ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА

Факултет по изчислителна техника и автоматизация

Катедра „Компютърни Науки и Технологии“

**Семестриална Домашна Работа**

по дисциплината „Базово програмиране”

на тема:„ Информационна система”

Вариант № 321

|  |  |
| --- | --- |
| Изготвил: Раффи Никогосян | Проверил: |
| Специалност: ИИ |  |
| Група: 1 |  |
| Факултетен номер: 23621923 |  |

2024

Съдържание

[I. Задание на проекта 3](#_Toc154834477)

[II. Анализ на решението 5](#_Toc154834478)

[1. Структура за данните в програмата 5](#_Toc154834479)

[2. Реализация на условие A 5](#_Toc154834480)

[3. Реализация на условие B 5](#_Toc154834481)

[4. Реализация на условие C 6](#_Toc154834482)

[5. Реализация на условие D 6](#_Toc154834483)

[6. Реализация на условие E 7](#_Toc154834484)

[7. Реализация на условие F 7](#_Toc154834485)

[8. Реализация на условие … - допълнение първо 8](#_Toc154834486)

[9. Реализация на условие … - допълнение второ 8](#_Toc154834487)

[10. Реализация на допълнение трето 9](#_Toc154834488)

[III. Упътване за употреба 9](#_Toc154834489)

[1. Впишете съответната част от проекта 10](#_Toc154834490)

[2. Впишете съответната част от проекта 10](#_Toc154834491)

[IV. Примерно действие на програмата 11](#_Toc154834492)

[1. Условие A 11](#_Toc154834493)

[2. Условие B 11](#_Toc154834494)

[3. Условие C 11](#_Toc154834495)

[4. Условие D 11](#_Toc154834496)

[5. Условие E 11](#_Toc154834497)

[6. Условие F 12](#_Toc154834498)

[7. Допълнение първо 12](#_Toc154834499)

[8. Допълнение второ 12](#_Toc154834500)

[9. Допълнение трето 12](#_Toc154834501)

Задание на проекта

**Информационна система за Студенти**

Да се напише компютърна програма, реализираща информационна система, която

поддържа студентска група (факултетен номер, ЕГН, три имена, списък с дисциплини, пол,

възраст, статут-активен/прекъснал/завършил). Програмата съхранява и обработва данни за

студентите в групата. Максималния брой студенти в една група е 30. Всеки студент има 5

оценки по дисциплините (име, оценка) от 1 семестър

**Базова задача**

1. Меню за избор на функциите от програмата.

**Функции от програмата са:**

1. Добавяне на студенти в група:
   1. Добавяне на студенти в групата, чрез добавяне в масива, като не трябва да превишава максималният брой студенти в една група(30).

Пример: Добавяне на списък от студенти. Въвежда се цяло число **n** и след него **n** на брой студенти. **N** не може да надвишава свободните елементи в масива със студенти. **Валидни оценки са 0 – няма оценка, 2, 3, 4, 5, 6,**

1. Извеждане на всички студенти на екрана
   1. Извеждане на всички студенти, чрез подходящо форматиране в таблица
2. Търсене и извеждане на екрана:
   1. Търсене и извеждане на студенти с оценка по БП в определен в интервал [минимален-максимален]
   2. Търсене и извеждане на студенти с минимум един предмет, с оценка Слаб 2
3. Сортиране и извеждане на студенти на екран:
   1. Сортиране на студентите по малко име по азбучен ред.
4. Работа с външен файл(двоичен или текстов):
   1. Извеждане на масива със студенти във файл.
   2. Въвеждане на масива от файл.

**Допълнение първо (+ базова задача):**

1. Да се създаде подменю към основното с нови функции за:
   1. Отделяне на студенти с определен статут (въвежда се от потребителят) и сортиране по факултетен номер.
   2. Отделяне на студенти с успех в интервал [минимален-максимален] и сортиране по ЕГН.

**Допълнение второ (+ базова задача):**

1. Въвеждане/Актуализация на оценки и среден успех за студент
   1. Добавяне в структурата на поле среден успех.
   2. Въвеждане/Актуализацията на данни (оценки и среден успех), става по въведен факултетен номер на студент.
      1. Ако студентът присъства в списъка(масива):
         1. Проверява се статута му:
            1. Активен – при въвеждане/актуализация на оценки, се изчислява автоматично средният успех. Всяка промяна на оценка изисква запитване за промяна! Ако няма оценка, тя се актуализира без запитване!
   3. Прекъснал/завършил – НЕ могат да се актуализират оценки c. Ако студента не присъства в списъка(масива), се извежда пояснително съобщение на екрана.
2. Смяна на статут на студент
   1. Въвежда се факултетният номер на студента(ако го има, се въвежда и новият статут). Ако статутът е активен или прекъснал се сменя с новият. Ако статутът е завършил НЕ СЕ допуска промяна

**Допълнение трето (+ базова задача) :**

1. Данните в програмата да се попълват автоматично от файл при стартиране и да се записват автоматично във файл при затваряне на програмата.

