Operating System Project 6

潘梓丞 517030910349

The goal of the project is to simulate the banker's algorithm
The porject was done by VM VirtualBox 5.2.18
The code are written by C and the library needed will be shown in code

idea

In order to simulate the process of the banker's algorithm, firstly define global variables $Max, allocation, Need, available$. Then we just do operations on these arrays. There's one thing left to do, check when a process is asking for resource, if the request can't be satisfied, report error condition. This project is simple and pretty easy to complete.

code

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#define NUMBER_OF_CUSTOMERS 5
#define NUMBER_OF_RESOURCES 4

int available[NUMBER_OF_RESOURCES];
int max[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];
int allo[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];
int need[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];

int request_resources(int customer_num, int request[])
{
    int s;
    int flag=-1;
    for(s=0;s<NUMBER_OF_RESOURCES;s++){
        if(available[s]<request[s])
            flag=0;
    }

    if (flag)
    {
        for (s=0;s<NUMBER_OF_RESOURCES;s++){
            available[s]-=request[s];
            allo[customer_num][s]+=request[s];
        }
    }

    return flag;
}

void release_resources(int customer_num, int release[])
{
    int s;
    int flag=-1;
    for (s=0;s<NUMBER_OF_RESOURCES;s++){
        if(allo[customer_num][s]<release[s])
            flag=0;
    }

    if(flag){
        for(s=0;s<NUMBER_OF_RESOURCES;s++){
            available[s]+=release[s];
            allo[customer_num][s]-=release[s];
        }
    }
    else{
        printf("customer %d doesn't have enough resources",customer_num);
    }
```

```
}

int main(int argc,char *argv[])
{
    int i,j;
    for (i=0;i<argc-1;i++)
        available[i]=atoi(argv[i+1]);

    FILE *in;
    char *temp;
    int row=0;
    char task[20];
    in=fopen("MAX.txt","r");

    while (fgets(task,20,in) != NULL){
        temp=strdup(task);
        for(i=0;i<NUMBER_OF_RESOURCES;i++)
            max[row][i]=atoi(strsep(&temp,","));
        row++;
    }

    char order[20];
    int consu;
    int req;
    int reqs[4];
    while(1)
    {
        scanf("%s",order);
        if(!strcmp("RQ",order)){
            scanf("%d",&consu);
            for(i=0;i<NUMBER_OF_RESOURCES;i++){
                scanf("%d",&req);
                reqs[i]=req;
            }
            if(!request_resources(consu,reqs)){
                printf("error request\n");
            }
        }

        if(!strcmp("RL",order)){
            scanf("%d",&consu);
            for(i=0;i<NUMBER_OF_RESOURCES;i++){
                scanf("%d",&req);
                reqs[i]=req;
            }
            release_resources(consu,reqs);
        }

        if(!strcmp("*",order)){
            printf("available:\n");
            for(i=0;i<NUMBER_OF_RESOURCES;i++)
                printf("%d ",available[i]);

            printf("\nmaximum:\n");
            for(i=0;i<NUMBER_OF_CUSTOMERS;i++){
                for(j=0;j<NUMBER_OF_RESOURCES;j++){
                    printf("%d ",max[i][j]);
                }
                printf("\n");
            }

            printf("allocation:\n");
            for(i=0;i<NUMBER_OF_CUSTOMERS;i++){
                for(j=0;j<NUMBER_OF_RESOURCES;j++){
                    printf("%d ",allo[i][j]);
                }
                printf("\n");
            }

            printf("need:\n");
            for(i=0;i<NUMBER_OF_CUSTOMERS;i++){
                for(j=0;j<NUMBER_OF_RESOURCES;j++){
                    printf("%d ",max[i][j]-allo[i][j]);
                }
                printf("\n");
            }
        }

        if(!strcmp("exit",order)){
            printf("closing\n");
            break;
        }
    }
```