**Banker’s Algorithm**

**刘宇 学号:2014213404 班号:2014215119**

In the project, I mainly achieve three tasks, multithreading, preventing race conditions, and deadlock avoidance.

Multithreading is easy to finish. I use following functions to achieve it.

I create a thread array used to hold different threads’ id. When I want to use these threads, I will use loop.

pthread\_t tids[NUMBER\_OF\_CUSTOMERS];

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

for(i=0;i< NUMBER\_OF\_CUSTOMERS;i++)

{

tids[i] = tid;

pthread\_create(&tid, &attr, req, i);

pthread\_join(tids[i], NULL);

}

And the number of threads is same as the number of Customers.

Preventing race condition can be achieved by using mutex lock.

pthread\_mutex\_init(&mutex, NULL);

pthread\_mutex\_lock(&mutex);

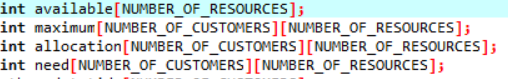
pthread\_mutex\_unlock(&mutex);

I use banker’s algorithm to achieve deadlock avoidance.

Firstly, I decide that the number of customers is five which means that I will create five threads and the number of types of resources is three.



What’s more, I define some array in order to store some information.



available is used to store the available amount of each resource.

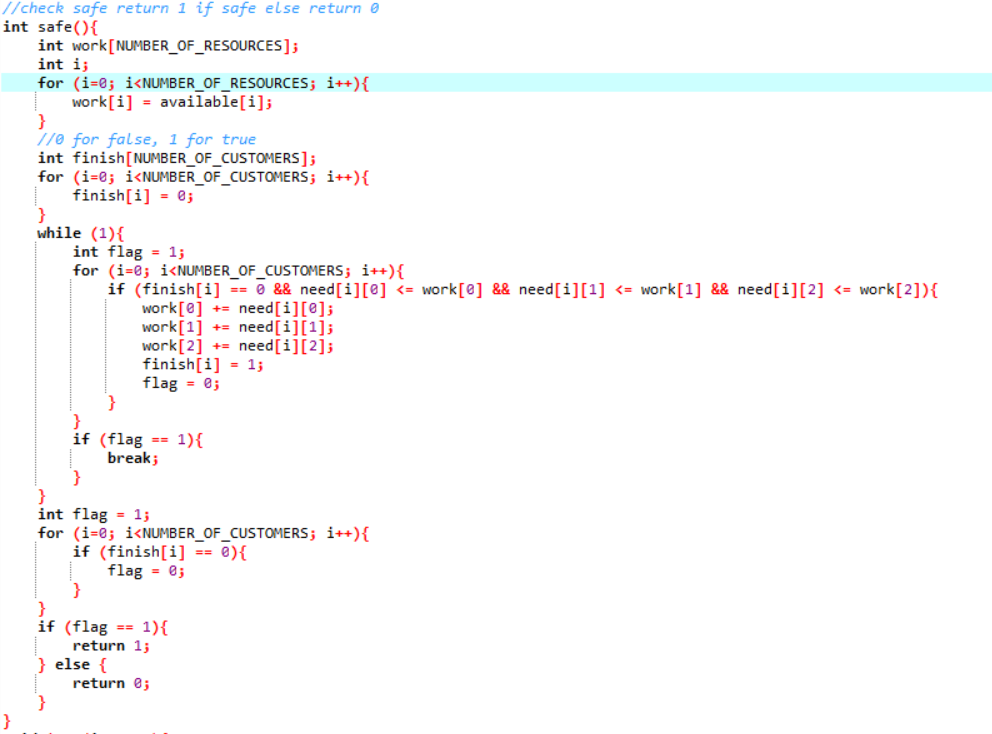
maximum is used to store the maximum demand of each customer.

allocation is used to store the amount currently allocated to each customer

need is used to store the remaining need of each customer

In the Banker’s algorithm, there are two main algorithms: **safety algorithm** and **resource-request algorithm**.

The safety algorithm is used to determine whether or not a system is in a safe state. If there is a sequence of processes such that every process can get enough resources, the system is a safe state. What’s more, the sequence of processes may not different if the system is a safe state.



If the system is safe, return 1; otherwise, return 0;

