# 实验报告

题目：停车场管理系统

班级：2019211316

姓名：宿文宇

学号：2018211503

完成日期：2020.10.20

### 需求分析

1. 本演示程序中，输入形式为以“回车符”为结束标志的字符串，字符串不允许多余空格，大小写敏感。
2. 演示程序以用户和计算机的对话方式执行，即由用户在键盘上输入演示程序中规定的运算命令，相应的运算结果显示在其后。
3. 程序执行的命令包括：

‘A’(Arrival)：表示到达

‘D’(Departure)：表示离去

‘E’(End)：表示输入结束

### 概要设计

为实现上述程序功能，应以顺序结构实现双向栈模拟停车场，以链表结构实现队列模拟便道，为此，需要包含双向栈和顺序表的自定抽象数据结构。

class parkinglot

{

public:

    cars stack[MAX\_STACK\_SIZE\*2-1];

    int stack\_top;

    int stack2\_base;

    int stack2\_top;

    node\* queue\_start;

    node\* queue\_end;

int queue\_length;

    parkinglot()//初始化

    void arrive(int number, int time)//执行到达指令

    void depart(int number, int time)//执行离去指令

    void to\_stack2()//将数据转移到第二个栈

    void to\_stack()//将数据转移回第一个栈

};

本程序包含三个模块

1. 主程序模块：

Int main()

{

while(true)

{

读入指令；

判断指令（若是结束指令则跳出循环）；

执行指令；

}

}

1. 链表节点模块——以链表形式实现队列模拟便道
2. parkinglot模块——定义包含双向栈和队列的新的类模拟停车场

### 详细设计

class parkinglot

{

public:

    cars stack[MAX\_STACK\_SIZE\*2-1];

    int stack\_top;

    int stack2\_base;

    int stack2\_top;

    node\* queue\_start;

    node\* queue\_end;

    int queue\_length;

    parkinglot()

    {

        stack\_top = 0;

        stack2\_base = MAX\_STACK\_SIZE \* 2 - 2;

        stack2\_top = MAX\_STACK\_SIZE \* 2 - 2;

        queue\_start = new node;

        queue\_end = queue\_start;

        queue\_length = 0;

    }

    void arrive(int number, int time)

    {

        if(stack\_top >= MAX\_STACK\_SIZE)

        {

            node\* new\_node\_ptr = new node;

            new\_node\_ptr->car.number = number;

            queue\_end->next = new\_node\_ptr;

            queue\_end = new\_node\_ptr;

            cout << "Arrived in sideway:" << queue\_length + 1 << endl;

            queue\_length++;

        }

        else

        {

            stack[stack\_top].number = number;

            stack[stack\_top].parking\_time = time;

            cout << "Arrived in parkinglot:" << stack\_top + 1 << endl;

            stack\_top++;

        }

    }

    void depart(int number, int time)

    {

        for(int i = stack\_top - 1; i >= 0; i--)

        {

            if(stack[i].number != number)

                to\_stack2();

            else

            {

                cout << "Departed,you've stayed for " << time - stack[i].parking\_time << endl;

                stack\_top--;

                break;

            }

        }

        for(int i = stack2\_top + 1; i <= stack2\_base; i++)

            to\_stack();

        if(queue\_length > 0)

        {

            stack[stack\_top] = queue\_start->next->car;

            stack[stack\_top].parking\_time = time;

            node\* tmp = queue\_start;

            queue\_start = queue\_start->next;

            free(tmp);

            stack\_top++;

            queue\_length--;

        }

    }

    void to\_stack2()

    {

        if(stack\_top > stack2\_top || stack\_top <= 0)

            return;

        stack\_top--;

        stack[stack2\_top] = stack[stack\_top];

        stack2\_top--;

    }

    void to\_stack()

    {

        if(stack\_top > stack2\_top || stack2\_top >= stack2\_base)

            return;

        stack2\_top++;

        stack[stack\_top] = stack[stack2\_top];

        stack\_top++;

    }

};

### 调试分析

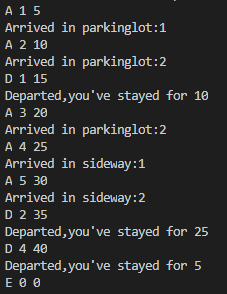
最开始时将双向栈大小设置为了2\*MaxStackSize+1，输出错误，经过gbd调试发现stack的大小超出了MaxStackSize，仔细分析可知，双向栈最多只需2\*MaxStackSize-1的空间，更正之后程序顺利运行

### 用户手册

1. 本程序运行环境为windows系统，执行文件为：2.ParkinglotManager.exe.
2. 进入演示程序后即可输入指令，输入完成后敲“回车键”完成操作。
3. 每次操作完成后，程序会输出信息。

### 测试结果

对所给样例的测试结果如下图：



### 附录

程序文件名：2.ParkingManager.cpp。