第三章作业.

1. 算法思路：将S1作为主要的数据存储栈，将队列"倒放"在S1中

将S2作用功能实现的辅助栈，用于临时存储一些数据.

三个运算的代码如下：

```
template <class T>
void enquene(T data){
    push(S1, data);//直接压入S1即可
}

bool queue_empty(){
    return empty(S1);//S1是数据栈，S1空则队列空.
}

template <class T>
void dequene(T&x){//x用来存储出列元素
    if(!empty(S2)){//队列已空
        cout<<"非法操作!"<<endl;
        return;
    }
    T tmp;
    while(!empty(S1)){//先把栈底元素取出放出S2，将队列"正放"在S2中.
        pop(S1,tmp);
        push(S2,tmp);
    }
    pop(S2,tmp);//弹出队头元素
    while(!empty(S2)){//再把剩余元素"倒放"回S1.
        pop(S2,tmp);
        push(S1,tmp);
    }
}
```

2. 解：开出车站的顺序有 $\frac{1}{n+1}C_{2n}^{n}$ 种可能，理由如下：

利用 Knuth 方法，设入栈为 I(in)，出栈为 O(out)，对于 n 辆火车进栈的出栈入栈序列共有 C_{2n}^{n} 种，而其中有一部分IO序列是不合法的，即在序列的某一个位置前 O 的数目大于了 I 的数目，对于这种序列，我们将该位置及之前的所有 I 变为 O，所有 O 变为 I，从而使得序列中有 $(n+1)$ 个 I $(n-1)$ 个 O，显然不合法序列与变换后序列是一一对应的，故不合法序列有 C_{2n}^{n-1} 个，

故合法序列有 $C_{2n}^{n}-C_{2n}^{n-1}=\frac{1}{n+1}C_{2n}^{n}$（Catalan 数）

③ 证：①必要性：利用反证法：

假设有在下标 i,j,k，$i<j<k \wedge P_j<P_k<P_i$，

$\because i<j<k \wedge P_j<P_k<P_i$

$\therefore$ 在 P_i 出栈时，P_j、P_k 还留在栈中

又 $P_j<P_k$ 且 $j<k$

故 P_j 在 P_k 上方，距栈顶更近，

即 P_k 先于 P_j 入栈（$P_k>P_j$）

这与初始输入序列 $1,2,\cdots,n$ 的递增性矛盾！

$\therefore$ 不存在下标 i,j,k，满足 $i<j<k$ 同时 $P_j<P_k<P_i$，

②充分性：用归纳法证明：

$n=3$ 时，可得到的输出序列为 123，132，213，231，321

不存在下标 i,j,k，$i<j<k \wedge P_j<P_k<P_i$，符合条件，

假设当 $n=m(m\in N^+)$ 时结论成立，则当 $n=m+1$ 时，

$P_1P_2\cdots P_mP_{m+1}$ 不存在下标 i,j,k，$i<j<k \wedge P_j<P_k<P_i$

设 $P_a=m+1$，则序列 $P_1P_2\cdots P_{a-1}P_{a+1}\cdots P_mP_{m+1}$

可以通过栈输出得到，

考虑 $P_b=m$，

① $b\in\{1,2,\cdots,a-1\}$，即序列为 $P_1P_2\cdots P_{a-1}\overset{m}{\textcircled{P_b}}P_{a+1}\cdots P_{a-1}P_{a+1}\cdots P_mP_{m+1}$，

可以按照如下操作通过栈得到该序列：当 P_{a-1} 从栈中弹出后立刻将 $m+1$ 压入后弹出，余下的 $P_{a+1}\cdots P_{m+1}$ 按原顺序弹出，

② $b\in\{a+1,\cdots,m+1\}$，即序列为 $P_1P_2\cdots P_{a-1}P_{a+1}\cdots P_b\cdots P_mP_{m+1}$

若 $b>a+1$，则在 P_a 与 P_b 间至少有 1 个数 P_k 满足 $a<k<b \wedge P_k<P_b<P_a$，

与条件矛盾！

$\therefore b=a+1$，

故可以按照如下操作通过栈得到该序列：在 P_{a-1} 从栈中弹出后立刻将 P_b 和 P_a 依次压入栈再弹出，余下的按原顺序弹出，

$\therefore$ 当 $n=m+1$ 时结论成立，

证毕。