# 实验报告

### 姓名：周宇 班级：电信1808 学号：U201813451

### 题目：车厢调度的递归实现

### 需求分析

车厢递归问题可以归结为数的进栈和出栈。当一个数字进栈后，有两种处理方法：要么立刻出栈，要么下个数继续进栈（如果还有数可以进栈）。这表明可以用递归的方式处理该问题。

1. 概要设计

该程序主要用到的ADT为顺序栈，主程序调用函数car\_remake，在car\_remake中调用栈的基本操作函数（push、pop）

1. 详细设计

源代码如下所示

#include<stdio.h>

#include<stdlib.h>

typedef struct

{

int \*base;

int \*top;

int size;

}stack;

stack \*initstack()

{

stack \*s;

s->base=(int \*)malloc(100\*sizeof(int));

s->top=s->base;

s->size=100;

return s;

}

void push(stack \*s,int e)

{

\*s->top=e;

s->top++;

}

int pop(stack \*s)

{

int i;

s->top--;

i=\*s->top;

return i;

}

void print(stack \*s)

{

int \*temp;

temp=s->top;

while(temp!=s->base)

{

temp--;

printf("%d\t",\*temp);

}

}

void car\_remake(stack \*s,int output[],int pos,int i,int n)

{

if(pos<n)

{

push(s,pos+1);

car\_remake(s,output,pos+1,i,n);

pop(s);

}

if(!(s->top==s->base))

{

int m;

m=pop(s);

output[i]=m;

i++;

car\_remake(s,output,pos,i,n);

push(s,m);

}

if(pos==n&&s->top==s->base)

{

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

{

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

}

printf("\n");

}

}

void main()

{

stack \*s;

s=initstack();

int n=4;

int output[100];

int pos=1;

push(s,1);

int i=0;

car\_remake(s,output,pos,i,n);

}

1. 调试分析

该算法运用双重递归，下一个元素处理完后返回，再处理出栈的递归，类似于嵌套循环。进栈递归跳出条件为最后一个元素进栈，出栈递归的跳出条件为栈空

1. 测试结果

输入4 得结果

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