ШИНЖЛЭХ УХААН ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл холбооны технологийн сургууль

БИЕ ДААЛТЫН АЖЛЫН ТАЙЛАН

Өгөгдлийн бүтэц ба алгоритм (F.CSM203)
2025-2026 оны хичээлийн жилийн намар

Хичээл заасан багш: Д.Батмөнх
Бие даалтын ажил гүйцэтгэсэн: Т.Тэмүүлэн (B241960047)

Лабораторийн цаг: 1-1

Улаанбаатар хот
2025 он

АГУУЛГА

1. ОРШИЛ

Энэхүү систем нь шугаман жагсаалт болох (Stack) өгөгдлийн бүтцийг ашиглан машины зогсоол, орж гарч буй машинуудыг хянах зорилготой юм.

2. СИСТЕМИЙН БҮТЭЦ

Программ нь дараах үйлдлүүдийг хийх боломжтой:

- Орсон машинуудыг бүртгэх
- Машиныг гаргах
- Байгаа машинуудыг харах
- Машиныг зогсоолд байгаа эсэхийг дугаараар нь хайж шалгах

Оролтын файлууд:

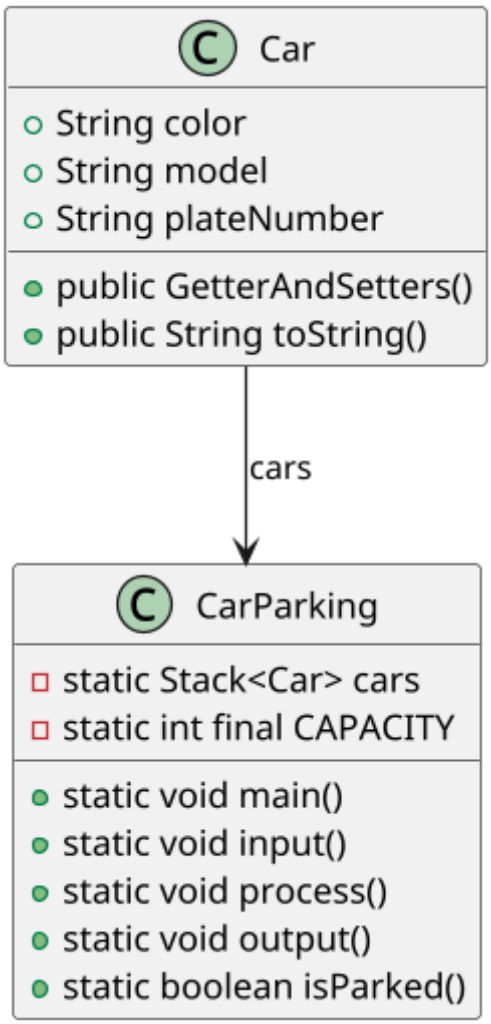
- cars.txt

3. АРХИТЕКТУР БОЛОН ДИЗАЙН

3.1 UML диаграмм

PlantUML ашиглан зурав.

3.2 Класс



Класс

- “Car” класс нь машиныг илтгэнэ. Үүнд машины дугаар, өнгө, загварыг хадгалах боломжтой.
- “CarParking” класс нь машины зогсоолын систем юм. Энэ класс нь машиныг оруулах, гаргах, бүртгэх, гэх мэт чухал үйлдлүүдийг хийнэ.

4. АЛГОРИТМ

4.1 Файл унших

```
public void input() {
    try (BufferedReader br = new BufferedReader(new FileReader("src/main/resources/cars.txt"))) {
        String line;
        while ((line = br.readLine()) != null) {
            String command = line.split("\\s+")[0];
            String plateNumber = line.split("\\s+")[1];
            process(command, plateNumber);
            output();
            System.out.println();
```

```
        }
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
```

“CarParking” классын “input” функц нь файлаас мэдээллийг уншсаны дараа зайгаар тусгаарлан салгаж “process” функцэд оролтын параметрийг онооно. Энэ кодыг бие даалтын даалгавраас авч өөрчлөн хэрэглэсэн.

4.2 “Process” функц

```
public void process(String command, String plateNumber) {
    if (command.equals("A")) {
        if (cars.size() < CAPACITY) {
            Car car = new Car(plateNumber);
            cars.add(car);
            System.out.printf("Arrival %s -> There is room.\n", plateNumber);
        } else {
            System.out.printf("Arrival %s -> Garage full, this car cannot enter.\n", plateNumber);
        }
    } else if (command.equals("D")) {
        if (isParked(plateNumber)) {
            Stack<Car> temp = new Stack();
            int counter = 0;
            for (int i = 0; i < CAPACITY; i++) {
                Car car = cars.pop();
                temp.push(car);
                counter++;
                if (car.getPlateNumber().equals(plateNumber)) {
                    temp.pop();
                    break;
                }
            }
            System.out.printf("Departure %s -> %d cars moved out.\n", plateNumber, counter);
            int loopSize = temp.size();
            for (int i = 0; i < loopSize; i++) {
                Car car = temp.pop();
                cars.push(car);
                System.out.printf("Arrival %s -> There is room.\n", car.getPlateNumber());
            }
        } else {
            System.out.printf("Departure %s -> This car not in the garage.\n", plateNumber);
        }
    }
}
```

"Process” функц нь зайгаа тооцоолж, боломжтой бол машиныг оруулна. Мөн машиныг гаргахдаа эхлээд байгаа эсэхийг нь шалгадаг. Хэрэв байгаа бол өөрийнх нь дараа орсон машинуудыг гаргасны дараа сүүлд нь орсон машинуудыг буцаагаад дарааллаар нь оруулдаг.

4.3 Машин зогсоолд байгаа эсэхийг шалгах

```
public boolean isParked(String plateNumber) {
    for (Car car : cars) {
        if (plateNumber.equals(car.getPlateNumber())) {
```

```
            return true;
        }
    }
    return false;
}
```

“CarParking” классад байх энэ функц нь дараах дугаартай машин зогсоолд байгаа эсэхийг шалгаж “үнэн“ эсвэл “худал” утга буцаана.

5. БИЧИЛ ШАЛГАЛТ

```
package org.example;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class CarParkingTest {
    @Test
    void testArrival() {
        CarParking carParking = new CarParking();
        carParking.process("A", "UB11-11");
        assertTrue(carParking.isParked("UB11-11"));
    }

    @Test
    void testGarageFull() {
        CarParking carParking = new CarParking();
        for (int i = 0; i < 10; i++)
            carParking.process("A", "CAR-" + i);

        carParking.process("A", "CAR-11");
        assertFalse(carParking.isParked("CAR-11"));
    }

    @Test
    void testDeparture() {
        CarParking carParking = new CarParking();
        carParking.process("D", "UB11-11");
        assertFalse(carParking.isParked("UB11-11"));
    }

    @Test
    void testDepartureFail() {
        CarParking carParking = new CarParking();
        carParking.process("D", "ZZ99-99");
        assertFalse(carParking.isParked("ZZ99-99"));
    }

    @Test
    void testIsParked() {
        CarParking carParking = new CarParking();
        carParking.process("A", "AA11-11");
        assertTrue(carParking.isParked("AA11-11"));
        assertFalse(carParking.isParked("BB22-22"));
    }

    @Test
```

```
    void testOutput() {
        CarParking carParking = new CarParking();
        carParking.process("A", "UB11-11");
        carParking.process("A", "UB22-22");
        carParking.output();
    }
}
```