(Лаборатори №4)

Н. Мөнхжин

ХШУИС, Компьютерийн ухааны оюутан, nyamkamunhjin@gmail.com

1. ОРШИЛ/УДИРТГАЛ

Програмчлалын хэлний ойлголт болох объект хандлагат технологийн битүүмжлэл, классын гишүүн өгөгдөл, гишүүн функцруу хэрхэн хандах талаар даалгаврыг хийж гүйцэтгэв.

2. ЗОРИЛГО

1. Үүсгэж тодорхойлсон классын гишүүн өгөгдөл, функцийг битүүмжлэх.
2. Private битүүмжлэлтэй гишүүн өгөгдөл рүү утга оноох, авах getter, setter функц тодорхойлох.

3. ОНОЛЫН СУДАЛГАА

3.1 Битүүмжлэл

Битүүмжлэл нь объектын өгөгдлийг бусдаас далдалж байдаг. Битүүмлэлд public, private, protected гэсэн төрлүүд байдаг.

- Public битүүмжлэлээр зарлагдсан гишүүн өгөгдөл, функцүүд нь программын хаанаас ч хандаж эрхтэй байдаг.

- Private битүүмжлэлээр зарлагдсан гишүүн өгөгдөл, функцүүд нь зөвхөн өөрийн класс дотроос хандах эрхтэй байдаг.

3.2 Getters, Setters

Классын private битүүмжлэлтэй гишүүн өгөгдөл, функцүүд рүү хандахын тулд getters, setters функцуудыг зарлаж өгдөг.

Getter, Setter функцууд нь public эрхтэй гишүүн өгөгдөлд утга оноох эсвэл авах функц байна. Getter, Setter ашиглахын давуу тал нь утга оноохдоо шалгуур тавьж өгөх боломжтой болгодог. Энэ нь гишүүн өгөгдөлд буруу утга оруулхаас сэргийлж найдвартай байдлыг хангадаг.

4. ХЭРЭГЖҮҮЛЭЛТ

Lab03-д тодорхойлсон классын захирлын цалин бодох функцийг private хандалтын түвшинтэй болгож өөрчил.

```
Захирлын цалинг олох private функц

private float bossSalaryCalculate() {
        if(this.occupation.toLowerCase().equals("boss"))
                return this.workedHours * SALARY_BOSS;
        else
                return 0;
}
```

Дараа нь цалин бодох функц дотор албан тушаал нь захирал байвал захирлын цалин бодох функцийг цалин бодох функц дотор дуудаж захирлын цалинг бодно. Энд гишүүн функц дотроос гишүүн функц дуудах үйлдлийг хийнэ.

```
public float calculateSalary() {
    float salary = 0;
    switch(this.occupation.toLowerCase()) {
      case "boss":
        salary = bossSalaryCalculate();
        break;
```

```
            case "manager":
                salary = this.workedHours * SALARY_MANAGER;
                break;
            case "employee":
                salary =  this.workedHours * SALARY_EMPLOYEE;
                break;
        }
        return salary;
    }
```

Мөн гишүүн өгөгдөл бүрийг private хандалтын түвшинтэй болгож лекц дээр заасан set болон get функц бичиж утга оноож, утгыг буцааж авна.

Өмнөх даалгавар дээрх Employee классын гишүүн өгөгдлүүдийг private хандалттай болгов.

```
    private int id;

    private String name;

    private String occupation;
    private float workedHours;
    public float getWorkedHours() {
        return workedHours;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getOccupation() {
        return occupation;
    }
    public void setOccupation(String occupation) {
        this.occupation = occupation;
```

```
    }
    public void setWorkedHours(float workedHours) {
        this.workedHours = workedHours;
    }
```

Ажилчин классын хүснэгт үүсгээд гараас хэд хэдэн ажилчны утга онооно.

```
        Employee sam = new Employee(123, "sam", "manager", 34);
        Employee mj = new Employee(119, "mj", "boss", 21);
        Employee badi = new Employee(111, "bad", "manager", 214);
        List<Employee> employees = new ArrayList<>();
        employees.add(sam);
        employees.add(mj);
        employees.add(badi);
```

