Operating System Project 4

潘梓丞 517030910349

The goal of the project is to implement scheduler 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

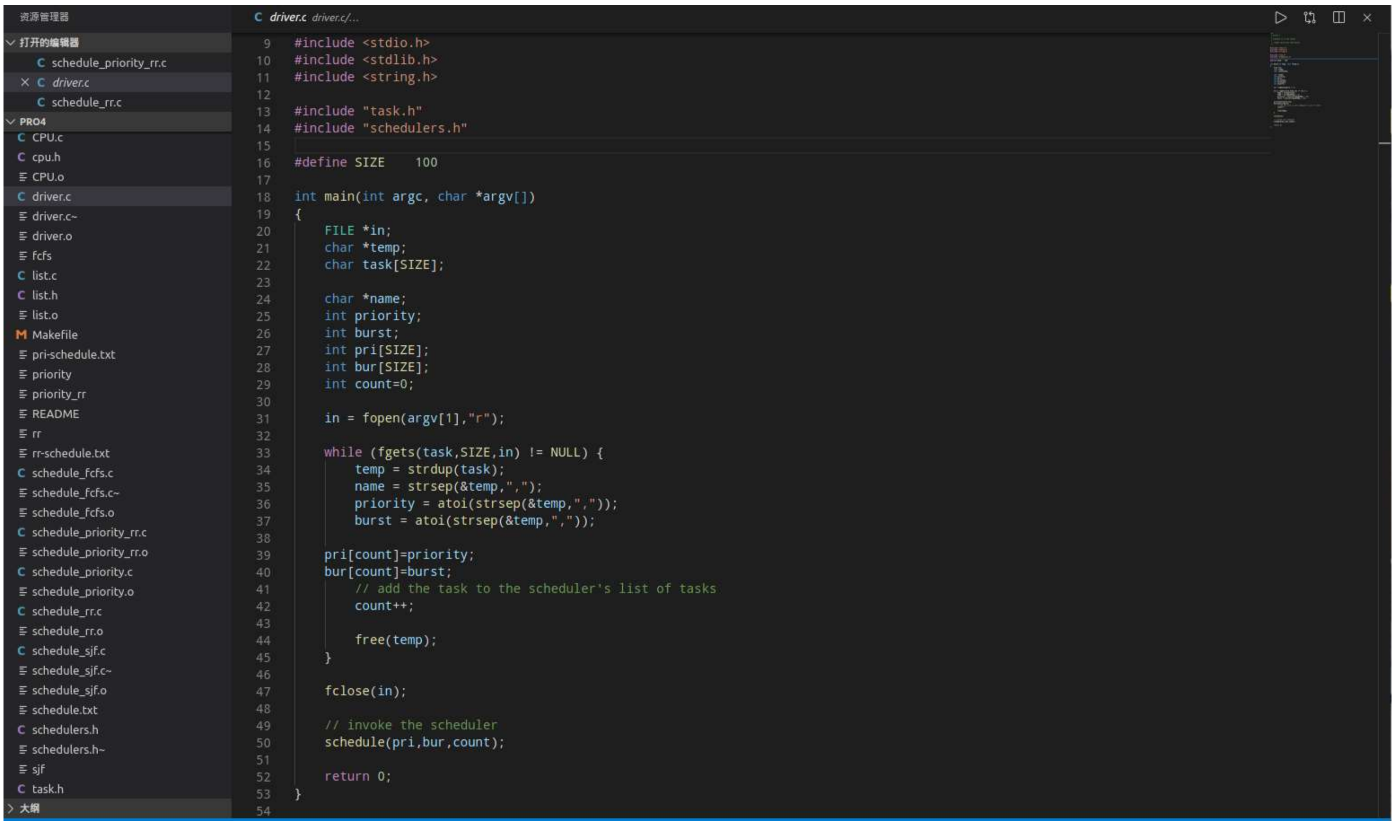
idea

Write different function $schedule()$ in different schedule.c to realize different schedule ways. When we are making the executable file, transfer different functions so that the program will schedule the process in different ways.
When we design a alorithm to schedule, add a variable $inttime$ to store the running time, so we can calculate the numbers of turnaround time, waiting time, respose time. Sometime the waiting time is hard to calculate, on this condition we can calculate the turnaround substract burst time instead.

