Курсов проект по ООП

Боян Иванов Дончев

фн: 72029

Софийски университет „Св. Кл. Охридски“

Факултет по математика и информатика

Информационни системи

I курс

Проект №13: СУСИ

*Да се напише програма, реализираща информационна система за обслужване на студенти. Програмата съхранява и обработва необходимите данни във файл.*

Юни, 2021

София

Съдържание

[Глава 1. Увод 3](#_Toc74457130)

[1.1. Описание и идея на проекта 3](#_Toc74457131)

[1.2. Цел и задачи на разработката 3](#_Toc74457132)

[Глава 2. Преглед на предметната област 4](#_Toc74457133)

[2.1. Основни дефиниции, концепции и алгоритми, които ще бъдат използвани 4](#_Toc74457134)

[2.2. Дефиниране на проблеми и сложност на поставената задача 4](#_Toc74457135)

[2.3. Подходи, методи за решаване на поставените проблемите 4](#_Toc74457136)

[Глава 3. Проектиране 7](#_Toc74457137)

[3.1. Обща архитектура – ООП дизайн 7](#_Toc74457138)

[Глава 4. Реализация, тестване 8](#_Toc74457139)

[4.1. Реализация на класове 8](#_Toc74457140)

[4.2. Планиране, описание и създаване на тестови сценарии (създаване на примери) 8](#_Toc74457141)

[Глава 5. Заключение 10](#_Toc74457142)

[5.1. Насоки за бъдещо развитие и усъвършенстване 10](#_Toc74457143)

[Използвана литература 11](#_Toc74457144)

Глава 1. Увод

## Описание и идея на проекта

Идеята на проекта СУСИ, е да се състави информационна система за студентите и дейностите в един университет. Целта е да се улесни достъпът и редакцията на информацията.

Приложението трябва да съхранява данни свързани със студентите (име, факултетен номер, специалност, записани дисциплини и оценки), специалностите (име на специалността и дисциплините които са част от тази програма) и дисциплините (име на дисциплината, задължителна или не и курс, в който може да бъде записана) в рамките на един университет, позволявайки бърз достъп и улеснена работа с тях (преместване на студент в друга специалност, добавяне на оценки и тн.)

## Цел и задачи на разработката

Основната цел на проекта е улеснена работа. Трябва да е по-интуитивно и по-бързо, от който и да е друг начин за съхранени и обработка на информация. Това е смисълът на тази информационна система, да облекчи административната работа, и да доведе до по-добри резултати – без грешки.

Една от задачите, е интерфейсът да не позволява на потребителя да допусне грешка, която ще доведе до прекъсване на програмата. Трябва да бъде помислено за всички възможни действия от страна на клиента, и входните данни да бъдат валидирани преди обработка.

Глава 2. Преглед на предметната област

## 2.1. Основни дефиниции, концепции и алгоритми, които ще бъдат използвани

Основното звено на програмата е класът Controller, той е връзката между потребителя и останалите класове в приложението, както и логиката им. Задачата на класа контролер е да приеме входните данни, и да разпредели задачите, които трябва да бъдат изпълнени, по другите класове, и да върне отговор на клиента.

Има и функционалности, която се изпълняват и директно от класа контролер, например четенето и записването във файлове, където се пази нужната информация.

Един от най-важните алгоритми, изпълняващ се при всяка заявка на потребителя е разпознаването на входните данни и разграничаването на команди от параметри. Това е фундаментално за работата на цялата програма.

## 2.2. Дефиниране на проблеми и сложност на поставената задача

Алгоритъмът, споменат в предната точка, се справя с една от най-сложните предизвикателства в този проект. Друг проблем изискващ повече работа и мисъл, за решението му, е валидацията на данните. Различните стойности са от различен тип, и трябва да се уверим, че когато искаме да работим с числа, потребителят е въвел число, което води и до последния проблем от категорията на проблеми с по-висока сложност: да превърнем въведените, под формата на текст числа, в числа с които могат да се извършват математически операции.

