数据结构实验报告—栈和队列

设计者姓名：张帆

设计者班级：2 班

设计者学号：20192131077

上机环境：DEV-C++

设计日期：2020-11-1

一、实验题目

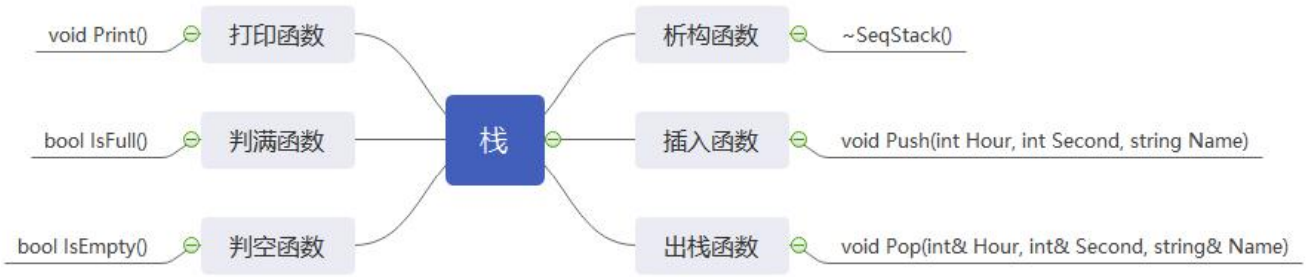
设停车场内只有一个可停放 n 辆汽车的狭长通道，且只有一个大门可供汽 车进出。 汽车在停车场内按车辆到达时间的先后顺序，依次由南向北排列（大门在最 北端，最先到达的第一辆车停放在车场的最南端），若车场内已停满 n 辆车，则 后来的汽车只能在门外的便道即候车场上等候，一旦有车开走，则排在便道上的 第一辆车即可开入；当停车场内某辆车要离开时，在它之后进入的车辆必须先退 出车场为他让路，待该辆车开出大门外，其他车辆再按原次序进入车场，每辆停 放在车场的车在他离开停车场时必须按它停留的时间长短交纳费用。用栈模拟停车场，用队列模拟车场外的便道，按照从键盘获取的数据序列进 行模拟管理。每一组输入数据包括 3 个数据项：汽车到达 1）或者离开 2）、汽车 牌照号码以及到达或离开的时刻。对每一组输入数据进行操作后的输出信息为: 若是车辆到达，则输出汽车在停车场内或便道上的停车位置； 若是车辆离开， 则输出汽车在停车场内停留的时间和应缴纳的费用（每小时收费 3 元，不足一小时按一小时计算，在便道上停留的时间不收费）。

