# COMP3500: The Banker’s Algorithm Part 2

**Exercise 1:** Please discuss with your group members tostudy how does the following safety algorithm check if a system is in a safe state. What is Work? Why *Work* = *Work* + *Allocationi* in step 3?

1. Let *Work* and *Finish* be vectors of length *m* and *n*, respectively. Initialize:

*Work* = *Available*

*Finish* [*i*] = *false* for *i* = 0, 1, …, *n-* 1

2. Find an *i* such that both:

(a) *Finish* [*i*] = *false*

(b) *Needi* <= *Work*

If no such *i* exists, go to step 4

*3. Work* = *Work* + *Allocationi*

*Finish*[*i*] = *true*  
 go to step 2

4. If *Finish* [*i*] == *true* for all *i*, then the system is in a safe state

**Exercise 2: Please complete the** Resource-Request Algorithm for Process *Pi*.

Request\_i = request vector for process Pi. If Request\_i [j] = k then process Pi wants k instances of resource type Rj

1. If Requesti \_\_\_\_\_\_\_\_\_\_ Needi go to step 2. Otherwise, raise error condition, since process has exceeded its maximum claim

2. If Requesti \_\_\_\_\_\_\_\_\_\_ Available, go to step 3. Otherwise Pi must wait, since resources are not available

3. Pretend to allocate requested resources to Pi by modifying the state as follows:

Available = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

Allocationi = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

Needi = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

**Exercise 3:** How to detect deadlocks in a system where there is a single instance per resource type?

**Exercise 4:** Please draw a corresponding wait-for graph. **Hint:** You need to remove all the resources from the graph.