2024 年秋季操作系统第一次作业

Part1. 进程与线程

1. 支持多道程序设计的操作系统在运行过程中，不断地选择新进程运行来实现 CPU 的共享，但其中（）不是引起操作系统选择新进程的直接原因。

A.运行进程的时间片用完

B.运行进程出错

C.运行进程要等待某一事件发生

D.有新进程被创建进入就绪态

2. 若一个进程实体由 PCB、共享正文段，数据堆段和数据栈段组成，请指出下列 C 语言程序中的内容及相关数据结构各位于哪一段中。

I. 全局赋值变量（）

II. 未赋值的局部变量（）

III.函数调用实参传递值（）

IV. 用 malloc()要求动态分配的存储区（）

V. 常量值（如 1995、"string"）（）

VI. 进程的优先级（）

A.PCB

B.正文段

C.堆段

D.栈段

3. 以下可能导致一个进程从运行态变为就绪态的事件是（）。

A. 一次 1/O 操作结束

C.运行进程结束

B. 运行进程需做 I/O 操作

D. 出现了比现在进程优先级更高的进程

4. 下列选项中，降低进程优先级的合理时机是（）。

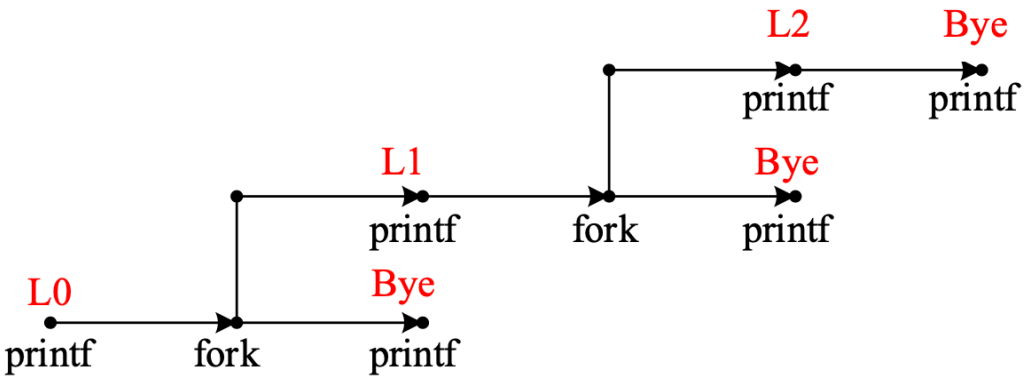
A. 进程时间片用完

B. 进程刚完成 I/O 操作，进入就绪队列

C. 进程长期处于就绪队列

D. 进程从就绪态转为运行态

5. 根据进程图，以下哪个输出是不正确的（）



A. L0, Bye, L1, Bye, L2, Bye
B. L0, L1, Bye, L2, Bye, Bye
C. L0, Bye, L1, L2, Bye, Bye
D. L0, Bye, L1, Bye, Bye, L2

6. 下列关于线程的描述中，错误的是()。

A. 内核级线程的调度由操作系统完成

B. 用户级线程间的切换比内核级线程间的切换效率高

C. 用户级线程可以在不支持内核级线程的操作系统上实现

D. 操作系统为每个用户级线程建立一个线程控制块

7. 关于线程的以下描述中，哪一个是正确的（）

A. 线程共享进程的地址空间和资源

B. 每个线程用有独立的进程上下文

C. 线程的切换比进程的切换代价更大

D. 线程是独立于进程的执行单元

1. 分析程序 homework_wait.c，回答下列问题：

```
1.  /* homework_wait.c */
2.  void homework_wait() {
3.      pid_t pid[N];
4.      int i, child_status;
5.      for (i = 0; i < N; i++) {
6.          if ((pid[i] = fork()) == 0) {
7.              exit(100+i); /* Child */
8.          }
9.      }
10.     printf("hello!\n");
11.     for (i = 0; i < N; i++) { /* Parent */
12.         pid_t wpid = wait(&child_status);
13.         if (WIFEXITED(child_status))
14.             printf("Child %d terminated with exit status %d\n",
15.                     wpid, WEXITSTATUS(child_status));
16.         else
17.             printf("Child %d terminate abnormally\n", wpid);
18.     }
19. }
```

1) 注释掉第 7 行代码后，程序执行到第 10 行，输出多少个“hello!”（用一个 N 的函数给出答案）？
2) N=2 时，程序正常运行两次，得到的结果是否相同？若不同，请解释原因；
3) 修改程序，使得子进程能够按照其创建的顺序退出。

