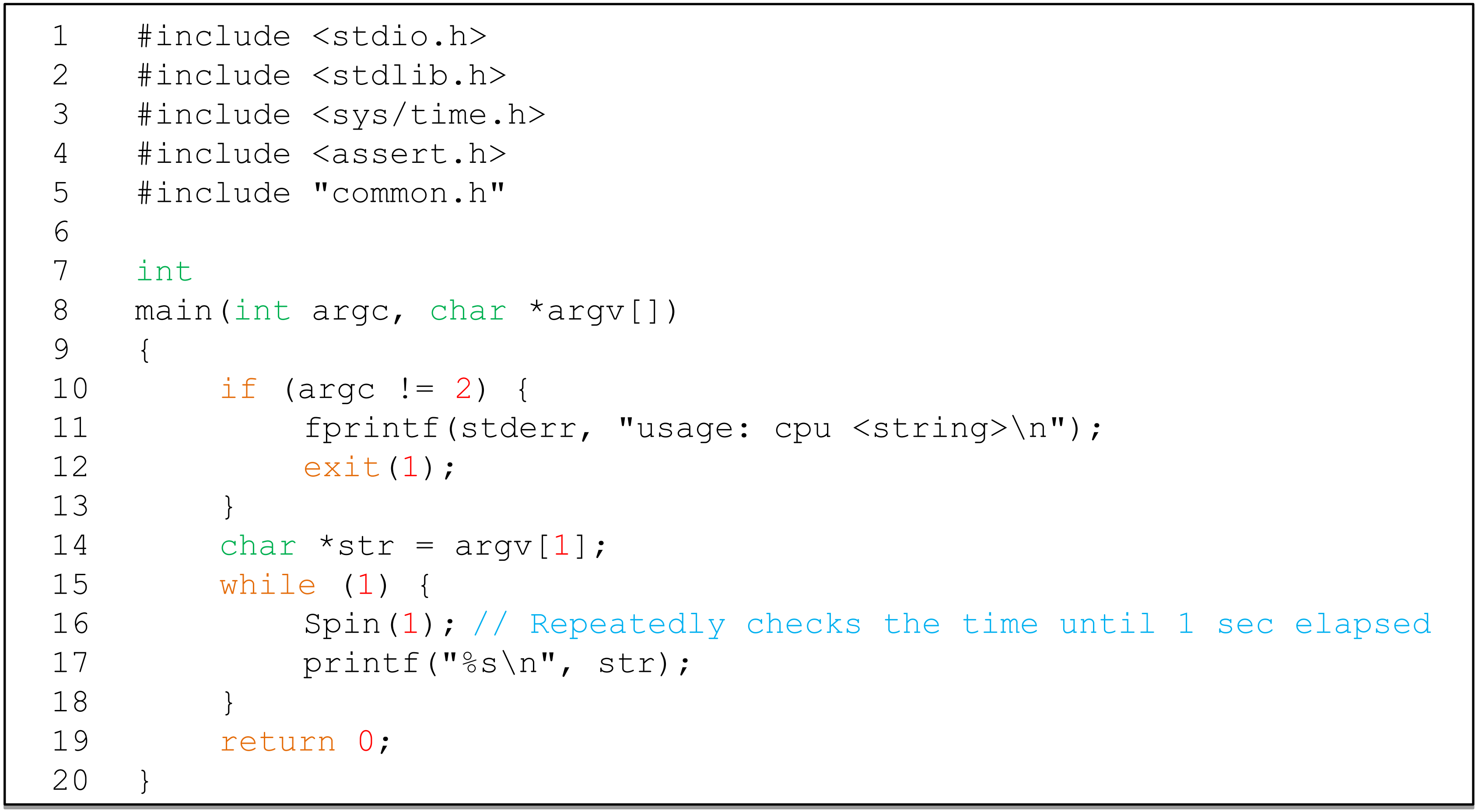
Operating Systems

Yannis Smaragdakis, U. Athens

What Does an OS Do?

- Makes sure that programs can share resources
  - CPU
  - memory
  - persistent storage
- Best form of sharing: virtual private resource
  - every program thinks it has its own copy of the resource
  - does this happen with CPU? Memory? Disk? How?

Demo: CPU



```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include "common.h"

int
main(int argc, char *argv[])
{
    if (argc != 2) {
        fprintf(stderr, "usage: cpu <string>\n");
        exit(1);
    }
    char *str = argv[1];
    while (1) {
        Spin(1); // Repeatedly checks the time until 1 sec elapsed
        printf("%s\n", str);
    }
    return 0;
}
```

Simple Example(`cpu.c`): Code That Loops and Prints

Result

```
prompt> gcc -o cpu cpu.c -Wall
prompt> ./cpu "A"
A
A
A
ˆC
prompt>
```

Run forever; Only by pressing “Control-c” can we halt the program

```
prompt> ./cpu A & ; ./cpu B & ; ./cpu C & ; ./cpu D &
[1] 7353
[2] 7354
[3] 7355
[4] 7356
A
B
D
C
A
B
D
C
A
C
B
D
...
```

Even though we have only one processor, all four of programs seem to be running at the same time!

Demo: Memory

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include "common.h"

int
main(int argc, char *argv[])
{
    int *p = malloc(sizeof(int));  // a1: allocate some memory
    assert(p != NULL);
    printf("(%d) address of p: %08x\n",
        getpid(), (unsigned) p);  // a2: print out address of memory
    *p = 0;  // a3: put zero into the first slot of the memory
    while (1) {
        Spin(1);
        *p = *p + 1;
        printf("(%d) p: %d\n", getpid(), *p); // a4
    }
    return 0;
}
```

Result

```
prompt> ./mem &; ./mem &
[1] 24113
[2] 24114
(24113) memory address of p: 00200000
(24114) memory address of p: 00200000
(24113) p: 1
(24114) p: 1
(24114) p: 2
(24113) p: 2
(24113) p: 3
(24114) p: 3
...
```

- Each program has its own copy of memory!

Coordination

- Virtualizing hides other programs
- But sharing is inevitable, often desirable
- An OS also needs to provide mechanisms for coordination
- Main coordination need: **concurrent** actions

Persistence

- Data need to be stored persistently
  - i.e., not be lost without power
- What models of persistence can you think of?
- We'll look into file systems
  - data structures, handling crashes, speed for specific operations, etc.

Background

- What kinds of hardware can you name?
  - single-core, multi-core, SMP, SMT, NUMA, heterogeneous MP, clusters, distributed systems, cloud
- What kinds of OSes?
- Refresh your binary arithmetic for this course