Анализ на решението

Структура за данните в програмата:

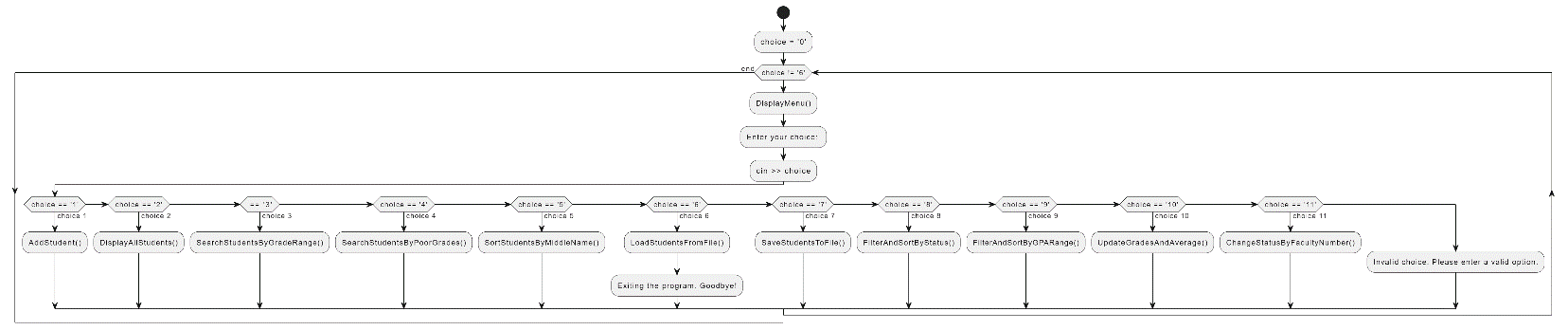
* Данните биват съхранение в две структури Students и Disciplines:

|  |  |  |
| --- | --- | --- |
| **Структура** | **Обяснение** | **Примерени стойности** |
| struct Students {  int facultyNumber =0;  int socialSecurityNumber =0;  string firstName;  string lastName;  string gender;  int age = 0;  string status;  float gPA = 0.00;  Disciplines disciplines[5];  };  struct Disciplines {  string name;  int grade = 0;  }; | Съхранява ф.№  Съхранява ЕГН  Име  Фамилия  Пол  Статус  Среден успех  Пет дисциплини  Име на дисциплина  оценка | 12342  04520202  Раффи  Никогосян  Мъж  Завършил  5.47  Математика , Физика…  4, 5….. |

Меню за избор на функциите от програмата:

### Анализ на алгоритъма:

Предоставя на потребителя да избира между различните функции в системата за студенти.

Блок схема на алгоритъма:

Функция, с която е реализиран алгоритъма:

void DisplayMenu(); int main();

#### Входни данни на функцията:

Функцията не приема входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат:

Извежда меню с функции

Търсене и извеждане на екрана по определен критерий:

Условие D, търси по определени критерии из потребителските данни , сортира ги и извежда на екрана :

Анализ на алгоритъма:

#### Търсене и извеждане на екрана студенти , чиято оценка по базово програмиране е в интервал въведен от потребителят:

Реализира се , чрез сортиране на всички оценки по БП според минимална оценка и максимална такава.

#### Търсене и извеждане на екрана студенти, с минимум един предмет оценен със Слаб 2

Реализира се, чрез сортиране на всички студенти които имат оценка оценена с 2.

Блок схема на алгоритъма:

#### Блок схема на алгоритъма:

#### Блок схема на алгоритъма:

Функция с която е реализиран алгоритъма:

#### void SearchStudentsByGradeRange();

#### void SearchStudentsByPoorGrades();

### Входни данни на функцията:

И двете функции използват входът от потребителя за критерий по който да търсят/сортират, като не използват входни данни, а глобано записаните данни в структурата **Students.**

