1. What are the benefits of multi-threading?

- 响应能力：即使一个程序的部分阻塞或者执行冗长操作，其他线程仍然可以继续执行，响应用户。
- 资源共享：线程的代码段、数据段、动态分配空间等资源是默认共享的，无需程序员显式安排。
- 经济：进程的创建需要消耗更加昂贵的内存和资源分配，而线程由于共享一部分资源，创建和切换的代价更低。
- 可伸缩性：对于多处理器体系结构，多线程可以在多核处理器核上并行执行。

2. Which of the following components of program state are shared across threads in a multi-threaded process?

Answer: B&C

a. Register values

b. Heap memory

c. Global variables

d. Stack memory

2. Consider the following code segment:

```
pid_t pid;
pid = fork();
if (pid == 0) { /* child process */

fork();

thread create( . . .); }

fork();
```

a. How many unique processes are created?

不太确定要不要算上第一个父进程，如果不算，那么总共创建了5个新的进程，否则算上父进程就是6个，其中5个是新创建的。

b. How many unique threads are created?

总共会有8个线程，fork后的新进程只保留那个调用它的线程。

3. The program shown in the following figure uses Pthreads. What would be the output from the program at `LINE C` and `LINE P`?

At LINE C，the output would be 5.

At LINE P , the output would be 0. 这是因为子进程虽然在fork时复制了父进程的全局信息，但是在复制完后是独立的PCB，修改子进程的全局变量，不会影响父进程的。

```
#include <pthread.h>
#include <stdio.h>

int value = 0;
void *runner(void *param); /* the thread */

int main(int argc, char *argv[])
{
pid_t pid;
pthread_t tid;
pthread_attr_t attr;

   pid = fork();

   if (pid == 0) { /* child process */
      pthread_attr_init(&attr);
      pthread_create(&tid,&attr,runner,NULL);
      pthread_join(tid,NULL);
      printf("CHILD: value = %d",value); /* LINE C */
   }
   else if (pid > 0) { /* parent process */
      wait(NULL);
      printf("PARENT: value = %d",value); /* LINE P */
   }
}

void *runner(void *param) {
   value = 5;
   pthread_exit(0);
}
```

4. What are the differences between ordinary pipe and named pipe?

对于普通管道，仅允许同一主机上具有父子关系的进程使用，仅允许单向通信，通信结束时管道关闭。

而命名管道在文件系统中有文件名，因此通信进程双方父子关系不是必须的（但仍需在同一主机上），支持双向通信（但仍是半双工），支持多个进程使用（可以有多个进程读/写），关闭管道需要明确的关闭指令。

5. List all the requirements of the entry and exit implementation when solving the critical-section problem. Analyze whether strict alternation satisfies all the requirements.

Requirement #1: Mutual Exclusion. No two processes could be simultaneously inside their critical sections. 同一时刻不允许两个进程都在critical sections。

Requirement #2. Each process is executing at a nonzero speed, but no assumptions should be made about the relative speed of the processes and the number of CPUs. 解决方法不能依赖于对进程临界区运行时间的推测，或者是CPU的数量。

Requirement #3: progress.No process running outside its critical section should block other processes. 运行在非临界区的进程不允许阻塞其他进程。

Requirement #4: Bounded waiting.No process would have to wait forever in order to enter its critical section. 进程不能无限等待进入临界区。

strict alternation 违反了Requirement #3

6.What is deadlock? List the four requirements of deadlock.

一组进程无限等待一个事件，但这个事件只能由这些等待的进程之一产生，导致了谁都不能执行，这就是死锁。

死锁需要满足4个条件：互斥、每个进程至少持有一个资源且等待另一个其他进程持有的资源、非抢占只能主动释放、循环等待。

7. What is semaphore? Explain the functionalities of semaphore in process synchronization.

信号量是一种数据类型（可以是int型），它除了初始化外仅允许通过两个标准原子操作。这两个原子的操作是down() 和 up()。操作系统拥有一个初始信号量的值，每当一个进程使用资源，那么操作系统就会把信号量减一，并把资源分配出去。如果信号量为 0，那么后来的进程就得等待。 信号量可以在软件层面上实现类似互斥锁的功能，更加方便程序员的使用，但可能会有死锁问题。