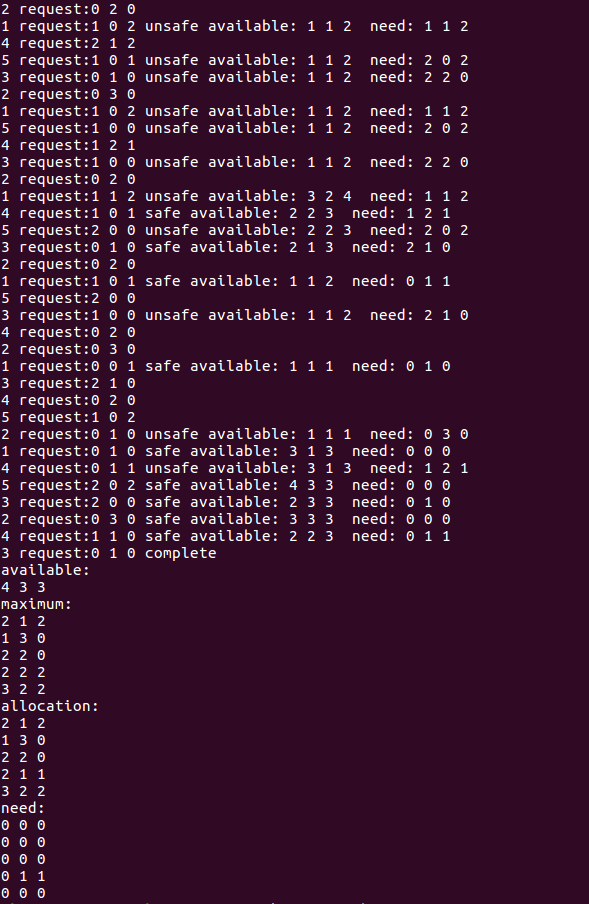
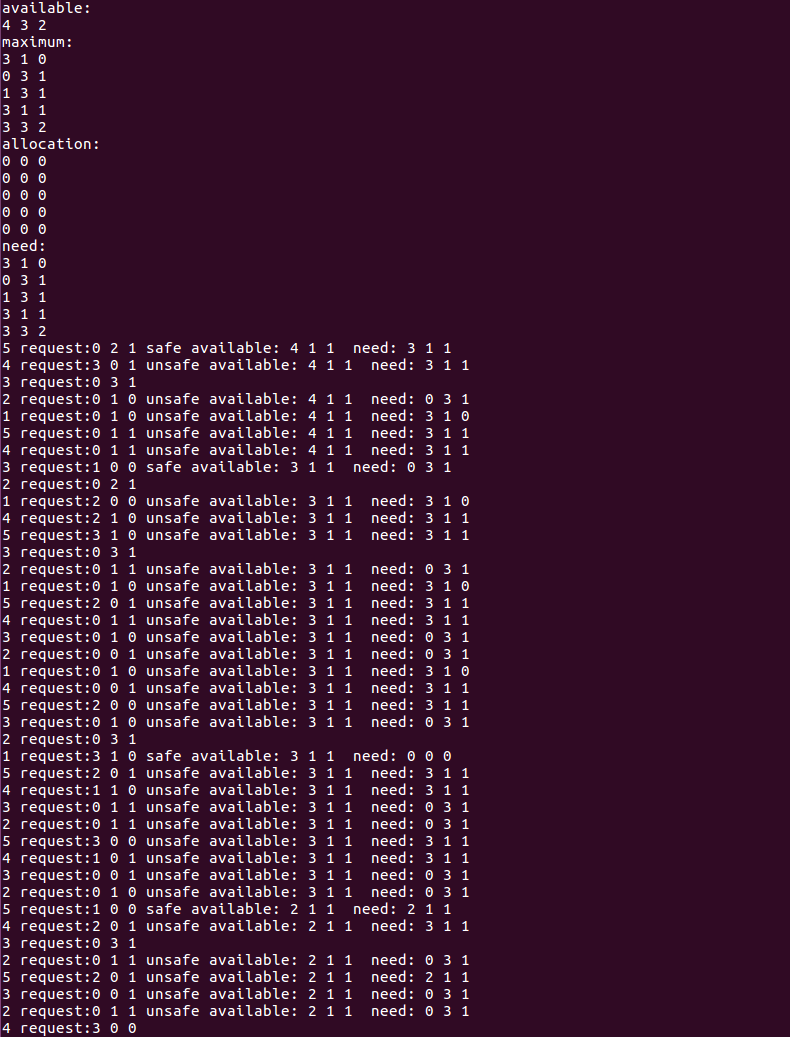
And I also write functions to achieve resource-request algorithm.

In the function, I mainly judge if available resources can satisfy each request and if the system is a safe state.



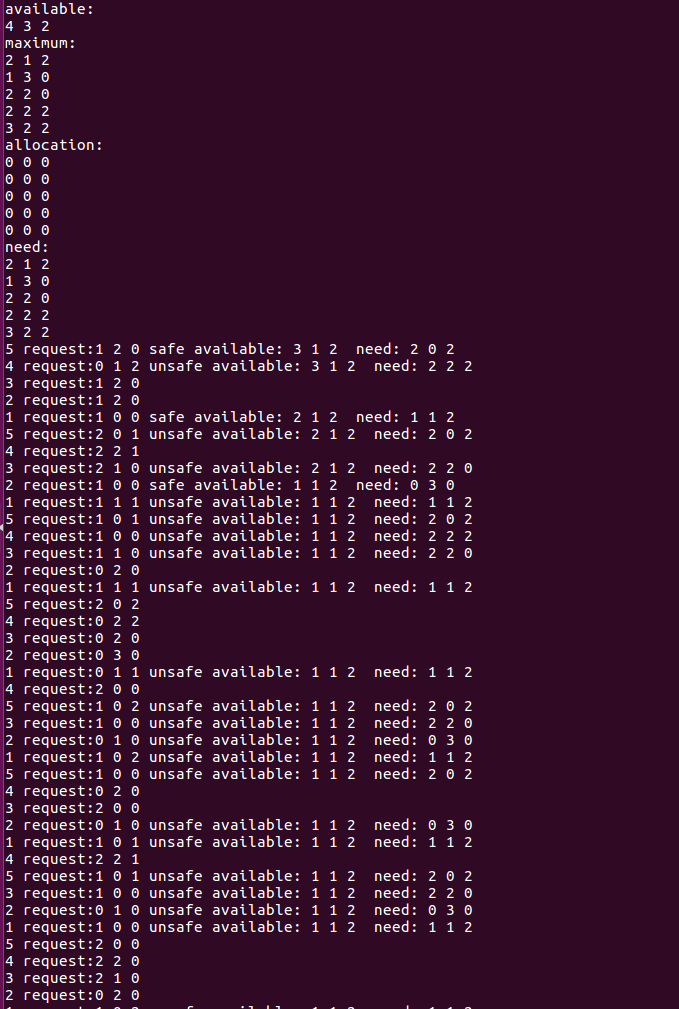
Test:

If I input ./a.out 4 3 2



Five threads will compete to request resources, in the case, the resources cannot satisfy needs of each thread, because not all elements in need array are zeros.

But if I change the maximum, the resources can satisfy each process.



You can find that need array is zero which means that each process can be satisfied.