#### Изходни данни на функцията или данни, които се извеждат

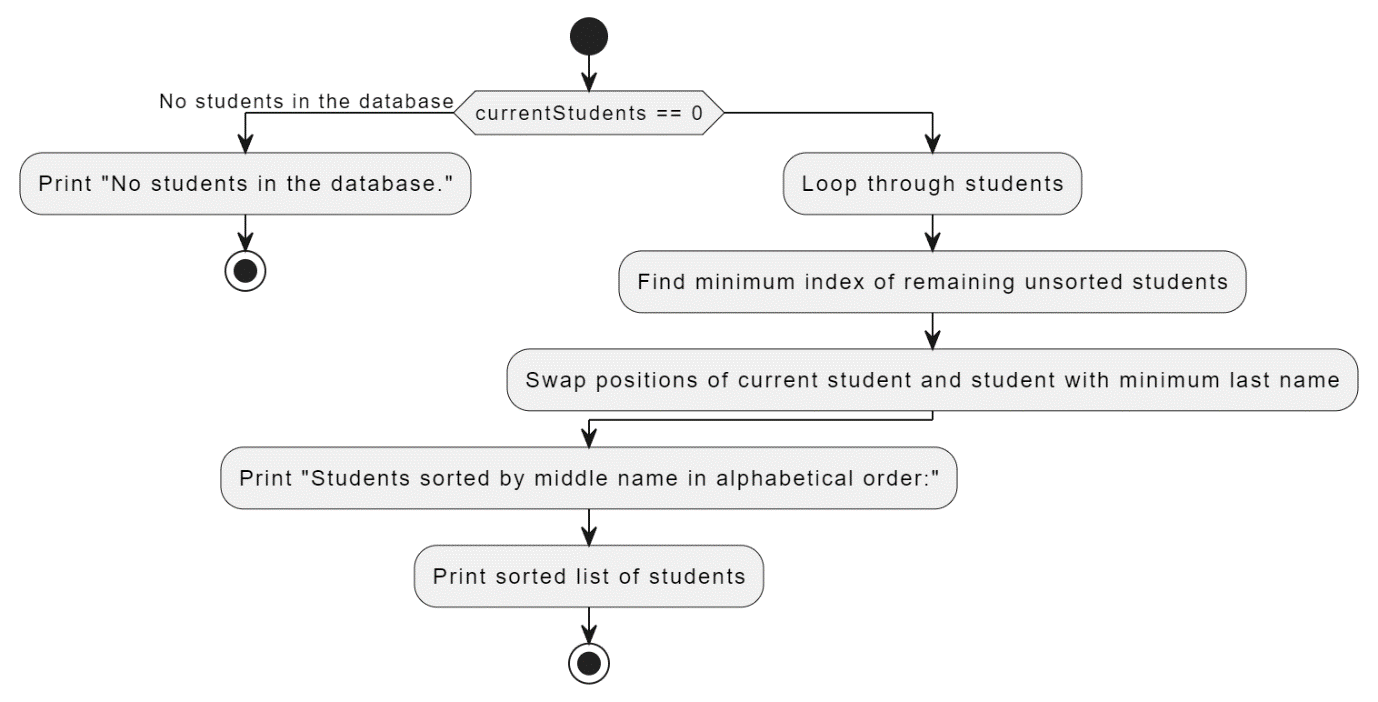
Извеждат се студентите чиито данни попадат в определеният критерий.

Сортиране и извеждане на студенти:

Анализ на алгоритъма:

Осъществява се сортиране на имената на всички студенти по фамилните им имена , подредени по азбучен ред.

Блок схема на алгоритъма:



Функция с която е реализиран алгоритъма:

void SortStudentsByMiddleName();

#### Входни данни на функцията:

Функцията не приема входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат

Извеждат се студентите подредени по азбучен ред.

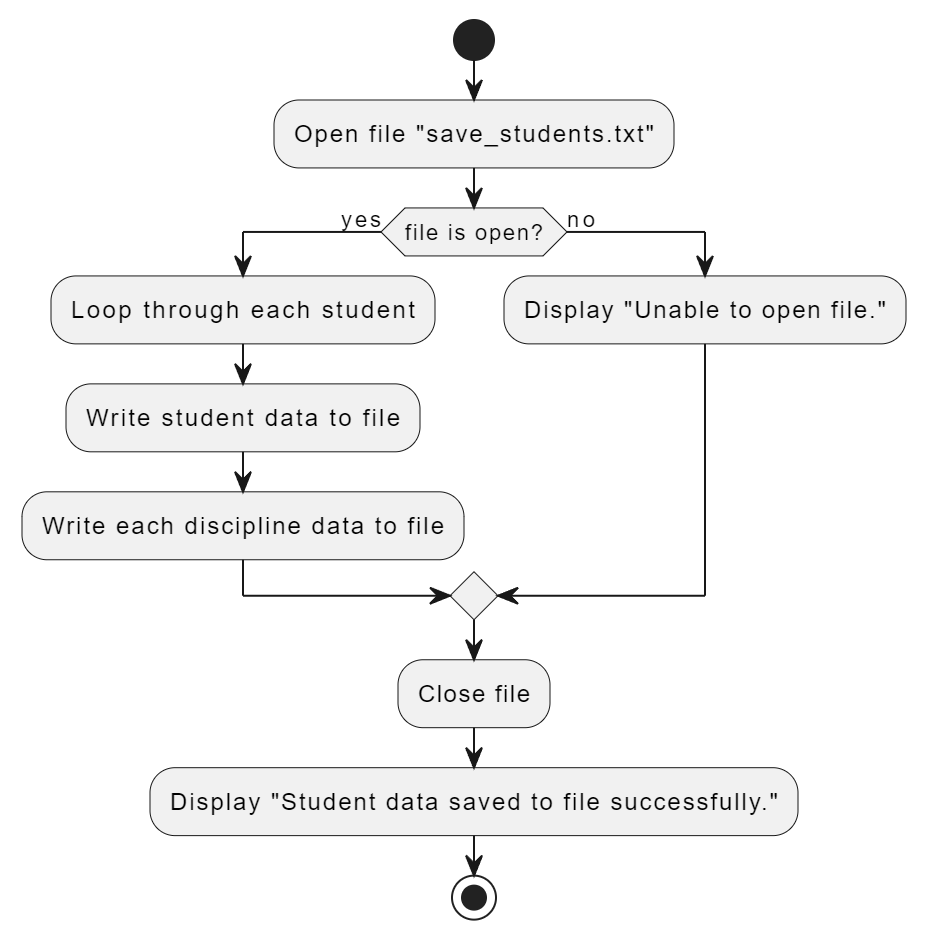
## Работа с външен файл:

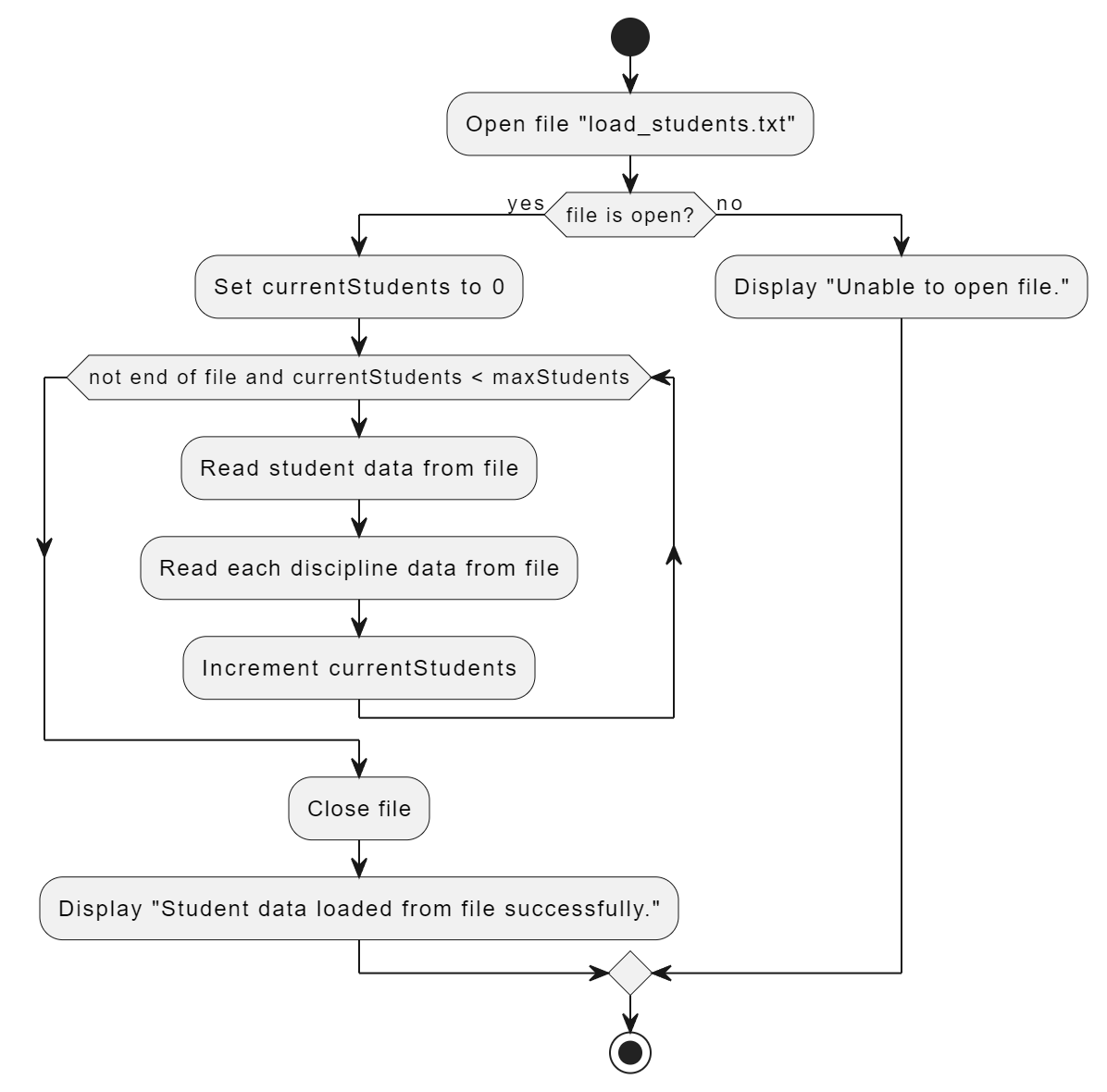
### Анализ на алгоритъма:

#### Функцията отваря файл за запис и записва данни за студенти, включително техните лични данни и оценки по различни дисциплини. Ако успешно отваря файла, данните се записват в определен формат, в противен случай се извежда съобщение за грешка.