8. Please use semaphore to provide a deadlock-free solution to address the dining philosopher problem.

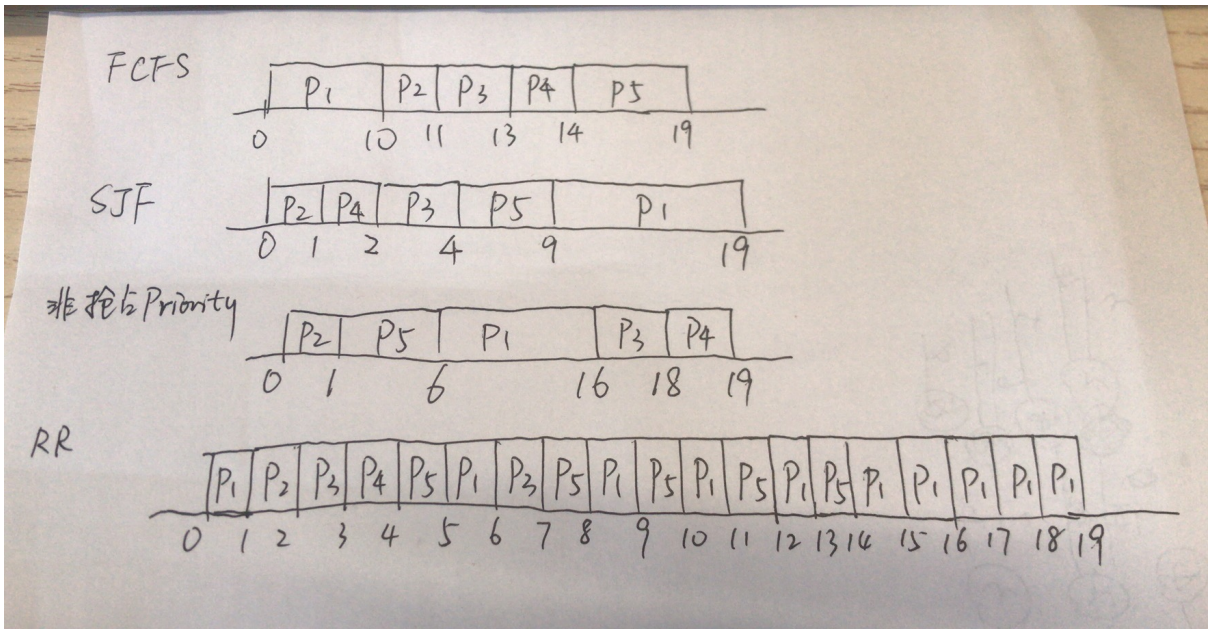
一种方法没有死锁，但可能会造成饥饿，仍然将筷子作为semaphore。首先哲学家拿起左边的筷子，如果此时右边的筷子被占用，就放下左边的筷子等待。否则拿起右边的筷子进食，吃完后放下两边的筷子并通知两侧的哲学家。

9. Consider the following set of processes, with the length of the CPU burst time given in milliseconds:

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

- **a) Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF (nonpreemptive), nonpreemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1).**



- **b) What is the turnaround time of each process for each of the scheduling algorithms in part a?**
  fsfs: p1=10; p2=11; p3=13; p4=14; p5=19

  sjf: p1=19; p2=1; p3=4; p4=2; p5=9

  nonpreemptive priority: p1=16; p2=1; p3=18; p4=19; p5=6

  RR: p1=19; p2=2; p3=7; p4=4; p5=14

- **c) What is the waiting time of each process for each of these scheduling algorithms?**
  fsfs: p1=0; p2=10; p3=11; p4=13; p5=14

  sjf: p1=9; p2=0; p3=2; p4=1; p5=4

nonpreemptive priority: p1=6; p2=0; p3=16; p4=18; p5=1

RR: p1=9; p2=1; p3=5; p4=3; p5=9

- **d) Which of the algorithms results in the minimum average waiting time (over all processes)?**

  平均等待时间SJF最短

10.Which of the following scheduling algorithms could result in starvation?

Answer: B&C

a) First-come, first-served

b) Shortest job first

c) Round robin

d) Priority

11.Give an example to illustrate under what circumstances rate- monotonic scheduling is inferior to earliest-deadline-first scheduling in meeting the deadlines associated with processes?

考虑下图这种情况，p1的周期为50，处理时间为25；p2周期为75，处理时间为30。如果采用单调速率调度算法，那么p1和p2的优先级是固定的，p1周期短，优先级更高。当p2执行到50时，p1可以抢占，导致p2的完成时间在80，已经超过了DDL。而对于EDF调度算法，优先级是动态的，则不会出现这种情况，起码P1、P2都能在DDL钱完成工作。