# 实验报告

题目：车厢调度问题，即栈的出列顺序排列

班级：电信1809 姓名：胡伊曼 学号：U201813486 完成日期：2019.4.8

1. 需求分析
2. 用栈的push pop来将排序存入数组
3. 通过递归来遍历所有排列可能性
4. 打印所有排序
5. 概要设计
6. 栈的抽象数据类型定义如下：

ADT Stack

{

数据对象：D={ai|ai属于CharSet，i1,2,3,,,n，n>=0}

数据关系：R1 = {<ai-1,ai>|ai-1,ai属于D,i=1,2,,,n}

基本操作：InitStack（&S）

操作结果：构建一个空的线性表

}ADT Stack

1. 主程序

Void main（）

{

初始化；

输入len；

Arrange操作；

输出排列顺序；

}

三．详细设计

#include <stdio.h>

#include <stdlib.h>

#define N 10

typedef struct

{

int a[N];

int top;

}stack;

stack tran;//声明全局栈

int a[N]={0};//声明一个数组用于存放原火车顺序

int b[N];//声明一个数组用于存放输出的出栈的火车

int len;//用于存放火车的列数

void init(stack \*p)

{

p->top=0;

printf("已初始化! \n");

}

int is\_empty(stack \*p)

{

if(p->top==0)

return 1;

else

return 0;

}

int push(stack \*p,int x)

{

if(p->top>=N)

{

printf("顺序栈已满！\n");

return 0;

}

p->a[p->top+1]=x;//从栈a[1]开始存

p->top++;

return 1;

}

int pop(stack \*p)

{

if(p->top<=0)

{

printf(" 顺序栈为空！\n");

exit(1);

}

return p->a[p->top--];

}

void print()//输出b,即排序之后的列车序号

{

int i;

for(i=0;i<len;i++)

{

printf("%d ",b[i]);

}

printf("\n");

}

void arrange(int k,int m)//处理到原始数据中的第k个元素，m：若出栈，应放在当前输出数组的第seq\_pos个位置

{

int pop\_num;

if(k<len)//在有效范围内入栈 对于入栈的元素：1继续入栈，2出栈

{

push(&tran,a[k]);//入栈

arrange(k+1,m);//选择1 下一列火车入栈

pop(&tran);//返回入栈前的情况 以便选择2 出栈

}

if(!is\_empty(&tran))//栈内有火车出栈 对于出栈的元素： 1继续出栈，2入栈

{

pop\_num=pop(&tran);

b[m]=pop\_num; // 放入b[]中，准备输出

m++;

arrange(k,m); //由于先前返回入栈前，当前这列火车并没入栈，所以返回当前火车重新入栈

push(&tran,pop\_num);//返回出栈前的情况 以便选择 2

}

if(k>=len&&is\_empty(&tran))//当最后一列火车入栈了且所有火车已出栈 一种情况分析完毕

{

print();

}

}

void main()

{

int i;

init(&tran);

printf("请输入火车的列数：");

scanf("%d",&len);

for(i=0;i<len;i++)

{

a[i]=i+1;

}

arrange(0,0);

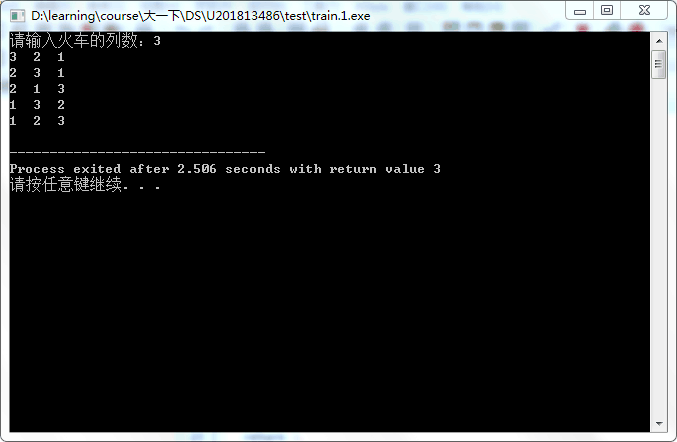
}

-

四．调试分析

1.一开始递归+分类没有理清楚->多方请教+尝试

五．测试结果

N=3

N=4