二、实验项目目的

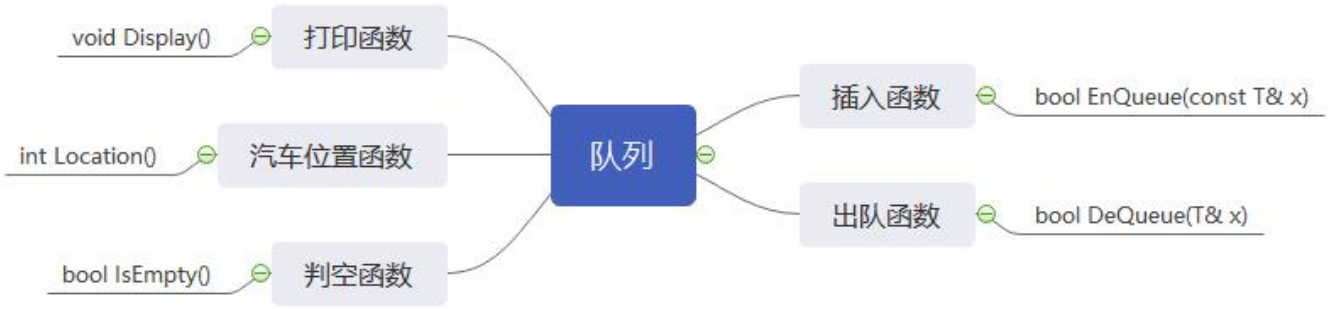
使学生深入掌握栈和队列应用的算法设计

三、实验项目的程序结构

顺序栈：



链队：



四、实验项目包含的各个文件中的函数的功能描述

```
#include <iostream>
#include <cmath>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <algorithm>
using namespace std;
/* run this program using the console pauser or add your own getch, system("pause") or input loop */

const int MaxSize = 100;

template <class T>
struct LinkNode
{
    T data;
    LinkNode<T> *link;
    LinkNode(LinkNode<T> *ptr = NULL)
    {
        link = ptr;
    }
    LinkNode(const T& item, LinkNode<T> *ptr = NULL)
    {
        data = item;
        link = ptr;
    }
};

template <class T>
class LinkedQueue
{
    public:
        LinkNode<T> *front,  *rear;

    public:
        LinkedQueue(): rear(NULL), front(NULL){}    //构造函数
        bool EnQueue(const T& x);                   //将 x 加入队列中
        bool DeQueue(T& x);                         //删除队头元素,x 返回其值
        bool IsEmpty()const{return (front == NULL) ? true : false;}  //判队空否
        void Display();   //打印
```

```
        int Location();   //返回位置
};

template <class T>
bool LinkedQueue<T>::EnQueue(const T& x)  //将 x 加入队列中
{
    if (front == NULL)
    {
        front = rear = new LinkNode<T>(x);
    }
    else
    {
        rear->link = new LinkNode<T>(x);
        rear = rear->link;
    }
    return true;
}

template <class T>
bool LinkedQueue<T>::DeQueue(T& x)   //删除队头元素
{
    if (IsEmpty() == true)  return false;
    LinkNode<T> *p = front;
    x = front->data;
    front = front->link;
    delete p;
    return true;
}

template <class T>
void LinkedQueue<T>::Display()      //打印
{
    if (IsEmpty() == true)
    {
        cout << "便道上没有停汽车！" << endl;
        return;
    }
    LinkNode<T> *cur = front;
    cout << "现在停在便道上的汽车为：" << endl;
    while (cur != NULL)
    {
        cout << "车牌号为" << cur->data << "的汽车" << endl;
        cur = cur->link;
    }
```

```
}

template <class T>
int LinkedQueue<T>::Location()    //求汽车在便道的位置
{
    int locat = 0;
    LinkNode<T> *cur = front;
    while (cur != NULL)
    {
        locat++;
        cur = cur->link;
    }
    return locat;
}


class SeqStack
{
    public:
        int *hour;      //达到小时
        int *second;    //达到分钟
        string *name;   //车牌号
        int top;        //栈顶指针
        int maxSize;

    public:
        SeqStack(int sz = 50);          //建立一个空栈
        ~SeqStack()                     //析构函数
        {
            delete[] hour;
            delete[] second;
            delete[] name;
        }
        void Push(int Hour, int Second, string Name);      //插入
        void Pop(int& Hour, int& Second, string& Name);    //删除栈顶元素
        bool IsEmpty()const {return(top == -1) ? true : false;}   //判空
        bool IsFull()const {return (top == maxSize - 1) ? true : false;} //判满
        void Print();  //打印
};

SeqStack::SeqStack(int sz): top(-1), maxSize(sz)
{
    hour = new int[maxSize];               //创建栈的数组空间
```

```
    second = new int[maxSize];
    name = new string[maxSize];
}

void SeqStack::Push(int Hour, int Second, string Name)  //进栈
{
    hour[++top] = Hour;                    //栈顶指针先加一，再进栈
    second[top] = Second;
    name[top] = Name;
}

void SeqStack::Pop(int& Hour, int& Second, string& Name) //出栈
{
    Hour = hour[top];
    Second = second[top];
    Name = name[top];
    top--;
}

void SeqStack::Print()
{
    int num = top;
    if (IsEmpty())
    {
        cout << "停车场为空！！！" << endl;
        return;
    }
