

Домашна работа №2

КН, 2-ри курс, 2-ри поток

СРОК ЗА ПРЕДАВАНЕ:
посочен в moodle

Изисквания за предаване:

- Предаване на домашното в указания срок от всеки студент във вид на .zip архив със следното име: **(номер_на_домашно)_КН_(курс)_(поток)_(група)_(факултетен_номер)**, където:
 - **(номер_на_домашно)** е число, отговарящо на номерът на домашното, съответстващо на решението (например 1);
 - **(поток)** е число, отговарящо на потока Ви (например 2);
 - **(курс)** е число, отговарящо на курс (например 1);
 - **(група)** е число, отговарящо на групата Ви (например 1);
 - **(факултетен_номер)** е число, отговарящо на факултетния Ви номер (например 12345);
- Архивът да съдържа само:
 - Изходен код (.cpp и .h файлове) с решение, отговарящо на условията на задачите, като файловете изходен код за всяка задача трябва да са разположени в папка с име **(номер_на_задача)**, където **(номер_на_задача)** е номера на задачата, съответстваща на решението;
 - Един текстов файл с име github.txt, съдържащ линк към изходния код в GitHub;
 - Файлове от друг тип (.db, .txt и др.) са допустими единствено в случай, че това се изисква явно в условието на дадена задача;
- Предаване на архива на посоченото място в moodle;
- Спазване на форматирането на данните посочено в Примерен вход и изход;

- Да се пишат коментари за пояснение;

Пример за .zip архив за домашно: 1_KN_1_2_1_12345.zip

Задача 1. Програма за управление и преглед на вече съществуващи записи с оценки на студенти по програмиране

Да се напише програма, предоставяща възможност за преглед, обновяване и изтриване в постоянната памет на предварително създадени записи с оценки на студенти (вече съществуващи във вид на файл в файловата система).

Ваша задача е да създадете предварително (по избран от Вас начин) текстов файл с име **StudentsGrades.db**, съдържащ поне 100 000 записа, всеки от които в следния формат: *FN FirstName LastName Grade*, където:

- *FN* е факултетен номер на студент (цяло положително число с максимална стойност 2^{50});
- *FirstName* е първото име на студента (низ, с максимална дължина 2^8 символа);
- *LastName* е фамилия на студента (низ, с максимална дължина 2^8 символа);
- *Grade* е оценка на студента (цяло положително число в интервала $[2;6]$);

Всеки запис за студент във файлът трябва да бъде отделен на нов ред. Факултетния номер *FN* за всеки запис за студент трябва да е уникален за файла (във файла не трябва да присъстват записи с дублиращи се стойности за *FN*).

Основно изискване е действията (приложени с описаните по-долу команди за обновяване и изтриване) да се прилагат върху информация в посочения файл за постоянно и да не се губят след изход от програмата и последващото и стартиране.

При стартиране програмата очаква на стандартния вход една от следните команди: **update**, **delete**, **sequentialSearch**, **exit**. Значението на всяка една от тези команди е следното:

- **update** – след въвеждане на тази команда от потребителя се очаква да въведе новата информация за оценка на студент по програмиране в

следния формат *FN Grade*. След въвеждане на факултетен номер, оценка и натискане на Enter:

- В случай на съществуващ запис за студент с посочения *FN*, програмата автоматично обновява въведената оценка *Grade* в файлът с име **StudentsGrades.db**, извежда съобщението „**Record saved!**“ и се връща в начало състояние (очаква от потребителя да въведе една от командите: **update**, **delete**, **sequentialSearch**, **exit**).
- В случай, че не съществува запис за студент с посочения *FN* програмата извежда съобщение „**Record not found!**“ и се връща в начало състояние (очаква от потребителя да въведе една от командите: **update**, **delete**, **sequentialSearch**, **exit**).
- **delete** – след въвеждане на тази команда от потребителя се очаква да въведе *FN*. След въвеждане на факултетен номер и натискане на Enter:
 - В случай на съществуващ запис за студент с посочения *FN*, програмата автоматично изтрива записът за студент с въведения *FN* от файлът с име **StudentsGrades.db**, извежда съобщението „**Record deleted!**“ и се връща в начало състояние (очаква от потребителя да въведе една от командите: **update**, **delete**, **sequentialSearch**, **exit**).
 - В случай, че не съществува запис за студент с посочения *FN* програмата извежда съобщение „**Record not found!**“ и се връща в начало състояние (очаква от потребителя да въведе една от командите: **update**, **delete**, **sequentialSearch**, **exit**).
- **sequentialSearch** – след въвеждане на тази команда от потребителя се очаква да въведе *FN*. След въвеждане на факултетен номер и натискане на Enter:
 - Програмата извършва последователно търсене търсене за време $O(n)$ във файлът с име **StudentsGrades.db** и в случай на точно съвпадение с въведения *FN*, тогава на стандартния изход се извежда пълната информация за студента в следния формат *FN FirstName LastName Grade*.
 - В случай, че не съществува запис за студент с посочения *FN*, програмата извежда съобщение „**Record not found!**“.
 - След извеждане на информацията, програмата се връща в начало състояние (очаква от потребителя да въведе една от командите: **update**, **delete**, **sequentialSearch**, **exit**).

- **exit** – след въвеждане на тази команда се излиза от програмата.

Извадка с първите два записа от примерен файл StudentsGrades.db, съдържащ поне 100 000 записа :

100000 Angelina Antonova 6 105000 Bilyana Gospodinova 5
--

Примерен вход и изход:

Примерен вход:	Изход:
delete 100000 sequentialSearch 100000 update 105000 6 sequentialSearch 105000 exit	Record deleted! Record not found! Record saved! 105000 Bilyana Gospodinova 6