0], s);
16     printf("%d_%d\n", f.x, f.y);
17 }
```




```
1 #include <stdio.h>

3 int main(void)
4 {
5     short n = 0x1234;
6     printf("%x\n", n);
7     printf("%x%x\n",
8           ((unsigned char *) &n)[0],
9           ((unsigned char *) &n)[1]);
10 }
```

```
1 #include <stdio.h>

3 int main(void)
4 {
5     float f;
6     int i, count;
7     char s[BUFSIZ];

9     const char *in = "3.14_42_Hello ,_world!";

11    count = sscanf(in, "%f_%d_%s", &f, &i, s);

13    printf("%d_%f_%d_%s\n", count, f, i, s);
14 }
```



C: Pointers

```
1
2  int i = 42; printf("%d\n", i);
3
4  int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5
6  char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7
8  int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```



C: Pointers

```
1  /* Integer: */
2  int i = 42; printf("%d\n", i);
3
4  int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5
6  char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7
8  int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```



C: Pointers

```
1 /* Integer: */
2 int i = 42; printf("%d\n", i);
3 /* Pointer to an integer: */
4 int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5
6 char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7
8 int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```

C: Pointers

```
1  /* Integer: */
2  int i = 42; printf("%d\n", i);
3  /* Pointer to an integer: */
4  int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5  /* Pointer to a char: */
6  char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7
8  int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```

C: Pointers

```
1  /* Integer: */
2  int i = 42; printf("%d\n", i);
3  /* Pointer to an integer: */
4  int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5  /* Pointer to a char: */
6  char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7  /* Pointer to a pointer to an integer: */
8  int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```

C: Pointers

```
1 /* Integer: */
2 int i = 42; printf("%d\n", i);
3 /* Pointer to an integer: */
4 int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5 /* Pointer to a char: */
6 char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7 /* Pointer to a pointer to an integer: */
8 int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9 /* Function returning a pointer to void: */
10 void *fn(void);
11
12 void (*fn)(void);
13
14 int (*fn)(int);
```


C: Pointers

```
1 /* Integer: */
2 int i = 42; printf("%d\n", i);
3 /* Pointer to an integer: */
4 int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5 /* Pointer to a char: */
6 char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7 /* Pointer to a pointer to an integer: */
8 int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9 /* Function returning a pointer to void: */
10 void *fn(void);
11 /* Pointer to function returning void: */
12 void (*fn)(void);
13
14 int (*fn)(int);
```

C: Pointers

```
1 /* Integer: */
2 int i = 42; printf("%d\n", i);
3 /* Pointer to an integer: */
4 int *ip = &i; printf("%p\n", ip); printf("%d\n", *ip);
5 /* Pointer to a char: */
6 char *s = "abc"; printf("%p\n", s); printf("%c\n", *s);
   printf("%s\n", s);
7 /* Pointer to a pointer to an integer: */
8 int **ipp = &ip; printf("%p\n", ipp); printf("%p\n", *ipp);
   printf("%d\n", **ipp);
9 /* Function returning a pointer to void: */
10 void *fn(void);
11 /* Pointer to function returning void: */
12 void (*fn)(void);
13 /* Pointer to function taking and returning int: */
14 int (*fn)(int);
```

C: Function pointers

```
1 #include <stdio.h>

3 int inc(int i)
4 {
5     return ++i;
6 }

8 int dec(int i)
9 {
10    return --i;
11 }

13 int main(int argc, char *argv[])
14 {
15     int (*f)(int) = inc;
16     printf("%d\n", f(2));
17 }
```



C: Pointer arithmetic

```
1 #include <stdio.h>

3 int main(void)
4 {
5     int i[] = { 0, 1 };
6     int *ip = &i[0];

8     printf("%d\n", ip[0]);
9     printf("%d\n", ip[1]);

11    printf("%d\n", *ip);
12    printf("%d\n", *(ip + 1));

14    printf("%ld\n", (long) ip);
15    printf("%ld\n", (long) (ip + 1));
16 }
```

```
1 #include <sys/socket.h>
2 #include <netdb.h>
3 #include <unistd.h>

5 int main(void)
6 {
7     int fd; ssize_t size; char buf[1024];
8     struct addrinfo hints = { .ai_socktype = SOCK_STREAM };
9     struct addrinfo *info = NULL, *p;
10    getaddrinfo("towel.blinkenlights.nl", "23", &hints, &info);
11    for (p = info; p; p = p->ai_next) {
12        fd=socket(p->ai_family, p->ai_socktype, 0);
13        if (fd != -1) {
14            connect(fd, p->ai_addr, p->ai_addrlen);
15            break;
16        }
17    } ...
```



C: Socket (continued)

```
1  ...  
  
3  for (;;) {  
4      size = recv(fd, buf, 1024, 0);  
5      if (0 == size) {  
6          break;  
7      } else if (size > 0) {  
8          write(STDOUT_FILENO, buf, size);  
9      } else {  
10         break;  
11     }  
12 }  
13 }
```

```
1 #include <stdio.h>

3 int main(void)
4 {
5     char buf[16];

7     gets(buf);

9     printf("%s\n", buf);
10 }
```

```
1 #include <stdio.h>

3 void foo(void)
4 {
5     int b = 42;
6 }

8 void bar(void)
9 {
10    int a;
11    printf("%d\n", a);
12 }

14 int main(void)
15 {
16    foo();
17    bar();
18 }
```


Please refer to the course website for reading and homework assignments.

<http://www.flyn.org/courses/cs356-2022-fall/schedule>