    cout << "现在停在停车场的汽车为：" << endl;
    while (num != -1)
    {
        cout << "车牌号为" << name[num] << "的汽车" << endl;
        num--;
    }
}



void Menu()
{
    int n;
    cout << "请输入停车场容量：";
    cin >> n;
    SeqStack Now(n);
    SeqStack Temp(n);
```

```
171.     LinkedQueue<string> Place;
172.     while(1)
173.     {
174.         int choice;
175.         cout << "*********停车场管理系统********" << endl;
176.         cout << "*          1、汽车到达          *" << endl;
177.         cout << "*          2、汽车离开          *" << endl;
178.         cout << "*3、输出停车场中的所有汽车牌号*" << endl;
179.         cout << "*4、输出候车场中的所有汽车牌号*" << endl;
180.         cout << "*        5、退出系统运行        *" << endl;
181.         cout << "*******************************" << endl;
182.         cout << "请输入要操作的选项：";
183.         cin >> choice;
184.         if (choice == 1)
185.         {
186.             int Hour, Second;
187.             string Car_Name;
188.             cout << "请依次输入汽车的车牌号  达到小时  达到分钟：";
189.             cin >> Car_Name >> Hour >> Second;
190.             if (Now.IsFull())
191.             {
192.                 Place.EnQueue(Car_Name);
193.                  cout << "车牌号为" << Car_Name << "的汽车在便道停车的位置为：
    " << Place.Location() << endl;
194.             }
195.             else
196.             {
197.                 Now.Push(Hour, Second, Car_Name);
198.                 cout << "车牌号为" << Car_Name << "的汽车在停车场停车的位置为：
    " <<  Now.top + 1 << endl;
199.             }
200.         }
201.         else if (choice == 2)
202.         {
203.             string left_Car, Name, get_Car;
204.              int left_hour, left_second, Hour, Second, get_hour, get_second;

205.             cout << "请输入要离开的汽车的车牌号  离开小时  离开分钟：";
206.             cin >> left_Car >> left_hour >> left_second;
207.             while (Now.top != -1)
208.             {
209.                 Now.Pop(Hour, Second, Name);
210.                 if (Name != left_Car)
211.                 {
```

```
                    Temp.Push(Hour, Second, Name);      //将车辆移到临时栈
                }
                else
                {
                     int Car_time = left_hour * 60 + left_second - Hour * 60 - Second;      //停留分钟
                     cout << "车牌号为：" << left_Car << "的汽车停车时长为：" << Car_time << "分钟，";
                     cout << "停车费为：" << ceil(Car_time / 60) * 3 << "元" << endl;
                    while (Temp.top != -1)      //将车辆移回停车场
                    {
                        Temp.Pop(get_hour, get_second, get_Car);
                        Now.Push(get_hour, get_second, get_Car);
                    }
                    if (!Place.IsEmpty())    //便道的车可以进入
                    {
                        string new_Car;
                        Place.DeQueue(new_Car);
                        Now.Push(left_hour, left_second, new_Car);
                    }
                    break;
                }
                if (Now.top == -1)    //找不到此车辆
                {
                    cout << "没有此车辆！！！" << endl;
                    while (Temp.top != -1)
                    {
                        Temp.Pop(get_hour, get_second, get_Car);
                        Now.Push(get_hour, get_second, get_Car);
                    }
                    break;
                }
            }
        }
        else if (choice == 3)
        {
            Now.Print();
        }
        else if (choice == 4)
        {
            Place.Display();
        }
        else break;
```

