操 作 系 统 原 理

实 验 报 告

实验题目： 实验 1：高响应比作业调度

实验时间： 2022 年 11 月 3 日

学号姓名： 22129390 吕强

实验目的和要求

1．掌握高响应比作业调度的概念和算法；

2．加深对处理机分配的理解。

实验内容及步骤

在 Visual C++ 6.0 集成开发环境下使用 C 或 C++语言，实现作业高响应比调度算法，学会运行程序和中断当前程序的运行。

1．在源程序上写出注释。

2．画出程序流程图。

3．调试程序并写出运行结果。

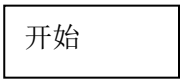
实验程序

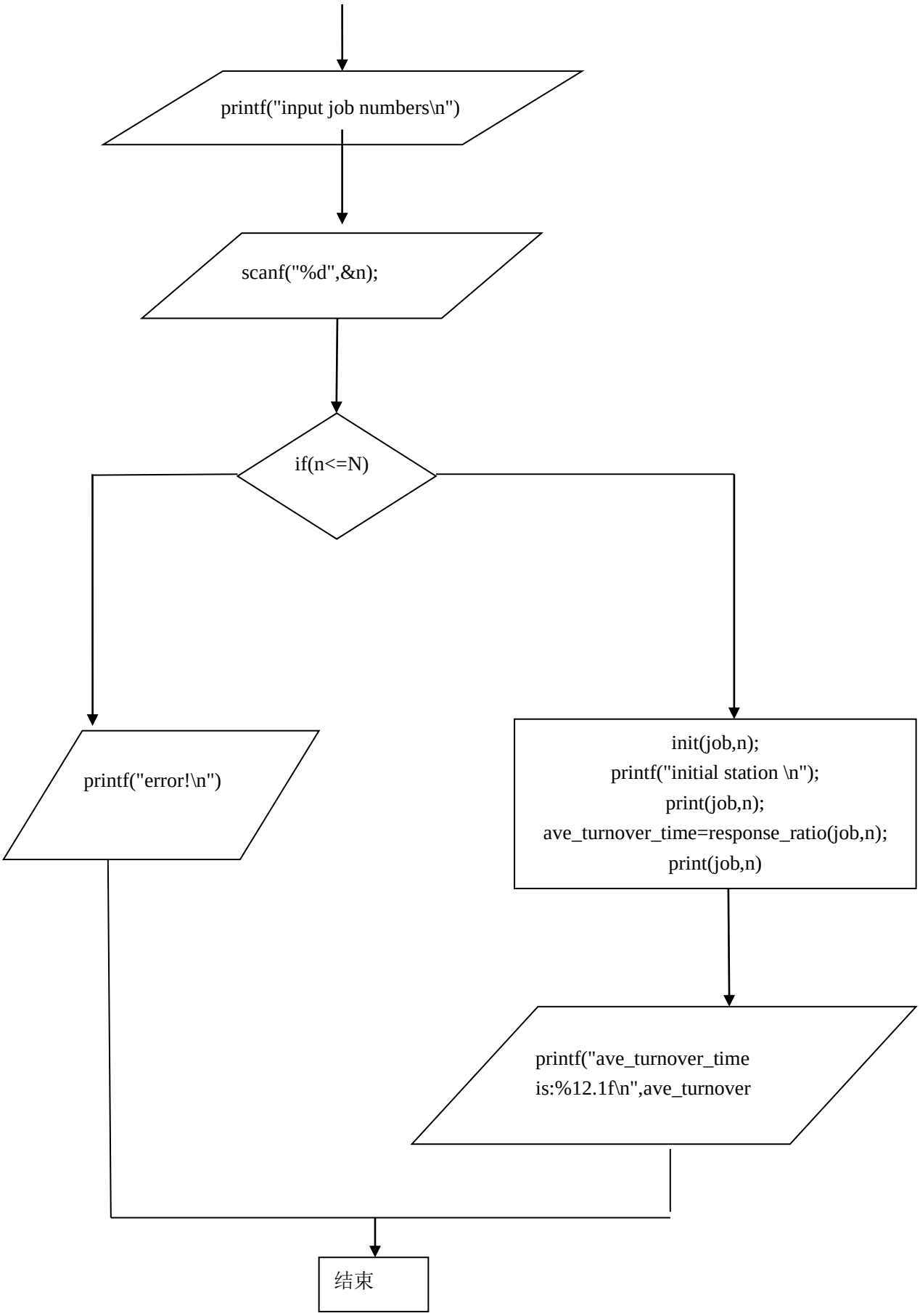
```
#include <malloc.h>
#include <stdio.h>
#include <string.h>
#define NULL 0
#define N 10
typedef struct table
{    char    name[8];        /*作业名*/
     float in_well;          /*进入输入井时间*/
     float begin_run;        /*开始运行时间*/
     float run_time;         /*运行时间*/
     float end_run;          /*结束运行时间*/
     float turnover_time; /*周转时间*/
  }jobtable;
void init(jobtable job[],int n) /*初始化作业表*/
{
     int i,j;
     printf("input %d job information\n",n);
     printf("in_well run_time name\n");
     for(i=0;i<n;i++)
     {
```

```
            scanf("%f %f %s",&job[i].in_well,&job[i].run_time,job[i].name);
            job[i].begin_run=0.0;
            job[i].end_run=0.0;
            job[i].turnover_time=0.0;
        }
}
void print (jobtable job[],int n)   /*输出链表*/
{   int i;
    printf("name   in_well   run_time   begin_run   end_run   turnover_time\n");
    for(i=0;i<n;i++)
    {   printf("%s\t%0.1f\t%0.1f\t",job[i].name,job[i].in_well,job[i].run_time);
        if(job[i].begin_run==0.0&&job[i].end_run==0.0&&job[i].turnover_time==0.0)
            printf("                                                \n");
        else
        printf("%9.1f%9.1f\t%0.1f\n",job[i].begin_run,job[i].end_run,job[i].turnover_time);
    }
}
 void swap(jobtable job[],int p,int q)
{       float temp1;
        char temp2[8];
        strcpy(temp2,job[p].name);strcpy(job[p].name,job[q].name);strcpy(job[q].name,temp2);
        temp1=job[p].in_well;job[p].in_well=job[q].in_well;job[q].in_well=temp1;
        temp1=job[p].run_time;job[p].run_time=job[q].run_time;job[q].run_time=temp1;
 }
 float response_ratio(jobtable job[],int n)/*模拟当前作业表的调度过程*/
 {      int i,j,temp;
        float average_time,ratio1,ratio2;
        job[0].begin_run=job[0].in_well;
        job[0].end_run=job[0].begin_run+job[0].run_time;
        job[0].turnover_time=job[0].end_run-job[0].begin_run;
        average_time=job[0].turnover_time;
        for(i=1;i<n;i++)
        {   if(job[i].in_well<=job[i-1].end_run)
            {   j=i+1;temp=i;
                ratio1=1+(job[i-1].end_run-job[i].in_well)*1.0/job[i].run_time;
                while(j<n&&job[j].in_well<=job[i-1].end_run)
                {
                    ratio2=1+(job[i-1].end_run-job[j].in_well)*1.0/job[j].run_time;
                    if(ratio2>ratio1)   temp=j;
                    j++;
                }
                if(temp!=i)
                    swap(job,i,temp);
            }
```