#### Функцията отваря файл за четене и зарежда данни за студенти от него. Ако отварянето на файла е успешно, функцията чете последователно данните за всеки студент, включително факултетен номер, социално осигуряване, име, фамилия, пол, възраст, статус и среден успех (GPA). Също така чете и данните за оценките по различни дисциплини. Тези данни се зареждат в структуриран масив от студенти. След успешното зареждане на данните или при грешка при отварянето на файла, функцията извежда съобщение.

### Блок схема на алгоритъма:





Функция с която е реализиран алгоритъма:

void SaveStudentsToFile();

#### Входни данни на функцията:

Функцията не приема входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат

Изписва съобщение, дали данни са успешно записани или не са.

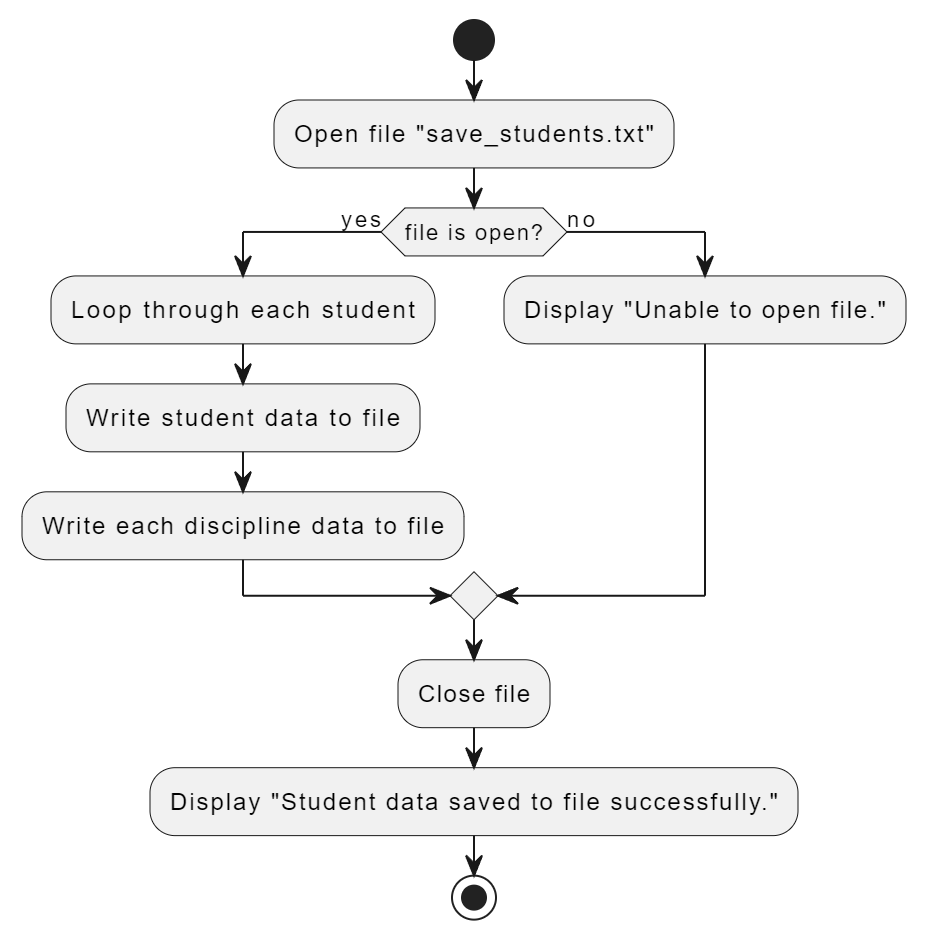
Подменю с алтернативни функции – допълнение първо:

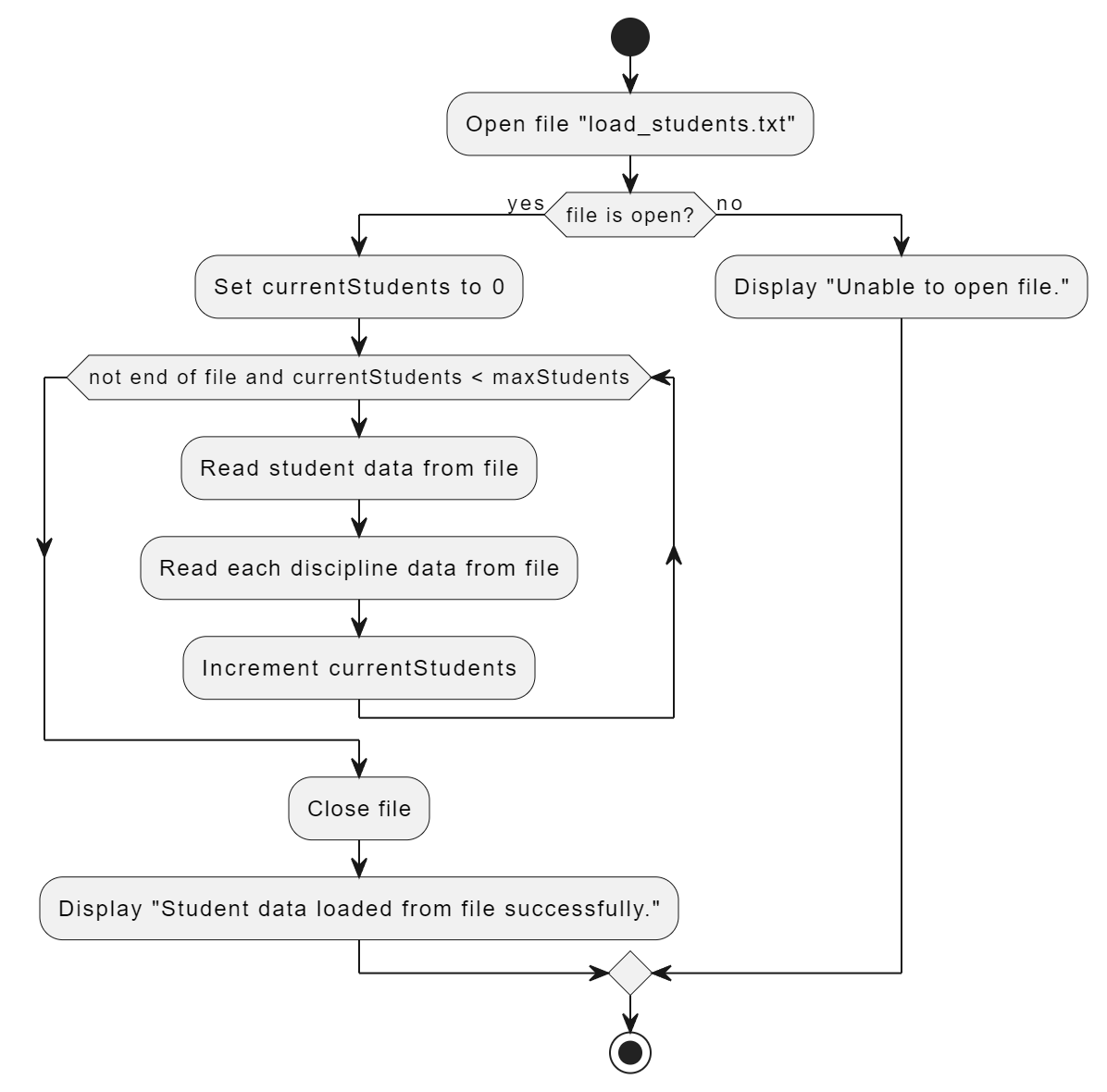
### Анализ на алгоритъма:

#### Функцията изисква от потребителя да въведе статус (Активен/Фамилия/Завършил). След това функцията филтрира студентите според въведения от потребителя статус и ги подрежда в нарастващ ред според факултетния им номер. Тя извежда информация за филтрираните и наредени студенти

#### Функцията изисква от потребителя да въведе статус (Активен/Фамилия/Завършил). След това функцията филтрира студентите според въведения от потребителя статус и ги подрежда в нарастващ ред според факултетния им номер. Тя извежда информация за филтрираните и наредени студенти

### Блок схема на алгоритъма:





Функция с която е реализиран алгоритъма:

#### void SaveStudentsToFile();

#### void LoadStudentsFromFile();

#### Входни данни на функцията:

Функциите не приема входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат

Изписва съобщение, дали данни са успешно записани или не са.

Допълнителни функции – допълнение второ:

### Анализ на алгоритъма:

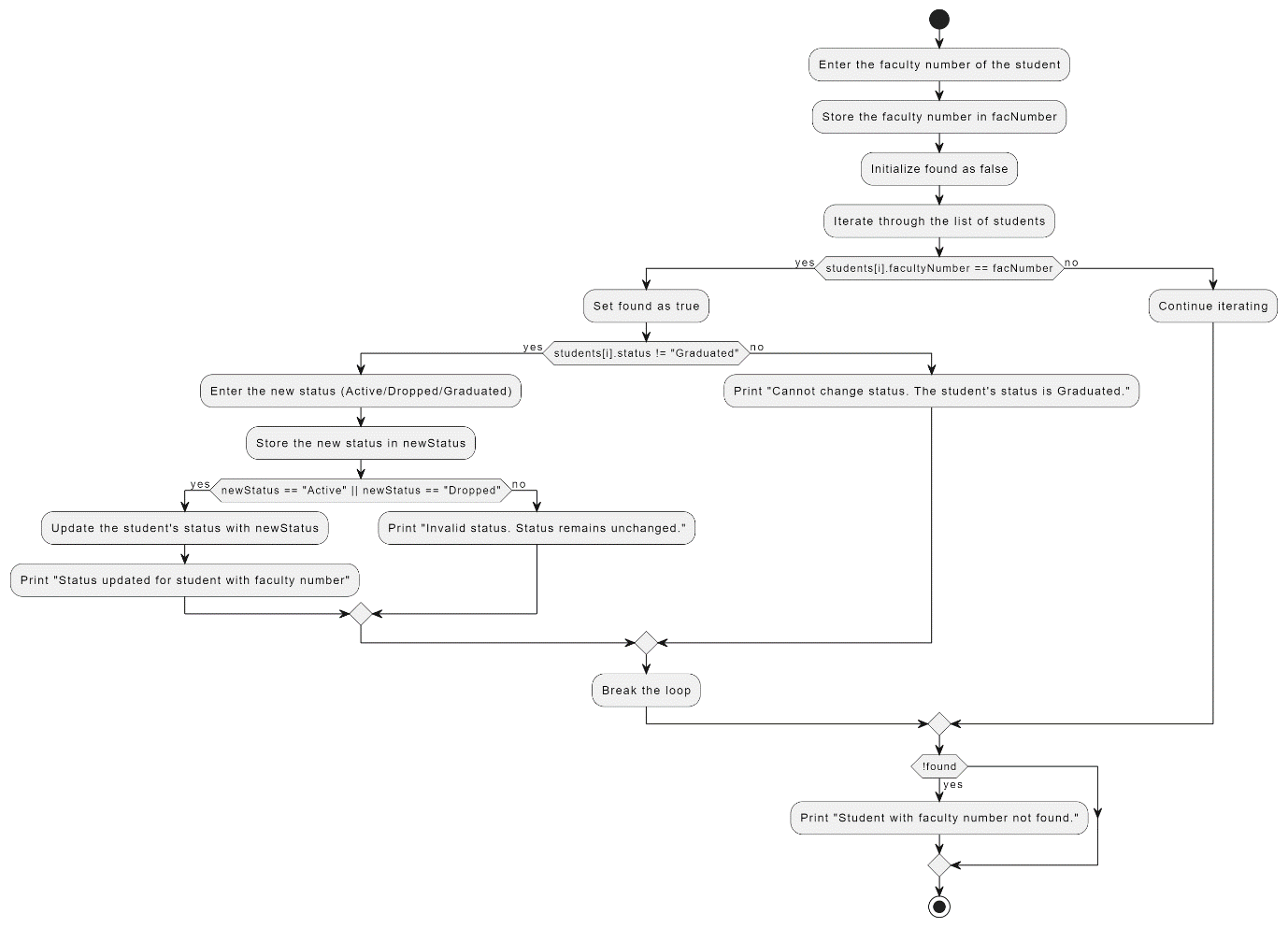
#### Въвеждане и актуализация на оценки:

Търси студентите според техният факултетен номер , като могат да се редактират оценки както и среден успех , според конкретни критерии.

#### Смяна на статут на студент:

Промяна на статус на студент , намерен по факултетният му номер, според конкретен критерий.

### Блок схема на алгоритъма:



Функция с която е реализиран алгоритъма:

void UpdateGradesAndAverage(); void ChangeStatusByFacultyNumber();

#### Входни данни на функцията:

Функциите не приема входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат

Сортировката се осъществява

Допълнителни функции – допълнение трето:

### Анализ на алгоритъма:

Извиква се вече готова функция при стартирането на програмата и в нейния край.

Функция с която е реализиран алгоритъма:

void SaveStudentsToFile(); void LoadStudentsFromFile();

#### Входни данни на функцията:

Функциите не приемат входни данни, данни използвани за обработка на операциите се получават от потребителският вход.

#### Изходни данни на функцията или данни, които се извеждат:

Съобщение оказващо статуса на записването/четенето

Упътване за употреба

Информационната система за студенти осигурява правилният мениджмънт и управление на студенти в група както и техните лични и академични данни.

Добавяне на студенти в група:

Очаквани входни данни:

Номер на студенти които трябва да се добавят, данни на студентите както и техните оценки

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Съобщение че данните са записани успешно, в случай че някаква част от данните е записани с различен формат , програмата кара потребителят да я въведе отново, осигурявайки безпроблемно записване.

Извеждане на студентите на екрана:

Очаквани входни данни:

Избира се от менюто.

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Данните на студентите се показват в табличен вид.

Търсене и извеждане на екрана:

Очаквани входни данни:

Избира се по кой критерии ще търси.

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Данните на студентите се показват в табличен вид.

Сортиране на студентите на екрана:

Очаквани входни данни:

Не са нужни

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Данните на студентите се показват в табличен вид.

Работа с външен файл:

Очаквани входни данни:

Не са нужни входни данни, избира се от менюто опциите за запазване/зареждане на данни

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Съобщение че файлът е зареден/записан успешно.

Подменю – първо допълнение:

Очаквани входни данни:

Данните по които се търси.

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Осъществява се диалог с потребителят , кои данни ще промени и дали има възможност да ги промени според конкретният критерий.

Въвеждане и актуализация на оценки – второ допълнение:

Очаквани входни данни:

Сортировката става само по факултетен номер

Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Показват се студента с конкретния факултетен номер, ако промяна на оценка е възможна програмата ще каже.