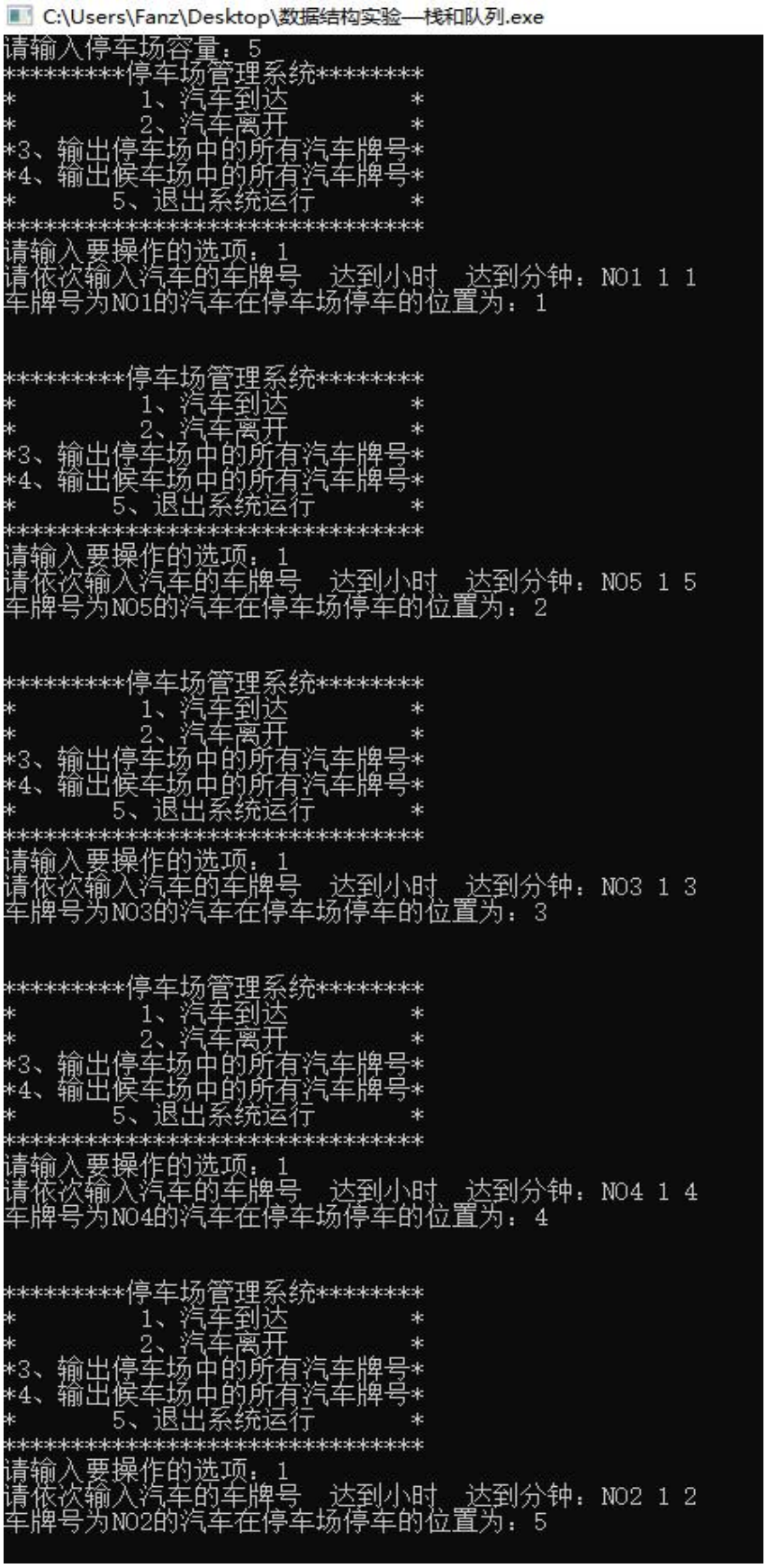
```
253.         cout << endl << endl;
254.     }
255. }
256.
257.
258. int main(int argc, char** argv)
259. {
260.     Menu();    //主菜单
261.     return 0;
262. }
```