Утга оноосон хүснэгтийг цалингаар нь эрэмбэлнэ.

```
    private static void sort(List<Employee> employees) {
        for(int i = 0; i < employees.size() - 1; i++){
            for(int j = i + 1; j < employees.size(); j++) {
                if(employees.get(i).getWorkedHours() > employees.get(j).getWorkedHours())
                    Collections.swap(employees, i, j);
            }
        }
    }
```

5. ДҮГНЭЛТ

Private хандалттай гишүүн өгөгдөлийг getters, setters функцуудтэй бичсэнээр шалгуур тавьж өгч буруу өгөгдөл орохоос хамгаалж программыг илүү найдвартай болгож байна.

6. АШИГЛАСАН МАТЕРИАЛ

1. Объект хандлагат технологийн C++ програмчлал, Ж.Пүрэв, 2008, Улаанбаатар.

7. ХАВСРАЛТ

```java
import java.util.*;

public class Main {

    public static void main(String[] args) {
     // write your code here
        Employee sam = new Employee(123, "sam", "manager", 34);
        Employee mj = new Employee(119, "mj", "boss", 21);
        Employee badi = new Employee(111, "bad", "manager", 50);
        List<Employee> employees = new ArrayList<>();
        employees.add(sam);
        employees.add(mj);
        employees.add(badi);
        sort(employees);
        for(int i = 0; i < employees.size(); i++) {
            System.out.println(employees.get(i).getWorkedHours());
        }



    }
    private static void sort(List<Employee> employees) {
        for(int i = 0; i < employees.size() - 1; i++){
            for(int j = i + 1; j < employees.size(); j++) {
```

```
                                if(employees.get(i).getWorkedHours()  >
employees.get(j).getWorkedHours())

                    Collections.swap(employees, i, j);

            }

        }

    }

}

class Employee {

    // fields

    private int id;

    private String name;

    private String occupation;
```

```
    private float workedHours;

    //salaries for employees

    private final float SALARY_BOSS = 30;

    private final float SALARY_MANAGER = 17;

    private final float SALARY_EMPLOYEE = 10;

        Employee(int id, String name, String occupation, float
workedHours) {

        this.id = id;

        this.name = name;

        this.occupation = occupation;

        this.workedHours = workedHours;

    }




    public void inputInfo() {


        System.out.print("Enter id: ");

        this.id = (new Scanner(System.in)).nextInt();


        System.out.print("\nEnter name: ");

        this.name = (new Scanner(System.in)).next();
```

```
        System.out.print("\nEnter occupation: ");
        this.occupation = (new Scanner(System.in)).next();

        System.out.print("\nEnter workedHours: ");
        this.workedHours = (new Scanner(System.in)).nextFloat();
    }

    public void showInfo() {
        System.out.println("id:" + this.id);
        System.out.println("name: " + this.name);
        System.out.println("occupation:" + this.occupation);
        System.out.println("worked hours: " + this.workedHours);
    }

    public float calculateSalary() {
        float salary = 0;
        switch(this.occupation.toLowerCase()) {
            case "boss":
                salary = bossSalaryCalculate();
                break;
```

```
            case "manager":
                salary = this.workedHours * SALARY_MANAGER;
                break;

            case "employee":
                salary =  this.workedHours * SALARY_EMPLOYEE;
                break;
        }


        return salary;
    }

    private float bossSalaryCalculate() {
        if(this.occupation.toLowerCase().equals("boss"))
            return this.workedHours * SALARY_BOSS;
        else
            return 0;
    }
```

```
public boolean addWorkHours(float hours) {
    if (hours >= 24)
        return false;

    if(this.occupation.toLowerCase().equals("employee")
        || this.occupation.toLowerCase().equals("boss")
        || this.occupation.toLowerCase().equals("manager")) {

        this.workedHours += hours;

        return true;
    } else
        return false;


}

public float getWorkedHours() {
    return workedHours;
}
public int getId() {
```

```
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getOccupation() {
        return occupation;
    }
    public void setOccupation(String occupation) {
        this.occupation = occupation;
    }
    public void setWorkedHours(float workedHours) {
        this.workedHours = workedHours;
    }

}
```