ISE 302–Operating Systems
2018–2019, Homework 3

Introduction

You will develop a simulation for the Banker's algorithm **("one type of resource" version)**. It will evaluate given state of the processes and resources to decide whether it is safe or unsafe. It will also print the correct execution order for safe states and evaluate the new request given as command line parameters.

Input

Program input is a text file that represents "Max_Request" and "Has" vectors as well as number of resources. First line of the file is an integer value which is the total amount of resources. After that, each line in the document is the resource values of a process. First value is for the allocated request ("Has") and the second one is for the maximum request ("Max_Request"). These values are separated by space characters.

A sample input is provided. There are 3 processes in this input. However, another input with **different number of processes** will be used to evaluate your programs. There is no limitation on the number of processes.

Program Flow

Your program should be run from the command line with the following format:

./program input.txt new_request_process_id new_request_resource

Program should first read the input file and print either SAFE or UNSAFE. If the state is unsafe, then it should terminate. You can assume that the input file will be in the same folder, but the name of the file might be different.

If the state is safe, the program should print the correct execution order (before evaluating the new request). Afterwards the request given as command line arguments (**new_request_process_id** argument determines which process does the request and **new_request_resource** argument shows how many resources the process requests) will be tested.

Correct execution order is the order of process completion where no deadlock occur. For the sample input, if we allocate remaining 2 resources to the process 1 it would complete and release its resources. Using these resources, either process 0 or 2 can be completed. That's why, execution orders 1->0->2 and 1->2->0 are both correct.

Output

A possible program output for the sample input is given below.

```
>./program input.txt 1 2
Input state is SAFE.
A correct process execution order is 1->0->2.
The new request CAN be granted.

>./program input.txt 2 1
Input state is SAFE.
A correct process execution order is 1->0->2.
The new request CANNOT be granted.
```

Notes

- You will not implement any actual processes or threads for this homework. **It is pure simulation.**
- Course slides contain both "one type of resource" and "multiple types of resources" versions of the algorithm. **Do not implement the "multiple types of resources" version.**
- **Testing a request does not actually update the state**. It only test whether a request can be granted, it does not actually allocate any resources.
- You should use **zero-based** numbering on process ids.
- You will lose points if your program does not work for arbitrary number of processes.
- Include development environment information, compilation and running commands as a comment in your code.

EXAMPLE:
```
// Development environment: Lubuntu 16.04 or ITU SSH servers
// To compile: g++ -c Homework.cpp –pthread
// To run: ./a.out input.txt
```

Appendix

You may make use of the following code example for File I/O operations in C and C++.

- http://rosettacode.org/wiki/Read_a_file_line_by_line