# 车厢调度

停在铁路调度站入口处的车厢序列的编号1至n。设计该程序，输入n，输出所有可能输出长度为n的车厢序列。

【测试数据】 输入n=3 输出 3 2 1

2 3 1

2 1 3

1 3 2

1 2 3

输入 n=2 输出 2 1

12

【概要设计】 定义栈tem[ ], out[ ]分别储存将要出站的车厢序列和已经出站的车厢序列。调用递归函数arrange输出结果。

入栈操作：

void Push(int a[],int e)

{

a[0]++;

a[a[0]]=e;

}

栈的赋值操作：

void equ(int a[],int b[])

{

for(int i=0;i<=a[0];i++) b[i]=a[i];

}

Arrange函数：

void arrange(int out[],int tem[],int i,int n)

{

int out1[100],tem1[100];

if(i>n)//若已出站和将出站的车箱数和大于n则输出

{

for(int a=1;a<=out[0];a++) printf("%d ",out[a]);

for(int j=tem[0];j>0;j--) printf("%d ",tem[j]);

printf("\n");

}

else

{

equ(tem,tem1);equ(out,out1);//保留tem out的原始值

Push(tem,i);//i车厢进入待出站栈

arrange(out,tem,i+1,n);

equ(tem1,tem);equ(out1,out);//返回tem out的原始值

if(tem[0]!=0)

{

out[0]++;out[out[0]]=tem[tem[0]];tem[0]--;

arrange(out,tem,i,n);

}

}

}

【调试分析】注意递归前要保留tem out的原始值并返回原始值。

【测试结果】

n=4

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

n=5

输出：

5 4 3 2 1

4 5 3 2 1

4 3 5 2 1

4 3 2 5 1

4 3 2 1 5

3 5 4 2 1

3 4 5 2 1

3 4 2 5 1

3 4 2 1 5

3 2 5 4 1

3 2 4 5 1

3 2 4 1 5

3 2 1 5 4

3 2 1 4 5

2 5 4 3 1

2 4 5 3 1

2 4 3 5 1

2 4 3 1 5

2 3 5 4 1

2 3 4 5 1

2 3 4 1 5

2 3 1 5 4

2 3 1 4 5

2 1 5 4 3

2 1 4 5 3

2 1 4 3 5

2 1 3 5 4

2 1 3 4 5

1 5 4 3 2

1 4 5 3 2

1 4 3 5 2

1 4 3 2 5

1 3 5 4 2

1 3 4 5 2

1 3 4 2 5

1 3 2 5 4

1 3 2 4 5

1 2 5 4 3

1 2 4 5 3

1 2 4 3 5

1 2 3 5 4

1 2 3 4 5

【附录】