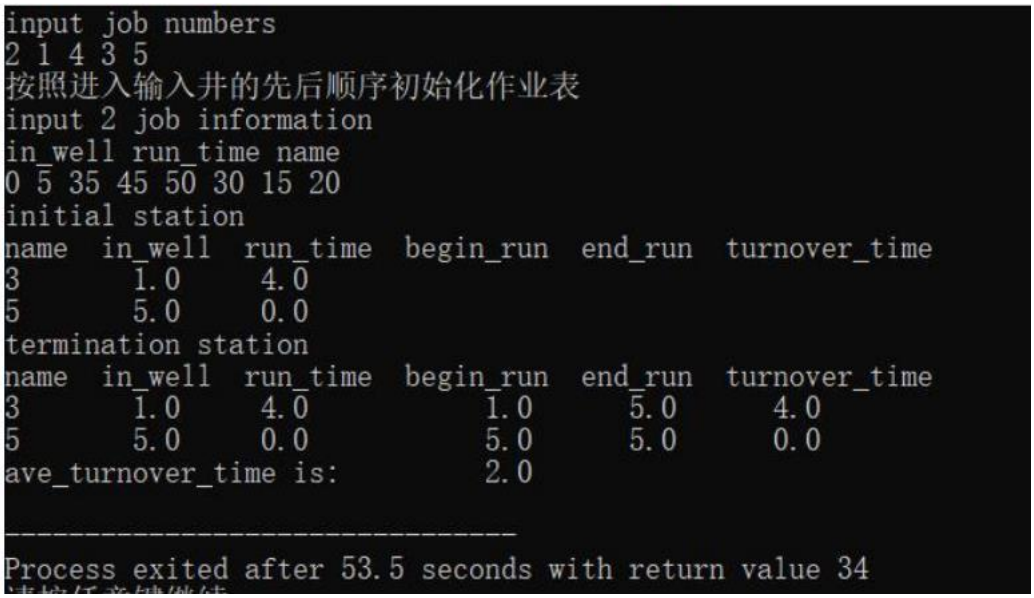
```
        job[i].begin_run=job[i-1].end_run;
        job[i].end_run=job[i].begin_run+job[i].run_time;
        job[i].turnover_time=job[i].end_run-job[i].in_well;
        average_time=average_time+job[i].turnover_time;
    }
    return(average_time/n);
}
void main()
{   int n;
    float ave_turnover_time;
    jobtable job[N];
    printf("input job numbers\n");
    scanf("%d",&n);
    if(n<=N)
    {
        printf("按照进入输入井的先后顺序初始化作业表 \n");
        init(job,n);
        printf("initial station \n");
        print(job,n);
        ave_turnover_time=response_ratio(job,n);
        printf("termination station \n");
        print(job,n);
        printf("ave_turnover_time is:%12.1f\n",ave_turnover_time);
    }
  else   printf("error!\n");
}
```

程序流程图




printf("input job numbers\n")
scanf("%d",&n);
if(n<=N)
printf("error!\n")
init(job,n);
printf("initial station \n");
print(job,n);
ave_turnover_time=response_ratio(job,n);
print(job,n)
printf("ave_turnover_time
is:%12.1f\n",ave_turnover
结束

运行结果



分析讨论

起初刚才是没有思路，经过老师的讲解有了思路，如何让去画流程图，对我这种没有计算机基础的有了一些帮助

教师评语及成绩