**实习报告**

题目：利用栈和队列编制一个模拟车厢调度的程序

班级：电信1809 姓名：朱燚 学号：U201813488 完成日期：2019/4/8

1. 需要分析

（1）本演示程序中，输入值都为int型。

（2）演示程序以用户和计算机的对话方式执行，即在计算机终端上显示“提示信息”之后，由用户在键盘上输入演示程序中规定的运算命令。

（3）程序执行的命令包括：

1）需要输入车厢的长度。

1. 概要设计

为了实现上述程序，应使用栈。

（1）栈的抽象数据类型定义：

ADT Stack{

数据对象：D={

数据关系：R1=

基本操作：

InitStack(&S)

操作结果：创造一个空栈S

Push(&S,e)

初始条件：栈S已存在

操作结果：插入元素e为新的栈顶元素

Pop(&S,&e)

初始条件：栈S已存在且非空

操作结果：删除S的栈顶元素，并用e返回其值

}ADT Stack

（2）本程序包含三个模块：  
 1）主函数模块：

int main()

{

初始化；

执行命令；

}

2) 栈模块：实现抽象数据类型

各模块之间调用关系如下：

主程序模块

栈模块

3.详细设计

（1）栈类型

typedef struct stack

{

Elemtype \*top; //栈顶

Elemtype \*base; //栈底

int size; //栈的大小

}stack,\*pstack; //栈的元素类型

栈的部分操作算法：

stack init()

{

stack S;

S.base=(Elemtype\*)malloc(Max\_Size\*sizeof(Elemtype));

if(!S.base)exit(0);

S.top=S.base;

S.size=Max\_Size;

return S;

}

void push(Elemtype e,pstack S)

{

if(S->top-S->base>=S->size)

{

S->base=(Elemtype\*)realloc(S->base,(Add\_size+Max\_Size)\*sizeof(Elemtype));

S->top=S->base+Max\_Size;

S->size=S->size+Add\_size;

}

\*(S->top)=e;

S->top++;

}

void pop(pstack S,Elemtype \*e)

{

if(S->base==S->top)exit(0);

S->top--;

\*e=\*(S->top);

}

//输出栈中的元素

void print(pstack S)

{

while(S->top!=S->base)

{

printf("%d\t",\*(S->top-1));

S->top--;

}

printf("\n");

}

//判断栈是否为空

int Empsta(pstack S)

{

if(S->base==S->top)

{

return 0;

}

else

{

return 1;

}

}

（2）主函数和其他函数的伪码算法

int main()

{

//主程序

Elemtype len;

Elemtype a[100];

stack S,L;

S=init(),L=init(); //初始化

printf("Please input the number:");

scanf("%d",&len);

dispatch(&S,0,0,len,&a); //模拟车厢调度

}

void dispatch(pstack S,int cur,Elemtype pos,Elemtype len,Elemtype \*a)

{

//模拟车厢调度

Elemtype e,m;

if(pos<len)

{

push(pos+1,S);

dispatch(S,cur,pos+1,len,a);

pop(S,&m);

}

if(Empsta(S)!=0)

{

pop(S,&e);

a[cur]=e;

cur++;

dispatch(S,cur,pos,len,a);

push(e,S);

}

if(Empsta(S)==0&&pos==len)

{

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

{

printf("%d\t",a[i]);

}

printf("\n");

}

}

（3）函数调用关系如下

主程序

Dispatch

pop push Empsta dispatch

4.调试分析

1. 本次作业虽然只有一个核心算法，但是该算法设计较为抽象，需要理解。
2. 算法的时空分析：
3. 该算法各操作时间复杂度比较合理。
4. 通过分析，该算法的复杂度为

5.用户手册（略）

6.测试结果：

执行命令’1’：键入 4确定车厢长度

输出：4 3 2 1

3 4 2 1

3 2 4 1

3 2 1 4

2 4 3 1

2 3 4 1

2 3 1 4

2 1 4 3

2 1 3 4

1 4 3 2

1 3 4 2

1 3 2 4

1 2 4 3

1 2 3 4

确定所有车厢调度的情况

7.附录：

源程序文件名清单（略）