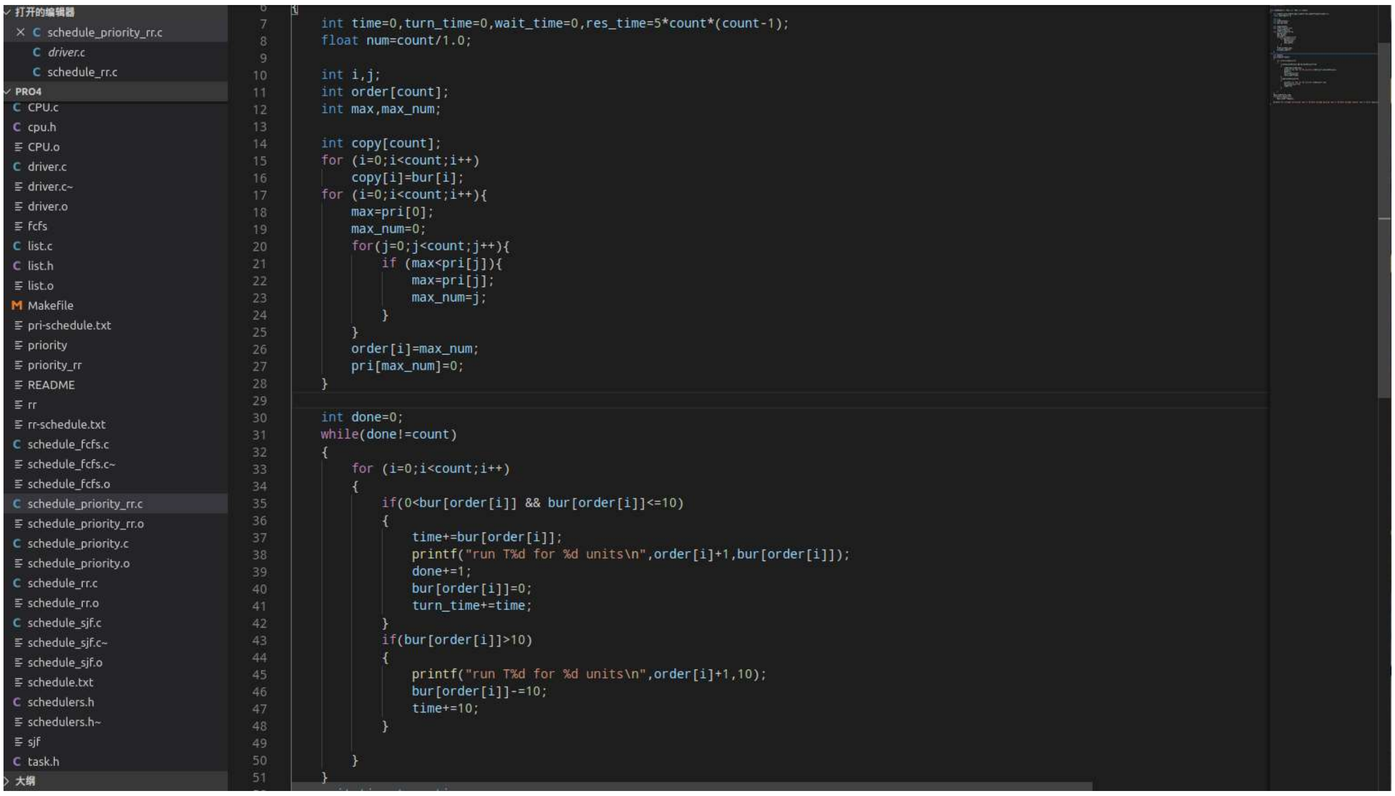
some problem

While running process using rr or priority_rr, it is hard to calculate waiting time straightly because once we change to run another process, we have to add the running time for all the process waiting to run. But fortunately, on the condition that all the process arrive at time 0 concurrently, the total waiting time is exactly the total turnaround time substact total burst time. By this way, we can simply record the time when a process is done.

code



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

#include "task.h"
#include "schedulers.h"

#define SIZE    100

int main(int argc, char *argv[])
{
    FILE *in;
    char *temp;
    char task[SIZE];

    char *name;
    int priority;
    int burst;
    int pri[SIZE];
    int bur[SIZE];
    int count=0;

    in = fopen(argv[1],"r");

    while (fgets(task,SIZE,in) != NULL) {
        temp = strdup(task);
        name = strsep(&temp,",");
        priority = atoi(strsep(&temp,","));
        burst = atoi(strsep(&temp,","));

    pri[count]=priority;
    bur[count]=burst;
        // add the task to the scheduler's list of tasks
        count++;

        free(temp);
    }

    fclose(in);

    // invoke the scheduler
    schedule(pri,bur,count);

    return 0;
}
```



```c
    int time=0,turn_time=0,wait_time=0,res_time=5*count*(count-1);
    float num=count/1.0;

    int i,j;
    int order[count];
    int max,max_num;

    int copy[count];
    for (i=0;i<count;i++)
        copy[i]=bur[i];
    for (i=0;i<count;i++){
        max=pri[0];
        max_num=0;
        for(j=0;j<count;j++){
            if (max<pri[j]){
                max=pri[j];
                max_num=j;
            }
        }
        order[i]=max_num;
        pri[max_num]=0;
    }

    int done=0;
    while(done!=count)
    {
        for (i=0;i<count;i++)
        {
            if(0<bur[order[i]] && bur[order[i]]<=10)
            {
                time+=bur[order[i]];
                printf("run T%d for %d units\n",order[i]+1,bur[order[i]]);
                done+=1;
                bur[order[i]]=0;
                turn_time+=time;
            }
            if(bur[order[i]]>10)
            {
                printf("run T%d for %d units\n",order[i]+1,10);
                bur[order[i]]-=10;
                time+=10;
            }

        }
    }
```