# 实验 停车场管理系统

【实验目的】

1. 掌握栈和队列的应用；
2. 掌握程序调试的基本方法和技巧；
3. 熟悉流程图及将流程图转换为程序代码的编程技能。

【实验要求】

1. 编码实现程序的功能并测试程序的有效性；
2. 按要求完成实验报告中的有关内容；
3. 实验报告文件命名规则：学号-姓名-实验报告一 停车场管理系统.docx。

【实验任务】

###### 一、数据结构设计

项目需要使用栈和队列两种数据结构。其中栈用于停车场和中转区，队列用于便道，对应的类名分别为：CarStack和CarQueue。CarStack为顺序栈，CarQueue为链队列，队列中的结点为CarNode类。Car类为车辆信息类，记录了车辆的车牌，到达时间和离开时间等信息。

设有车辆栈和队列，其定义如下：

#define MAX 3 // 停车场最大容量为3辆，便于观察

// 车辆信息

typedef struct {

char num[10]; // 车牌

long reach; // 车辆到达时间

long leave; // 车辆离开时间

}Car;

// 车辆顺序栈：停车场park和临时停车temp

typedef struct {

Car stack[MAX + 1];

int top;

int base;

}SeqStack;

// 便道停车wait队列的结点

struct CarNode {

Car data;

CarNode \*next;

};

// 便道停车链式队列

typedef struct {

CarNode \*head;

CarNode \*rear;

}LinkQueue;

**1、完成车辆栈CarStack的基本操作函数。其中：**

private static int MAXSIZE = 3; // 最大容量为3辆，便于观察

（1）public boolean push(Car car) // 车辆进栈

|  |
| --- |
| // 函数源码： |

（2）public Car pop() // 车辆出栈

|  |
| --- |
| // 函数源码： |

（3）public boolean isEmpty() // 判空

|  |
| --- |
| // 函数源码： |

（4）public boolean isFull() // 判满

|  |
| --- |
| // 函数源码： |

**2、完成车辆队列CarQueue的基本操作函数。其中：**

private CarNode front; // 队头

private CarNode rear; // 队尾

（1）public boolean isEmpty() // 车辆队列判空

|  |
| --- |
| // 函数源码： |

（2）public boolean offer(Car car) // 车辆入队

|  |
| --- |
| // 函数源码： |

（3）public Car poll() // 车辆出队

|  |
| --- |
| // 函数源码： |

###### 二、业务逻辑设计

**1、停车场业务功能为CarPark类。**在CarPark类中实现了汽车驶入、汽车驶离、费用计算和显示停车场车辆信息等功能。

|  |
| --- |
| ////////////////////////////////////////////////////  // 停车场业务  /\*\*  \* 功能： 将carNo车牌的汽车驶入，如果停车场有车位则进入停车场，设定入场时间，否则在便道等待进入  \* 参数：  \* carNo -- 车牌信息  \* 返回值：  \*/  public boolean arrival(String carNo) {  【代码1】 // 参考相关业务流程，完成该函数模块的编码任务。  }  /\*\*  \* 功能： 将carNo车牌的汽车驶离停车场，设定离开时间，同时便道汽车进入停车场  \* 参数：  \* carNo -- 车牌信息  \* 返回值：离开汽车  \*/  public Car leave(String carNo) {  【代码2】 // 参考相关业务流程，完成该函数模块的编码任务。  }  /\*\*  \* 功能： 根据车辆的出入时间，计算费用及停车时长  \* 参数：  \* car -- 车辆信息  \* 返回值：停车费用  \*/  public double charging(Car car) {  long offset = car.leave.getTime() - car.arrive.getTime();  return offset / 60000 \* price ;  }  // 显示所有入库车辆信息  public void showPark(){  【代码3】 // 参考相关业务流程，完成该函数模块的编码任务。  }  // 显示所有在便道上等待信息  public void showWaiting() {  【代码4】 // 参考相关业务流程，完成该函数模块的编码任务。  } |

**2、系统菜单及主类。**

**(1)系统菜单**

// 操作菜单

public static void menu() {

System.out.println("\n §※§※§※§※§※§ 欢迎使用停车场系统.§※§※§※§※§※§\t\n");

System.out.println("\t※◎※◎※◎※◎ 1. 车辆到达登记.※◎※◎※◎※◎\t");

System.out.println("\t※◎※◎※◎※◎ 2. 车辆离开登记.※◎※◎※◎※◎\t");

System.out.println("\t※◎※◎※◎※◎ 3. 显示车辆信息.※◎※◎※◎※◎\t");

System.out.println("\t※◎※◎※◎※◎ 4. 退出系统.※◎※◎※◎※◎\t");

System.out.println("\n\t请选择：\t");

}

**(2)主函数**

public static void main(String[] args) {

CarPark carPark = new CarPark();

Scanner scanner = new Scanner(System.in);

String carNo;

while(true) {

menu();

int item = -1;

while (true) {

item = scanner.nextInt();

if(item>0 && item <5)

break;

System.out.println("\n 输入有误，请重新选择： 1~4: ");

}

switch(item) {

case 1:

System.out.println("请输入车牌号：");

carNo = scanner.next();

carPark.arrival(carNo);

break;

case 2:

System.out.println("请输入车牌号：");

carNo = scanner.next();

Car car = carPark.leave(carNo);

long time = (car.leave.getTime() - car.arrive.getTime())/60000;

DecimalFormat df = new DecimalFormat("#.00");

String fee = df.format(carPark.charging(car));

System.out.println("车辆"+carNo+"停车时长"+time+"分钟，共收费"+fee+"元。");

break;

case 3:

carPark.showPark();

carPark.showWaiting();

break;

case 4:

System.exit(0);

}

}

}

###### 三、系统集成

###### 四、系统设计体会（不少100字）

|  |
| --- |
|  |