COMP 3511
Operating Systems

Lab 03

Outline

- Review Questions
- Process Control
- fork()
- Examples on fork()
- exec family: execute a program
- Project #1 description

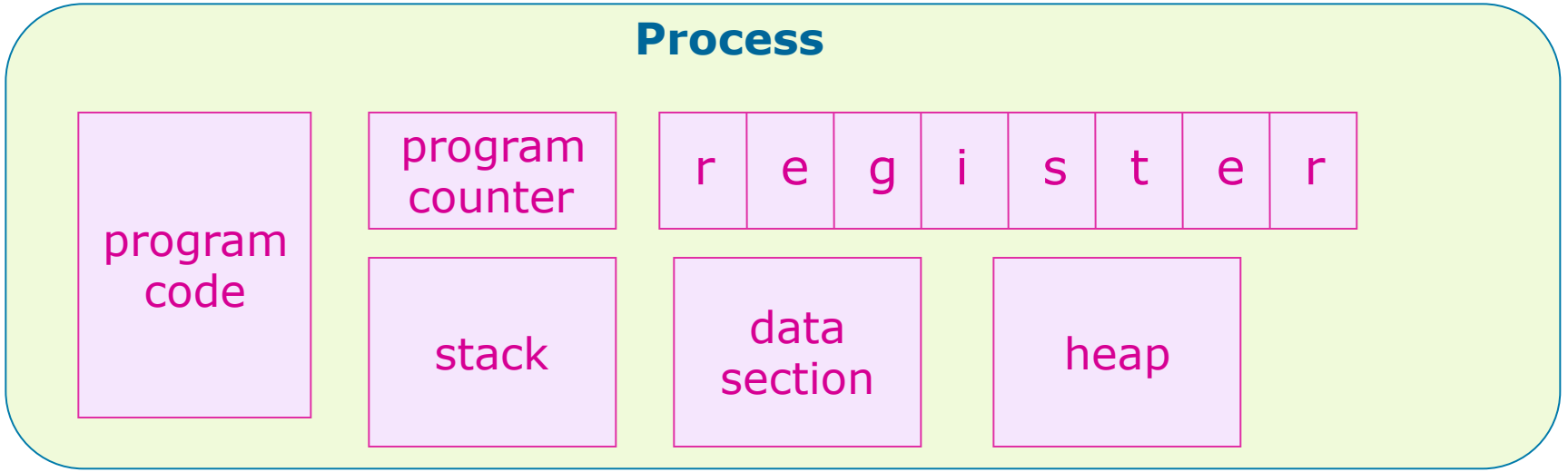
Q. 1

- What is the main difference between a **program** and a **process**?

A program is static (lines of codes stored)