Смяна на статус – второ допълнение:

Очаквани входни данни:

Сортирането става по факултетен номер

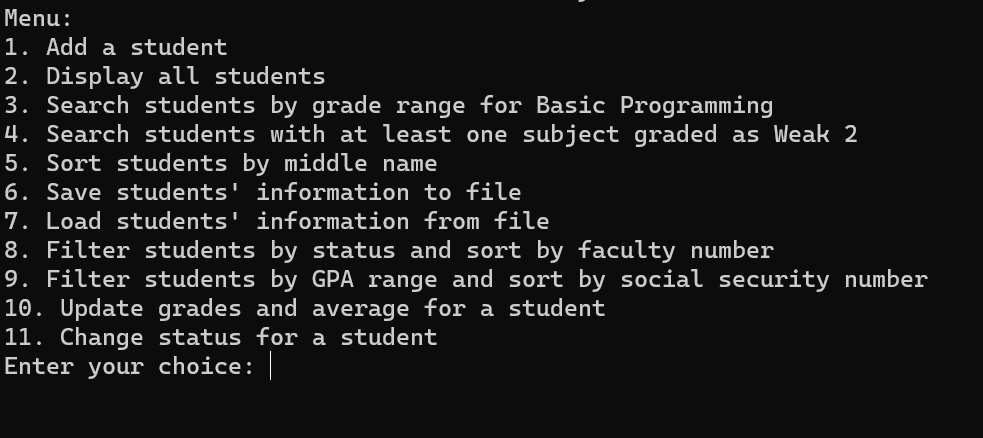
Очакван резултат от изпълнението на конкретния фрагмент от проекта:

Промяна ще бъда възможна само по конкретен критерий.

Примерно действие на програмата

Меню за избор:

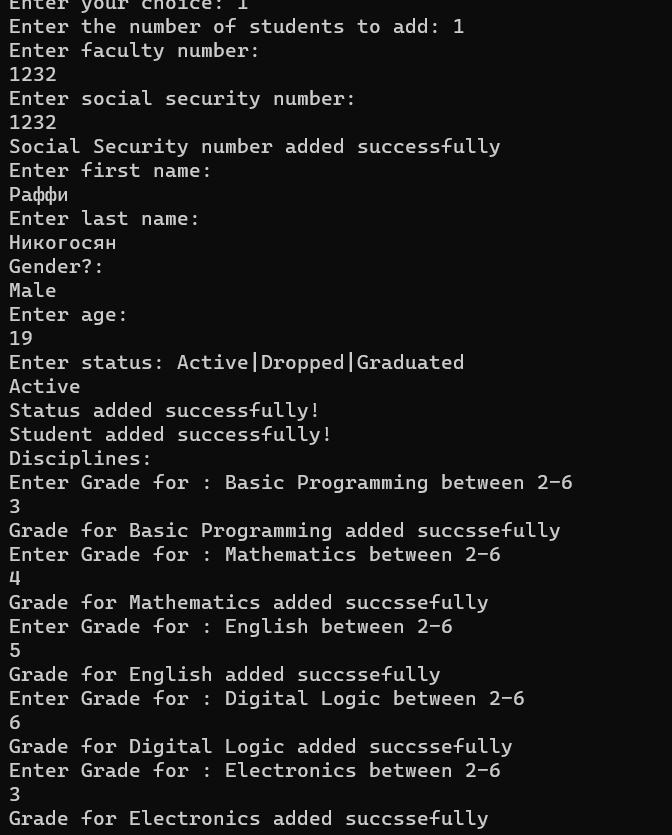
Снимка на изгледа с примерни входни данни:



Снимка на изгледа с примерни изходни данни:

Добавяне на студенти:

Снимка на изгледа с примерни входни данни:



Снимка на изгледа с примерни изходни данни:

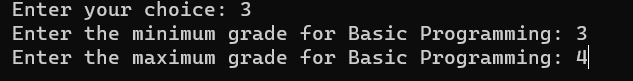


Извеждане на екрана:

### Снимка на изгледа с примерни входни данни:

Търсене и извеждане на екрана:

Снимка на изгледа с примерни входни данни:



Снимка на изгледа с примерни изходни данни:

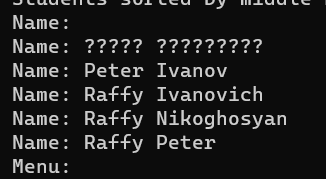


Сортиране и извеждане на студенти:

Снимка на изгледа с примерни входни данни:

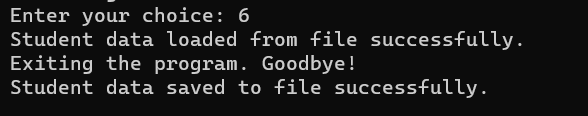


Снимка на изгледа с примерни изходни данни:



Работа с файл:

Снимка на изгледа с примерни входни данни:



Сортиране по факултетен номер – допълнение първо:

Снимка на изгледа с примерни входни данни:

Снимка на изгледа с примерни изходни данни:

:

