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
2024 он

АГУУЛГА

1. ОРШИЛ 2
2. СИСТЕМИЙН БҮТЭЦ.....2
3. АРХИТЕКТУР БОЛОН ДИЗАЙН.....2
3.1 UML диаграмм 2

3.2 Класс....3
4. АЛГОРИТМ....4
4.1 Файл унших....4
4.2 GPA тооцоолох....4
4.3 Унасан оюутнуудын жагсаалт....5
5. БИЧИЛ ШАЛГАЛТ....6

1. ОРШИЛ

Энэхүү систем нь шугаман жагсаалт болох (ArrayLinearList, Chain) өгөгдлийн бүтцийг ашиглан оюутнуудыг хичээл, мэргэжил, шалгалтын дүн зэргээр нь ангилан хадгална.

2. СИСТЕМИЙН БҮТЭЦ

Программ нь дараах үйлдлүүдийг хийх боломжтой:

- Хичээлүүдийн жагсаалтыг харуулах
- Мэргэжлүүдийн жагсаалтыг харуулах
- Сурагчийг кодоор нь хайх
- Оюутнуудын жагсаалтыг харуулах
- Оюутны дундаж “GPA” тооцох
- Нийт оюутны дундаж “GPA” тооцох
- Гурваас дээш “F” үнэлгээтэй оюутнуудыг харуулах
- Хичээл бүрээр дүнгийн жагсаалт харуулах
- Мэргэжил бүрээр дүнгийн жагсаалт харуулах
- Унасан сурагчдийг харуулах

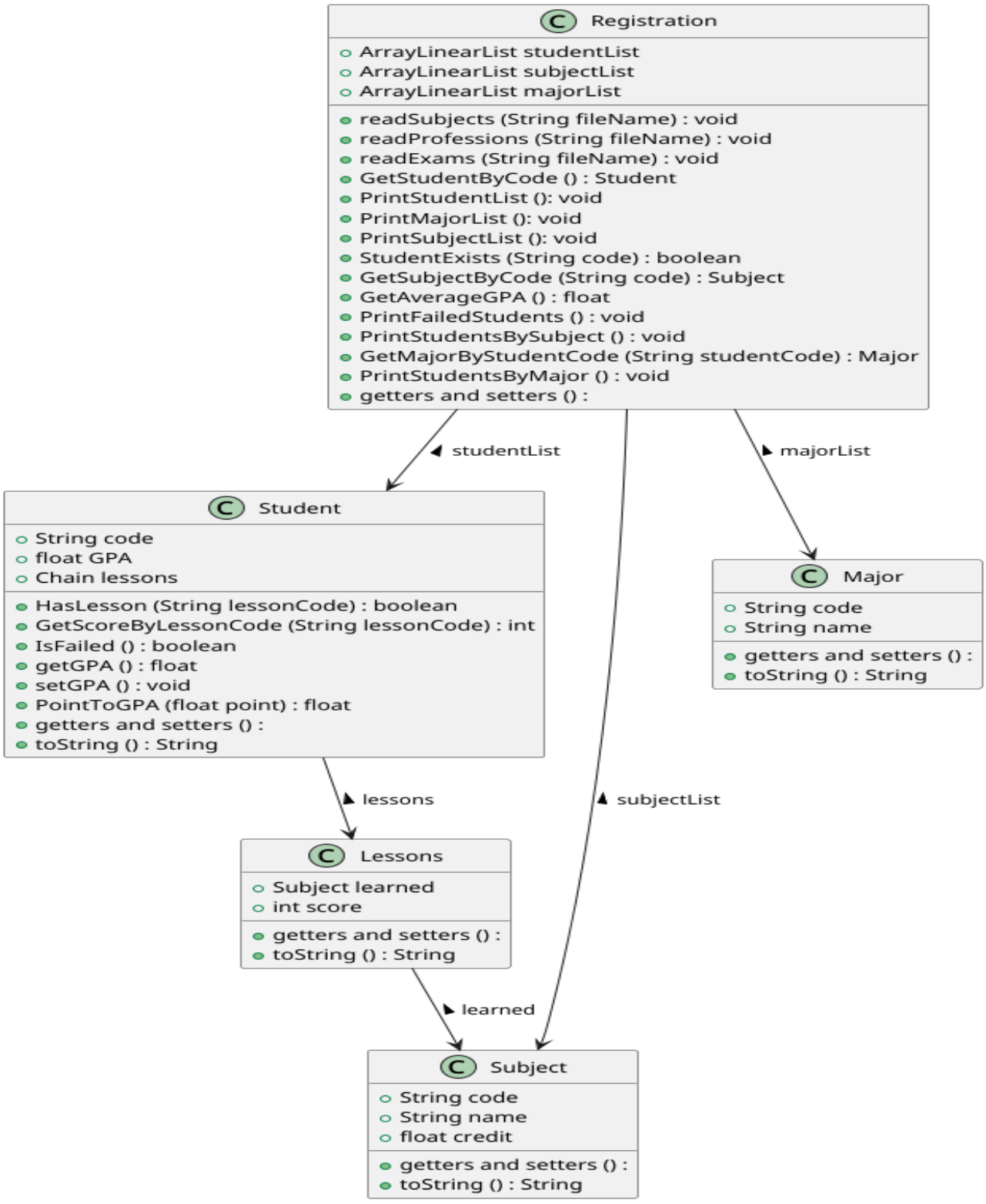
Оролтын файлууд:

- Subjects.txt
- Professtions.txt
- Exams.txt

3. АРХИТЕКТУР БОЛОН ДИЗАЙН

3.1 UML диаграмм

PlantUML ашиглан зурав.



3.2 Класс

- “Registration” класс нь гурван ArraylinearList агуулах ба “Subject”, “Lessons”, “Student” төрөлтэй классууд элемент нь болно.

- “Student” класс нь сурагчийн кодыг “String” төрлөөр хадгална. Харин голч оноог “float” төрлөөр, үзсэн хичээлүүдийг (Lessons) “Chain” шугаман жагсаалтын элемент болгож хадгалагдана.
- “Lessons” класс нь сурагчийн аль хэдийн судалсан хичээлүүдийг хадгалах зориулалттай. Үүнд оноо нь “int” төрөл, хичээл нь “Subject” төрөл байна.
- “Subject” класс нь хичээлийг хадгалах зориулалттай. Код болон нэр нь “String” төрөл, кредит нь “float” төрөл байна.
- “Major” класс нь оюутнуудыг мэргэжлээр нь ялгахад хэрэгтэй. Сурагчийн эхний 2 үсэг мэргэжлийг нь илтгэнэ. Код болон нэр талбарууд нь “String” төрөл байна.

4. АЛГОРИТМ

4.1 Файл унших

```
public void readSubjects(String fileName) {
   try {
      BufferedReader input = new BufferedReader(new FileReader(fileName));
      String line;

      while ((line = input.readLine()) != null) {
         String[] values = line.split("/");
         Subject subject = new Subject();
         subject.setCode(values[0]);
         subject.setName(values[1]);
         subject.setCredit(Float.parseFloat(values[2]));
         this.subjectList.add(this.subjectList.size(), subject);
      }
   } catch (Exception e) {
      throw new ExceptionInInitializerError(e);
   }
}
```

“Registration” классын “readSubjects” функц нь файлыг уншиж мэдээллийг хадгалах зориулалттай. Энэ кодыг бие даалтын даалгавраас авч өөрчлөн хэрэглэсэн. Файлаас мэдээллээ уншиж авсны дараа “String” төрөлтэй “Array” дотор индексжүүлж хийсэн. Энэ нь бусад өгөгдлийн бүтцээс хялбар бичиглэлтэй учраас сонгосон.

4.2 GPA тооцоолох

```
public float getGPA() {
   this.setGPA();
   return GPA;
}

public void setGPA() {
   float sumScore = 0;
   float sumCredit = 0;
   for (int i = 0; i < this.lessons.size(); i++) {
      Lessons lesson = (Lessons) this.lessons.get(i);
```

```
        sumScore += lesson.getScore() * lesson.getLearned().getCredit();
        sumCredit += lesson.getLearned().getCredit();
    }
    float avgScore = sumScore / sumCredit;
    this.GPA = PointToGPA(avgScore);
}

private static float PointToGPA(float point) {
    if (point >= 93 && point <= 100) {
        return 4.0F;
    } else if (point >= 90F && point <= 92F) {
        return 3.7F;
    } else if (point >= 87F && point <= 89F) {
        return 3.3F;
    } else if (point >= 83F && point <= 86F) {
        return 3F;
    } else if (point >= 80F && point <= 82F) {
        return 2.7F;
    } else if (point >= 73F && point <= 76F) {
        return 2F;
    } else if (point >= 67F && point <= 69F) {
        return 1.3F;
    } else if (point >= 65F && point <= 66F) {
        return 1F;
    } else {
        return 0F;
    }
}
```

Оноог голч болгохын тулд 2 функц ашигласан.

1. “PointToGPA” оноог голч дүн болгож хөрвүүлээд буцаана.
2. “setGPA” энэ нь “getGPA” ажиллах бүрд дуудагдаж ажиллаж байгаа нь шинэ хичээлийн оноо бодогдоогүй байх эрсдэлээс сэргийлнэ. Функц нь товчхондоо тухайн оюутны бүх судалсан хичээлийн голч дүнг тооцоолж хадгалдаг.

4.3 Унасан оюутнуудын жагсаалт

```
public boolean IsFailed() {
    int counter = 0;
    for (int i = 0; i < this.lessons.size(); i++) {
        Lessons lesson = (Lessons) this.lessons.get(i);
        if (PointToGPA(lesson.getScore()) == 0) {
            counter++;
        }
    }
    return counter >= 3;
}
```

“Student” классад байх энэ функц нь гурваас дээш хичээл дээр унасан эсэхийг шалгана. Ингэснээр “Registration” классад бичиглэл илүү хялбар болно.

```
public void PrintFailedStudents() {
    System.out.println("_____FAILED STUDENTS_____");
    for (int i = 0; i < this.studentList.size(); i++) {
        Student student = (Student) this.studentList.get(i);
        if (student.IsFailed()) {
            System.out.println(student);
```

```
        }
    }
    System.out.println();
}
```

“Registration” бүх сурагчаар давталт гүйлгэж унасан сурагчдыг хэвлэнэ.

5. БИЧИЛ ШАЛГАЛТ

```
package org.example;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class RegistrationTest {
    private Registration registration;
    private Student student1, student2;
    private Subject subject1, subject2;
    private Major major1, major2;
    private Lessons lesson1, lesson2, lesson3, lesson4;

    @BeforeEach
    void setUp() {
        registration = new Registration();

        major1 = new Major("AI", "Hiimel Oyun Uhaan");
        major2 = new Major("UH", "Utasgui Holboo");

        subject1 = new Subject("CS101", "Programmchlal undes", 3);
        subject2 = new Subject("CS102", "Tuuh", 4);

        lesson1 = new Lessons();
        lesson1.setLearned(subject1);
        lesson1.setScore(85);

        lesson2 = new Lessons();
        lesson2.setLearned(subject2);
        lesson2.setScore(60);

        lesson3 = new Lessons();
        lesson3.setLearned(subject1);
        lesson3.setScore(20);

        lesson4 = new Lessons();
        lesson4.setLearned(subject2);
        lesson4.setScore(45);

        student1 = new Student("AI12345");
        student1.lessons.add(student1.lessons.size(), lesson3);
        student1.lessons.add(student1.lessons.size(), lesson4);
        student1.lessons.add(student1.lessons.size(), lesson2);

        student2 = new Student("UH67890");
        Lessons lesson3 = new Lessons();
        lesson3.setLearned(subject1);
        lesson3.setScore(95);
```

```
        student2.lessons.add(student2.lessons.size(), lesson3);

        registration.majorList.add(registration.majorList.size(), major1);
        registration.majorList.add(registration.majorList.size(), major2);

        registration.subjectList.add(registration.subjectList.size(), subject1);
        registration.subjectList.add(registration.subjectList.size(), subject2);

        registration.studentList.add(registration.studentList.size(), student1);
        registration.studentList.add(registration.studentList.size(), student2);
    }

    @Test
    void testGetAverageGPA() {
        float avg = registration.GetAverageGPA();
        assertEquals(2.0f, avg, 0.1f);
    }
}
```