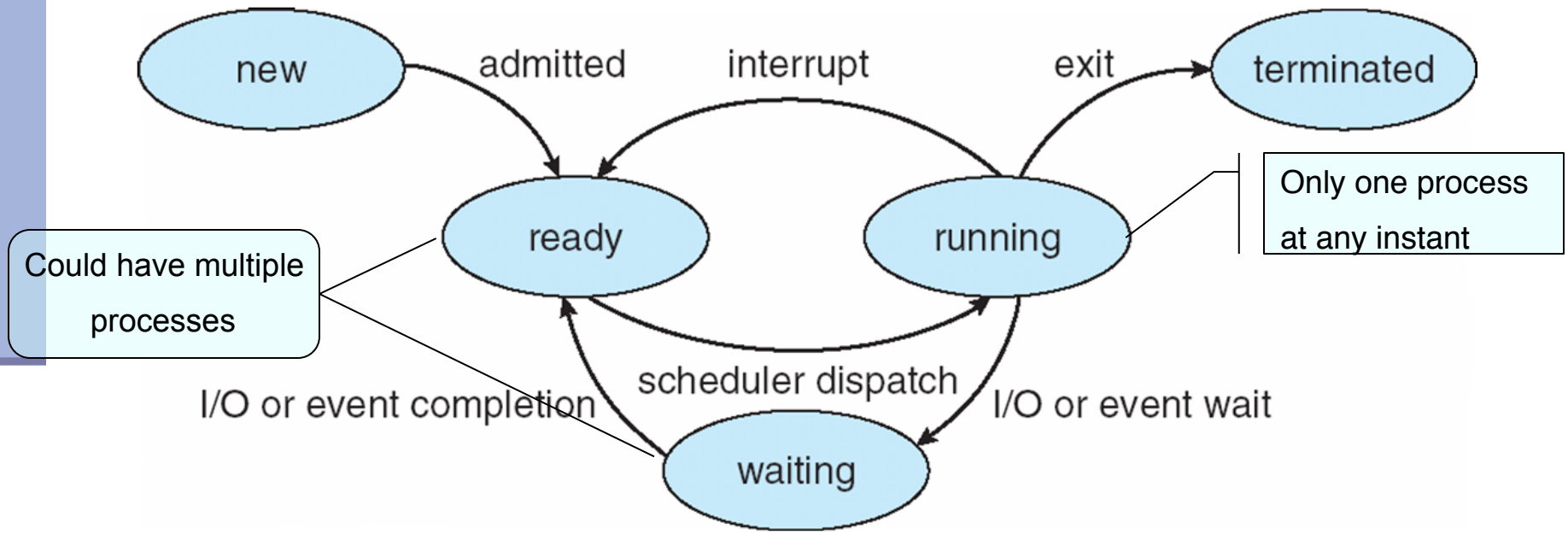
A process is active in execution,

which has a life cycle and can be in different states



Q. 2

- Briefly describe the process lifecycle with different states.



Q. 3

- When a process creates a child process, what are the four tasks that need to be done?

Creates a new PCB for the child process

allocate address space for the child process

copy data from the parent process

copy I/O state if any

Q. 4

- What are the two possibilities in terms of the address space of a newly created process?

A. The child process has a new program loaded into it

B. The child process has an exact copy of the address space of the parent process including program

C. The child process has an exact copy of the address space of the parent process including data

D. The child process has an exact copy of the address space of the parent process including program and data

Cont.

- What are the two possibilities in terms of the address space of a newly created process?

A. The child process has a new program loaded into it

B. The child process has an exact copy of the address space of the parent process including program

C. The child process has an exact copy of the address space of the parent process including data

D. The child process has an exact copy of the address space of the parent process including program and data

Q. 5

- Describe the differences between **short-term** and **long-term scheduling**.
  - Short-term scheduling (CPU scheduler) selects which process should be executed next and allocates CPU
  - Long-term scheduling (job scheduler) determines which processes should be brought into the ready queue

Cont.

- The primary difference is in the frequency of their execution
  - Short-term scheduling must select a new process quite often
  - Long-term scheduling is used much less often since it handles placing jobs in the system and may wait a while for a job to finish before it admits another one

Fork() example in C

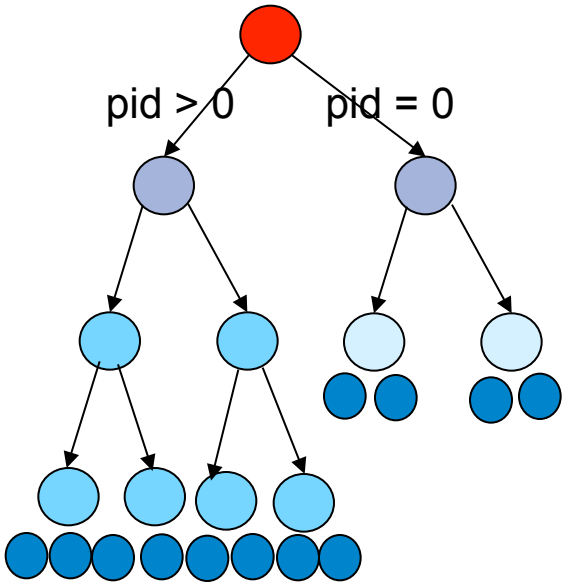
```
int main(void)
{
    pid_t pid = fork();
    if (pid == -1 ) {
                    /* when fork() return -1, an error occurred */
         fprintf(stderr, "Fork Failed");
         exit(EXIT_FAILURE);
    }
    else if (pid == 0) {
                    /* when fork() return 0, we are in the child
 process */
       printf("Hello from the child process!");
                    _exit(EXIT_SUCCESS);
               }
         else {
       /* when fork() return a positive integer, we are in the
 parent process */
                    /* the return value of the process id of the
 newly created child process */
                    int status;
       (void) waitpid(pid, &status, 0);
               }
                    return EXIT_SUCCESS;
```

Q. 6

- Consider the following code segment:
- pid_t pid;
- pid = fork();
- if (pid == 0) {
- fork();
- }
- if (pid > 0) {
- fork(); fork();
- }
- fork();
- Q: How many distinct child processes will be generated?

Q. 6

- Consider the following code segment:
- pid_t pid;
- pid = fork();
- if (pid == 0) {
- fork();
- }
- if (pid > 0) {
- fork(); fork();
- }
- fork();
- Q: How many distinct child processes will be generated?



Q. 7

- What are the benefits of process cooperation?

Information sharing

Modularity

Computation speed-up

Convenience

Q. 8

- Why is it simpler to handle independent processes than **cooperating processes**?
  - The execution and results of independent processes do not depend on one another, thus no consideration for any synchronization

Q. 9

- What are the two fundamental models for **Inter -Process Communication (IPC)**?

A. Shared address space

B. Shared memory

C. Code sharing

D. Message passing