## 2.3. Подходи, методи за решаване на поставените проблемите

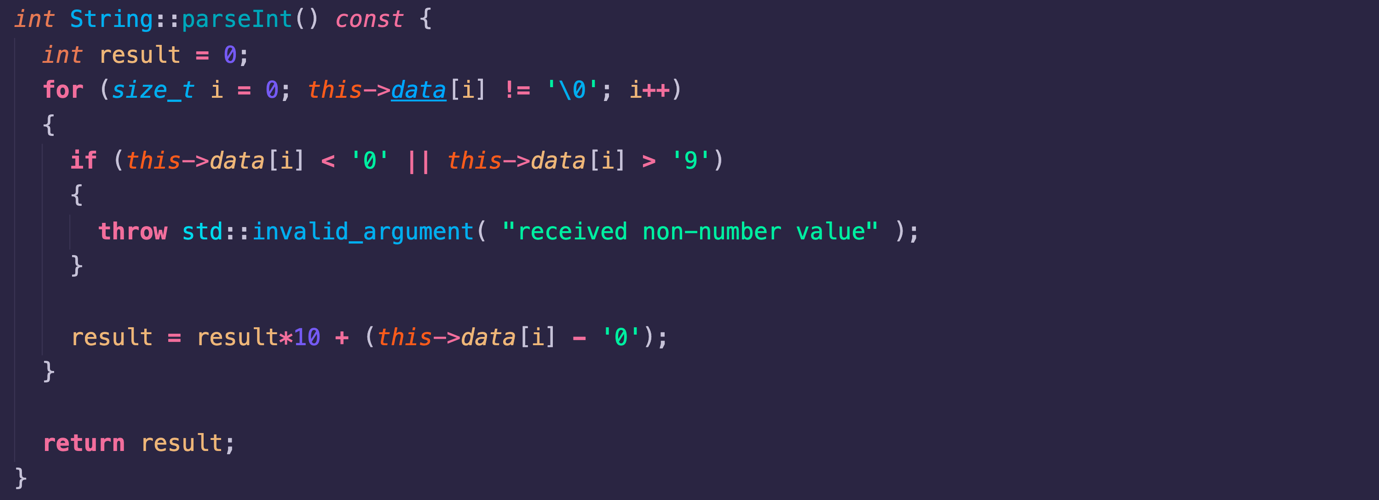
И трите споменати проблема в 2.2., имат елегантни решения, които не изискват големи и дълги алгоритми за решението им.

Един начин, по който може да бъде заобиколен проблема с разграничаването на входните данни, е да разделим дадения символен низ на, съставящите го, думи.



След тази кратка стъпка, можем да приемем, че първата дума е име на команда, а всички следващи са нейни параметри.

Другите два проблема, могат да се разгледат, и като един общ проблем, решението, на който е просто, но затормозяващо за написване.



Минавайки през всеки символ от низа, оценяваме дали символът има числена стойност, и ако има добавяме в края на резултата, използвайки математическа формула. По-интересно е обаче, ако искаме да работим с не-целочислени стойности, тогава алгоритъмът е по-дълъг, но идеята е същата. 

Глава 3. Проектиране

## 3.1. Обща архитектура – ООП дизайн

Обектно ориентираният подход е най подходящ, за този проект. Класовете от които имаме нужда са: Student, Program, Discipline и Record.







