实习报告

题目:车厢调度。

班级：电信9班 姓名：王嘉成 学号：U201813481 日期：2019.04.08

**需求分析。**

1. 本次程序中，用户需输入车厢序列的长度n。执行程序后将会输出所有满足要求的序列。
2. 执行程序后，车厢序列先按1，2，3…n的顺序排列，然后在每个操作状态都有车厢入栈和出栈的操作，直到车厢序列全部经过入栈和出栈的操作后，输出此时的输出序列。
3. 测试数据 n=1: 1

n=2: 2 1

1. 2

n=3：3 2 1

2 3 1

2 1 3

1 3 2

1 2 3

n=4: 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

**概要设计。**

1. 本程序抽象数据类型只有栈。

typedef int Elemtype ;

ADT Stack{

数据对象:D={ai|ai∈Elemtype,i=1,2,3,…,n,n>=0}

数据关系R1={<ai-1,ai>|ai-1,ai∈D,i=1,2,3,…,n}

基本操作：

Stack init(&S)

操作结果：构造一个空栈S。

Void push(&S,e)

初始条件：栈S存在。

操作结果：将e存入栈S。

Void pop(&S,&e)

初始条件：栈S存在且非空。

操作结果：将S中的栈顶元素弹出，并赋值给e。

Void print(&S)

初始条件：栈S存在。

操作结果：将栈中元素输出。

Int Empsta(&S)

初始条件：栈存在

操作结果：判断栈是否为空，为空，返回0，不为空，返回1。

其余与本程序无关的基本操作再次就不赘述了。

}ADT List

**详细设计。**

以下使是该程序的完整代码：

#include<stdio.h>

#include<stdlib.h>

#define Max\_Size 10

#define Add\_size 10

typedef int Elemtype;

typedef struct stack

{

Elemtype \*top;

Elemtype \*base;

int size;

}stack,\*pstack;

stack init()//初始化一个栈

{

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)//将e存入栈中

{

if(S->top-S->base>=S->size)//如果栈满，则多开辟空间给栈

{

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)//将栈顶元素弹出，并用e存储

{

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

S->top--;

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

}

void print(pstack S)//打印栈元素

{

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

{

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

S->top--;

}

printf("\n");

}

int Empsta(pstack S)//判断是否栈空

{

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

{

return 0;

}

else

{

return 1;

}

}

void dispatch(pstack S,int cur,Elemtype pos,Elemtype len,Elemtype \*a)//车厢调度函数，递归思想

{

Elemtype e,m;

if(pos<len)//当当前车厢序号小于最大值，将该序号存入栈中

{

push(pos+1,S);

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

pop(S,&m);//将已经操作过的数据弹出，操作上一种情况。

}

if(Empsta(S)!=0)//当当前车厢序号大于等于最大值且栈不空，弹出栈顶元素并给赋值给输出数组。

{

pop(S,&e);

a[cur]=e;

cur++;

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

push(e,S);

}

if(Empsta(S)==0&&pos==len)//如果栈空且当当前车厢序号大于等于最大值且栈不空，将输出数组输出。

{

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

{

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

}

printf("\n");

}

}

int main()

{

Elemtype len;

Elemtype a[100];

stack S,L;

S=init(),L=init();

printf("Please input the number:");

scanf("%d",&len);

dispatch(&S,0,0,len,&a);

}

**调试分析。**

1. 该程序需要用到递归函数，比较绕，并且不易想清楚，所以当写dispatch函数时，递归的位置很难确定，但经过一步一步的分析，再参照网上的代码，终于将该函数写出。
2. 算法的时空分析：

dispatch函数的时间复杂为O(1)，主函数中执行dispatch操作时要用到递归，故主函数的时间复杂度为O(n\*n)。

**用户使用说明。**

1. 程序的运行环境为Windos操作系统，执行文件为Dispatch.c。
2. 进入执行页面输入车厢序列总数，之后程序就会输出所有的结果。

**测试结果。**

n=1: 1

n=2: 2 1

1. 2

n=3：3 2 1

2 3 1

2 1 3

1 3 2

1 2 3

n=4: 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

**附录。**

#include<stdio.h>

#include<stdlib.h>

#define Max\_Size 10

#define Add\_size 10

typedef int Elemtype;

typedef struct stack

{

Elemtype \*top;

Elemtype \*base;

int size;

}stack,\*pstack;

stack init()//初始化一个栈

{

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)//将e存入栈中

{

if(S->top-S->base>=S->size)//如果栈满，则多开辟空间给栈

{

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)//将栈顶元素弹出，并用e存储

{

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

S->top--;

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

}

void print(pstack S)//打印栈元素

{

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

{

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

S->top--;

}

printf("\n");

}

int Empsta(pstack S)//判断是否栈空

{

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

{

return 0;

}

else

{

return 1;

}

}

void dispatch(pstack S,int cur,Elemtype pos,Elemtype len,Elemtype \*a)//车厢调度函数，递归思想

{

Elemtype e,m;

if(pos<len)//当当前车厢序号小于最大值，将该序号存入栈中

{

push(pos+1,S);

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

pop(S,&m);//将已经操作过的数据弹出，操作上一种情况。

}

if(Empsta(S)!=0)//当当前车厢序号大于等于最大值且栈不空，弹出栈顶元素并给赋值给输出数组。

{

pop(S,&e);

a[cur]=e;

cur++;

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

push(e,S);

}

if(Empsta(S)==0&&pos==len)//如果栈空且当当前车厢序号大于等于最大值且栈不空，将输出数组输出。

{

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

{

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

}

printf("\n");

}

}

int main()

{

Elemtype len;

Elemtype a[100];

stack S,L;

S=init(),L=init();

printf("Please input the number:");

scanf("%d",&len);

dispatch(&S,0,0,len,&a);

}