1. **Here are one keeper(保管员) and two groups of students. The keeper is responsible for managing pencils and pieces of paper. Each student in group A owns some pieces of paper, while every student in group B possesses pencils; for a student, holding pencil and paper enables him to write a letter, so the student in group A tries to gain a pencil, and the student in group B attempts to acquire a piece of paper.**

**There is also a little box that can contain only one pencil or one piece of paper.**

**The box is initially empty, and the keeper arbitrarily put a pencil or a piece of paper into the box, then one student is permitted to fetch item he/she needs from the box.**

**Once the student takes away a pencil or a piece of paper from the box, the keeper is allowed to supplement a new pencil or a piece of paper to the empty box. The keeper knows what’s in the box and notifies who need it to take it. At one time, only one person is permitted to operate on the box.**

**Design semaphore-based processes to describe the behaviors of the keeper, the students in group A and B.**

**It is required that**

**i) definitions and initial values of the semaphores should be given, and**

**ii) the structures of processes for the keeper and two groups of students should be presented.**

**Answer:**

**假设：keeper知道放入box中的是纸还是笔，据此可以分别与groupA、groupB中的学生进行同步协调**

**第1部分：信号量定义**

**Counting semaphore empty = 1; —代表box容量、有无纸笔**

**Binary semaphore Sa = 0; — keeper—group A间同步，代替生产者-消费者问题中的full；**

**Binary semaphore Sb = 0; — keeper—group B间的同步**

**Binary semaphore mutex = 1; — 用于控制keeper、group A、group B对box的互斥操作**

**第2部分：keeper、group A、group B的业务过程定义，Keeper为生产者，group A、group B中的学生为consumer**

**Keeper:**

**wait(empty);**

**wait(mutex);**

**put a material, i.e. paper or pencil, into the box;**

**if 放入paper**

**then signal(Sb)**

**else signal(Sa);**

**signal(mutex);**

**Student in group A:**

**wait(Sa);**

**wait(mutex);**

**fetch the pencil from the box;**

**signal(mutex);**

**signal(empty);**

**write a letter;**

**Student in group B:**

**wait(Sb);**

**wait(mutex);**

**fetch the paper from the box;**

**signal(mutex);**

**signal(empty);**

**write a letter;**

1. **(20 points)**

**Consider the following snapshot of a system**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Allocation | | | Max | | | Need | | | Available | | |
| R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 |
| P1 | 1 | 0 | 0 | 3 | 2 | 2 |  |  |  | 2 | 1 | 2 |
| P2 | 4 | 1 | 1 | 6 | 1 | 3 |  |  |  |  |  |  |
| P3 | 2 | 1 | 1 | 3 | 1 | 4 |  |  |  |  |  |  |
| P4 | 0 | 0 | 2 | 4 | 2 | 2 |  |  |  |  |  |  |

**(1) Fill in the contents of the matrix Need for each process in the space above.**

**(2) Is the system in a safe state? If it is safe, give the safe sequence.**

**(3) If both P1 and P2 make resource requests of <1, 0, 1>, how should we grant the requests while keeping the system in a safe state?**

**Answer:**

**(1)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Allocation | | | Max | | | Need | | | Available | | |
| R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 |
| P1 | 1 | 0 | 0 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 |
| P2 | 4 | 1 | 1 | 6 | 1 | 3 | 2 | 0 | 2 |  |  |  |
| P3 | 2 | 1 | 1 | 3 | 1 | 4 | 1 | 0 | 3 |  |  |  |
| P4 | 0 | 0 | 2 | 4 | 2 | 2 | 4 | 2 | 0 |  |  |  |

**(2) It’s safe.**

**One safe sequence: P2, P1, P3, P4(只要以P2开头即可)**

**(3) Only grant the request of P2 and refuse the request of P1. If the request of P1 is granted, then no process will have enough resources to execute, which will cause deadlock.**

**必须答出“拒绝P1请求，接受P2请求”这层意思。**

**相应的要有拒绝P1的理由，接受P2的理由。**