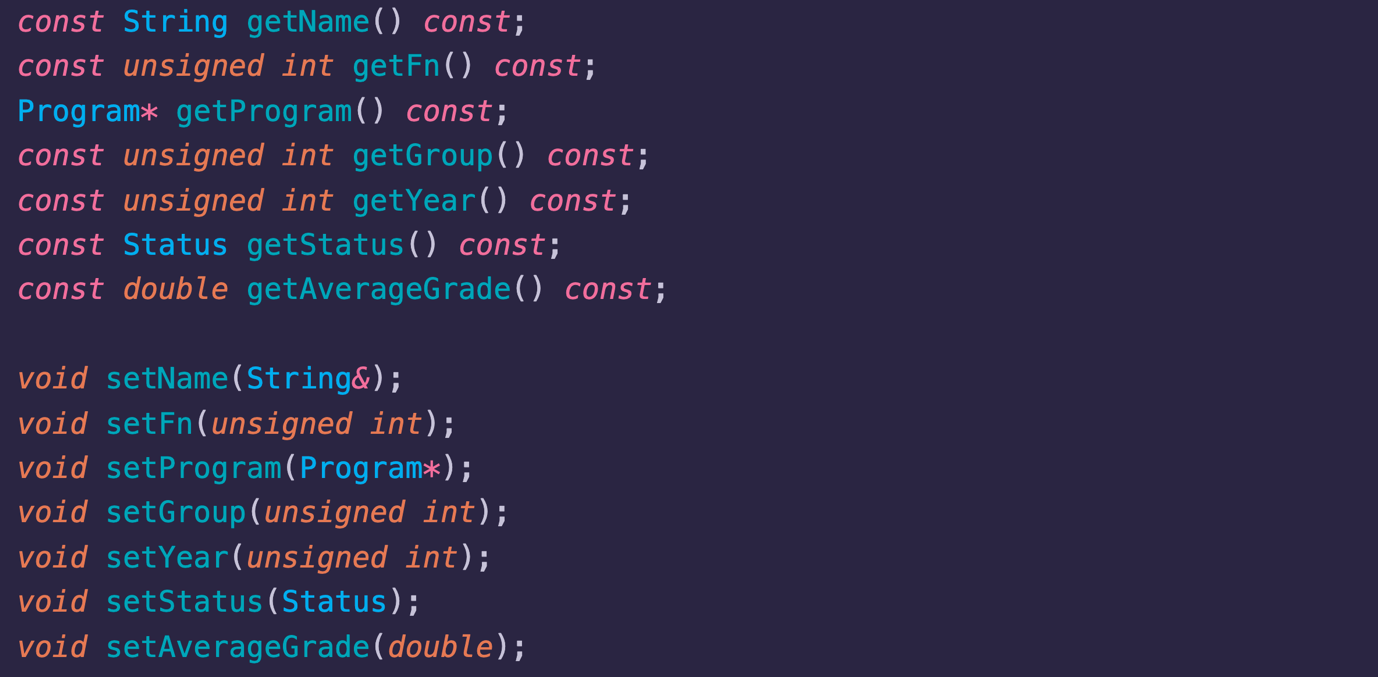


Record е класът, който представлява оценка по даден предмет, на даден студент. Когато студент се запише за курс, отново ще използваме Record като в паметта заделена от член-данната grade, ще запишем 0. По този начин отбелязваме студент записан в дисциплина, но не-положил изпит.

Глава 4. Реализация, тестване

## 4.1. Реализация на класове

Реализацията на класовете е проста, включва най-вече енкапсулация (getters & setters).



Най-интересните неща се случват в контролера, там са реализирани по сложните операции.

## 4.2. Планиране, описание и създаване на тестови сценарии (създаване на примери)

Тестовете са от огромно значение за един проект. Добра практика е да се напишат в началото, преди да се напише логиката на приложението. По този начин, когато се правят промени е лесно да се провери, дали след редакцията всичко работи, както се очаква. Тестовете правят, процеса значително по-лесен.

При създаването на тестове трябва да се помисли за възможно най-много примери и edge case-ове, за да е сигурно, че всичко работи както трбва.

Изключително важно е, тестовете да са написани правилно, защото иначе е възможно да си мислим че имаме грешка в кода, а всъщност самият тест е написан грешно. Времето, което може да се изгуби в търсене на грешка, която не съществува, трябва да се избегне.

Глава 5. Заключение

## 5.1. Насоки за бъдещо развитие и усъвършенстване

В бъдеще, е възможно да се добави, функционалност, отговаряща за съхранението на информация за преподаватели и други длъжностни лица в университета.

Използвана литература

-