五、算法描述或流程图

顺序栈:
插入函数是将汽车的达到小时、达到分钟、车牌号压入栈
出栈函数是将汽车移除
判满函数是判断栈是否满了，满则返回 True
判空函数是判断栈是否空了，空则返回 Ture
打印函数是依次遍历栈，输出在停车场的车的车牌号

链队：
插入函数是将汽车的车牌号压入队列
出队函数是将汽车移出便道
判满函数是判断栈是否满了，满则返回 True
汽车位置函数是返回汽车在便道停车的位置
打印函数是依次遍历队列，输出在便道的车的车牌号

六、实验数据和实验结果分析

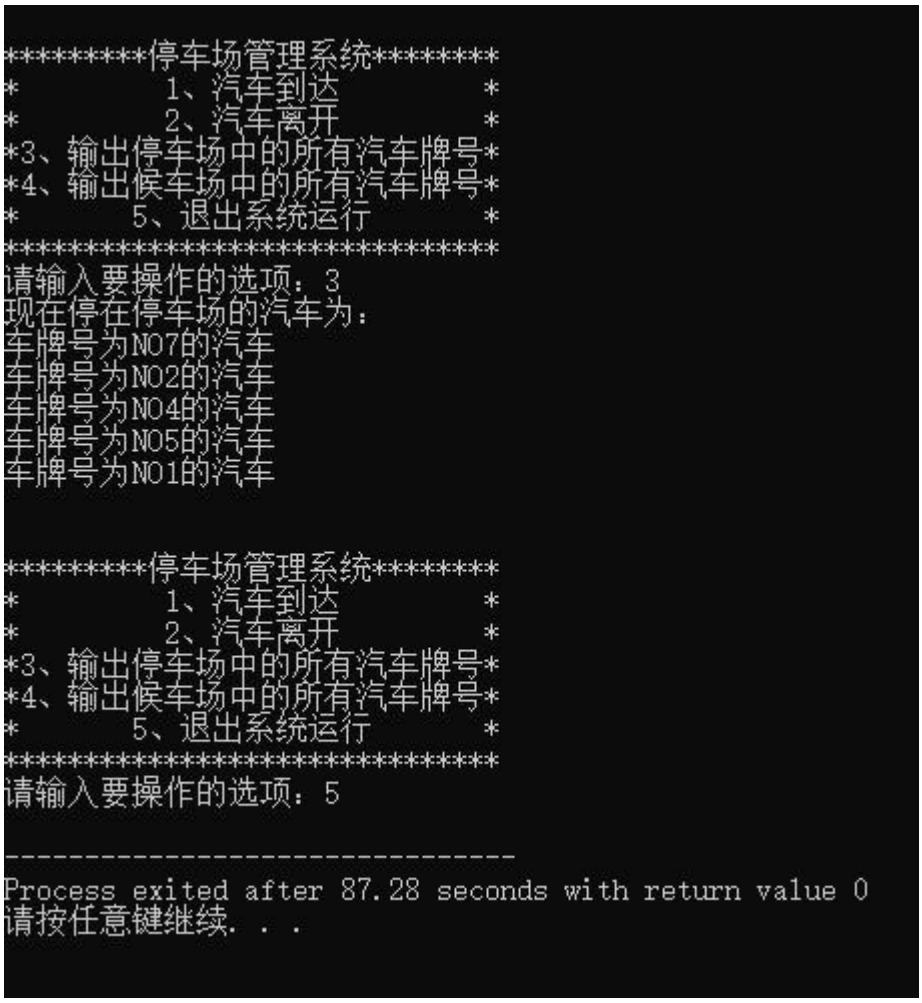


```
*********停车场管理系统********
*          1、汽车到达          *
*          2、汽车离开          *
*3、输出停车场中的所有汽车牌号*
*4、输出候车场中的所有汽车牌号*
*        5、退出系统运行        *
**********************************
请输入要操作的选项：3
现在停在停车场的汽车为：
车牌号为NO2的汽车
车牌号为NO4的汽车
车牌号为NO3的汽车
车牌号为NO5的汽车
车牌号为NO1的汽车


*********停车场管理系统********
*          1、汽车到达          *
*          2、汽车离开          *
*3、输出停车场中的所有汽车牌号*
*4、输出候车场中的所有汽车牌号*
*        5、退出系统运行        *
**********************************
请输入要操作的选项：4
便道上没有停汽车!


*********停车场管理系统********
*          1、汽车到达          *
*          2、汽车离开          *
*3、输出停车场中的所有汽车牌号*
*4、输出候车场中的所有汽车牌号*
*        5、退出系统运行        *
**********************************
请输入要操作的选项：1
请依次输入汽车的车牌号  达到小时  达到分钟：NO7 1 7
车牌号为NO7的汽车在便道停车的位置为：1


*********停车场管理系统********
*          1、汽车到达          *
*          2、汽车离开          *
*3、输出停车场中的所有汽车牌号*
*4、输出候车场中的所有汽车牌号*
*        5、退出系统运行        *
**********************************
请输入要操作的选项：2
请输入要离开的汽车的车牌号  离开小时  离开分钟：NO3 2 14
车牌号为：NO3的汽车停车时长为：71分钟，停车费为：6元
```



七、实验体会

这次实验，让我更加理解了栈和队列内部的操作，对栈和队列有了一个更深入的了解。