2. 在以下程序中，会创建一个新进程，然后父进程会等待该进程终止。之后，父进程会创建一个新进程并重复整个过程。

修改此代码以创建两个不同的进程（并行执行），父进程会等待两个进程的终

止。

```
#include ‹sys/types.h›
#include ‹stdio.h›

int main (int argc, char *argv[])
{
    pid_t pid;
    int status;
    pid = fork ();
    if (pid!= 0)
        while (pid != wait (&status));
    else {
        sleep (5);
        exit (5) ;
    }
    pid = fork ();
    if (pid != 0)
        while (pid != wait (&status) );
    else {
        sleep (1);
        exit (1);
    }
}
```

3. 阅读以下代码

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

void *mythread(void *arg) {
    printf("%s\n", (char *) arg);
    return NULL;
}

int main(int argc, char *argv[]) {
    pthread_t p1, p2, p3;
    int rc;
    printf("main: begin\n");
    rc = pthread_create(&p1, NULL, mythread, "I "); assert(rc == 0);
    rc = pthread_create(&p2, NULL, mythread, "LIKE "); assert(rc == 0);
    rc = pthread_create(&p3, NULL, mythread, "OS "); assert(rc == 0);
    // join waits for the threads to finish
    rc = pthread_join(p1, NULL); assert(rc == 0);
    rc = pthread_join(p2, NULL); assert(rc == 0);
    rc = pthread_join(p3, NULL); assert(rc == 0);
    printf("\nmain: end\n");
    return 0;
}
```

(1) 以上程序运行的输出结果共有几种?
(2) 如何修改上述程序，让其按照"main:start"、"I LIKE OS"、"main:end"的顺序输出。

Part2. 并发与同步

1. 对于两个并发进程，设互斥信号量为 mutex（初值为 1），若 mutex=-1，则（ ）。

A. 表示没有进程进入临界区

B. 表示有一个进程进入临界区

C. 表示有两个进程进入临界区

D. 表示有一个进程进入临界区，另一个进程在等待进入

2. 有两个优先级相同的并发程序 P1 和 P2，他们的执行过程如下所示。假设当前信号量 s1=0，s2=0。当前的 z=2，进程运行结束之后，x,y,z 的值分别为（ ）。

```c
P1: y=1;                    P2: x=1;
    y=y+2;                      x=x+1;
    z=y+1;                      P(s1);
    V(s1);                      x=x+y;
    P(s2);                      z=x+z;
    y=z+y;                      v(s2);
```

A. 5, 9, 9

B. 5, 9, 4

C. 5, 12, 9

D. 5, 12, 4

3. 下列准则中，实现临界区互斥机制必须遵循的是（ ）。

I. 两个进程不能同时进入临界区

II. 允许进程访问空闲的临界资源

III. 进程等待进入临界区的时间是有限的

IV. 不能进入临界区的执行态进程立即放弃 CPU

A. 仅 I、IV

B. 仅 II、III

C. 仅 I、II、III

D. 仅 I、III、IV

4. 设与某资源关联的信号量初值为 3，当前值为 1．若 M 表示该资源的可用个数，N 表示等待该资源的进程数，则 M,N 分别是（ ）。

A. 0,1

B. 1,0

c. 1,2

D. 2,0

5. 有三个进程共享同一程序段，而每次只允许两个进程进入该程序段，若用 PV 操作同步机制，则信号量 S 的取值范围是（ ）。

A. 2,1,0,-1

B. 3,2,1,0

C. 2,1,0,-1,-2

D. 1,0,-1,-2

6. 若一个信号量的初值为 3，经过多次 PV 操作后当前值为-1，这表示等待进入临界区的进程数是（ ）。

A. 1

B. 2

C. 3

D. 4

7. 信号作为一般进程间通信机制的实用性是有限的，因为 _____

A. 它们不能在进程之间工作

B. 它们是由用户生成的

C. 它们不能直接携带信息

D. 以上都不是

1. 请说明如果 P()的信号量操作没有原子执行，那么可能会违反互斥。

2. 说明如何使用二进制信号量（信号量取值为 0 和 1）来实现 n 个进程之间的互斥？

3. 假设你有一个只有二进制信号量的操作系统，但你希望使用计数信号量。请你演示如何使用二进制信号量实现计数信号量的 P_new()和 V_new()操作。提示：可以使用两个二进制信号量，一个用来实现计数互斥，另一个用来实现阻塞。