*Документация на проект 1 по ООП*

Тема 5 – Личен календар

Изготвил: Габриел Тупаров, 2MI0800131

Съдържание:

1. Увод
2. Преглед на предметната област
3. Проектиране
4. Реализация и тестване
5. Заключение

Линк към github хранилище на финалната версия: [Release Last version · icydingo29/Personal-Calendar (github.com)](https://github.com/icydingo29/Personal-Calendar/releases/tag/final)

1. Увод

**Описание и идея на проекта** - Проектът „Личен календар“ е създаден с целта да улесни една монотонна и досадна задача, с която всеки човек се сблъсква – именно да организира и запазва разнообразните събития, на които трябва да присъства. Програмата работи в диалогов режим, като въведените от потребителя данни се запазват в текстови файлове, годни за по-нататъшна обработка.

**Цел и задачи на разработката** – Цел на разработката е естествено да се осъществят функционалности, които могат да обработват данните, въведени от потребителя. След като данните биват преработени и оформени в подходящ вид програмата ги запазва в текстови файлове, за да може в бъдеще те отново да бъдат достъпни за промяна или заличаване. Осъществена е валидация при всяко вкарване на данни от страна на потребителя, за да се елиминират потенциални грешки, застрашаващи работата както и на програмата, така и на потребителят. При всяко отваряне на файл се прави проверка, дали файлът е отворен успешно, за да се предотврати губене на информация. За цел стои реализацията на седем функции.

Функция 1- За създаване на ново събитие. След като потребителят въведе име и коментар за конкретно събитие, започва проверка за коректно въвеждане на следните данни – дата, начален час, краен час. След като дадените характеристика бъдат потвърдени за правилно въведени, се проверява дали потенциалната среща не би се случила по време на някоя друга, въведена по-рано. В зависимост от това, дали има такава среща, се изкарва подходящо съобщение, казващо на потребителя, че вече има друг ангажимент по това време.

Функция 2 – За изтриване на събитие. След като потребителят въведе датата, на която отмененото събитие е щяло да се състои, му се показва списък със всички събития на този ден. Той въвежда индекса на ангажимента, който желае да премахне. Поради това, че директното триене от файл крие рискове, свързани със загуба на данни, се създава нов файл, където се записват всички срещи, освен отменената. Старият файл за четене се изтрива, а след преименуване новосъздаденият файл заема неговата роля.

Функция 3 – За показване на събития. След като потребителят въведе дата, програмата показва всички събития на датата, подредени в хронологичен ред. Ако няма такива, потребителят получава подходящо съобщение.

Функция 4 – За промяна на събитие. След като потребителят въведе дата, апликацията показва всички събития на датата. Ако липсват такива, изскача подходящо съобщение. След въвеждане на индекса на тази, която ще претърпи промяна, потребителят трябва да въведе кое иска да промени на срещата. След валидация на въведеното и проверка за пресичане с други събития. Повтаря се техниката със писането на нов файл, като този път се записва същата среща с един променен параметър.

Функция 5 – За намиране на среща. След като потребителят въведе дума в конзолата всички срещи, чието име или коментар съдържат дадената дума биват изведени.

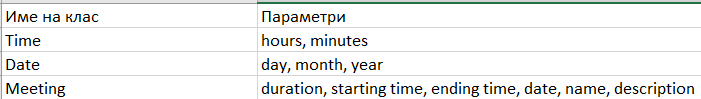
Функция 6 – За извеждане на натовареност на срок от време. След като потребителят въведе начална и крайна дата, програмата създава списък, които представлява ден и брой заети часове. Списъкът се сортира във възходящ ред и се извежда. Същият се записва във текстов файл.

Функция 7 – За търсене на време за среща. След като потребителят въведе начална дата и времетраене на нова среща, програмата търси дали е възможно в следващите 7 дни, започващи с въведеният ден, да се състои такова събитие. Извежда резултата.

1. Преглед на предметната област

Проектът е изграден с множество класове, които опростяват процеса на структуриране, обработване и писане на данни. Създадени са също и помощни структури, със същата цел. Често в програмата се налага да бъде подреден масив от елементи, като това се достига по следния начин – програмата минава през целия масив, намира индексът на най-малкия елемент, и разменя стойностите на този и на сегашния индекс. След това измества индекса, и повтаря процеса докато не се стигне последният възможен. Друг използван алгоритъм е този за търсене на елемент в масив – по подаден масив и негов размер се обхожда масива, алгоритъмът обхожда всички елементи, докато не срещне търсеният. Проблем би могъл да бъде входът, който потребителят въвежда, но програмата проверява дали въведените данни са в коректен формат. Ако не са, потребителят бива накаран да въвежда, докато написаното от него не е правилно. Изключение е когато потребителят трябва да въведе думи. Тъй като е възможно да бъдат въведени ненужни празни полета(“space“) програмата премахна тези в началото и края на входа, като така се избягва нуждата от повторно въвеждане. Сложен е процесът на проверяване дали една среща се препокрива с друга среща. Това препятствие може да бъде превъзмогнато, като след въвеждане на дата, начален и краен час на нова среща, програмата намира всички срещи на съответната дата, превръща началните и крайните им часове в минути, и проверява дали ново въведените начален и кран час превърнати в минути се препокриват. Всеки човек с достъп до програмата има една и съща роля, тъй като апликацията не е направена да дава различни привилегии на различни потребители.

1. Проектиране

На следната таблица са показани трите основни класа, както и имената на техните характеристики. 

Time е клас, описващ дадена точка от времето в деня – с час и минути. Date е клас, описващ дадена дата от годината, съответно с ден, месец и година. До известна степен те са помощни класове на ключовият клас на програмата – а именно Meeting.

Meeting описва среща със следните характеристики: начален час, краен час, дата, име, дескрипция. За разлика от горните класове, параметри на Meeting са други класове –„ starting time“ и „ending time“ са обекти от тип Time, „date“ – от Date, а „name“ и „description“ – от myString, който ще бъде пояснен малко по-късно.

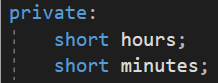
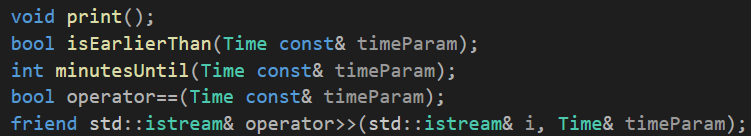
Още една характеристика е “duration” – времетраене. За удобство на потребителят той не трябва да го въвежда – при създаване на нова среща се прави проверка, дали началното време е след крайното, ако не е, и останалите входни данни са коректни, на срещата автоматично се генерира времетраене. При функционалността за промяна на среща, при обновяване на наачлния или крайния час, продължитеността автоматично се прекалкулира. Времетраенето е една от най-важните характеристика на срещата, тъй като програмата също изкарва най-заети дни, спрямо въведени краен и начален ден, а да се пресметне времетраенето на всяка една среща в самата фунцкия би отнело много време. Възможно е естествено и некоректно пресмятане на продължителността, което би довело до неправилна работа на аплокацията.

Другият помощен клас е класът myString. Той съхранява масив от символи („низ“) както и неговата дължина. Той е създаден с мисълта да улесни входа на програмата – вместо да обработваме масиви от символи всеки път, използваме този клас, за да въвеждаме командите, които програмата да изпълнява, както и името и коментара на въвеждана среща. Други начини, по който този клас е от помощ, е при генерирането на име на файл, в който да бъдат записани двойка дата и брой заети часове, което е работата на една от главните функции. Улеснено е също и разпознаването на въведена команда, тък като сравняваме обекти директно от myString, а не низове, което прави кода по-разбираем и четим. Освен това myString има и фунцкионалности за промяна на входа на потребителя – при въведени главни букви и ненужни шпации myString ги премахва, така че да няма нужда потребителят да въвежда едно и също нещо няколко пъти поради невнимателност.

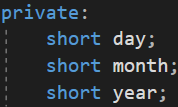
За улеснение са реализирани подходящи селектори и мутатори, конструктори без параметри, конструктори със параметри, оператор =, оператор ==, „копи-конструктор“ и други на класовете, които ги изискват, или с които би се работило по-лесно с реализирани такива функционалости.

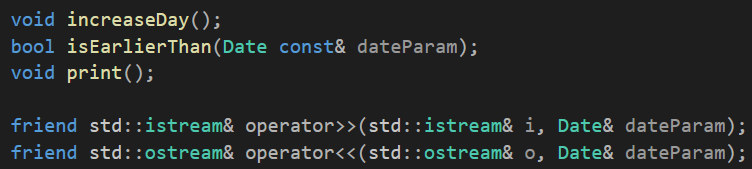
Реализирани са и помощни функции, чиято цел е да правят кода на главните команди по-четим и по-кратък.

1. Реализация и тестване

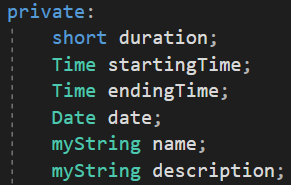
**4.1** Най-малкият и най-просто устроен клас е Time. Параметрите му са от тип short int, защото максималните възможни стойности са 23 и 59, които short int удовлетворява, като така използвмае по-малки памет в сравнение с int. Освен селектори и мутатори, констуктор без параметри, конструктор с параметри и оператор =, са реализирани и следните функционалости: 

print() – за извеждане на конзолата; isEarlierThan() връща true ако текущият обект е по-рано от друг; minutesUntil() връща като резултат броят минути до друго време, а ако другото време е по-рано от текущото, се връща -1; операторът == е с цел да покаже, дали два обекта от Time са с равни параметри; операторът >> е за да може да се записват данни във файл.

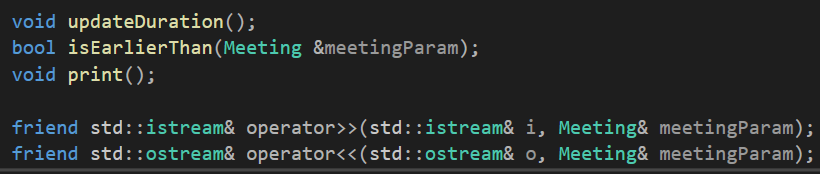
**4.2** Следващия клас е Date. Тук параметрите отново са от тип short int с цел заемане на по-малко памет. Освен селектори и мутатори, конструктор по подразбиране, конструктор с параметри, копи-конструктор, оператор =, оператор за сравнение ==, са реализирани и следните функционалности:



increaseDay() – за промяна на ден; isEarlierThan() – за проверка дали текущата дата е преди друга; ; print() -за извеждане на конзолата; оператори >> и << - съответно въвеждане на данни и писане във файл.

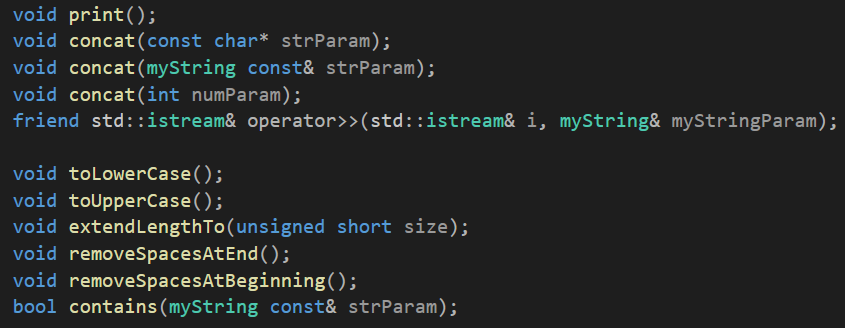


**4.3** Главният клас на проекта – Meeting. Отново използваме short за пестене на памет. Освен селектори, мутатори, конструктор по подразбиране, конструктор с параметри, оператор =, са реализирани и следните функционалности:



updateDuration() – обновява продължителността; isEarlierThan() – проверява дали текущата среща е по-рано от друга. Прави се справка и по дата, и по час;мprint() – за извеждане; опеаратори >> и << - съответно въвеждане на данни и писане във файл.

**4.4** Помощният клас – myString. Освен селектори, мутатори, „голяма четворка“, оператор ==, са реазлизирани и следните функционалности:



print() – за извеждане на конзолата; concat() – за конкатениране на низ, обект от myString или цяло число; оператор >> за въвеждане от конзолата; toLowerCase() и toUpperCase() – съответно за преобръщане на всички букви от низа в главни или малки;

extendLengthTo() – добавя празни места, за да се удължи дължината му до даден размер; removeSpacesAtEnd() и removeSpacesAtBeginning() – премахва празни полета в началото или края на низа; contains() – проверява дали низът на текущият обект съдържа низът на друг обект от тип myString като подниз.

Единственият случай извън командите, където има динамична памет, е в помощният клас myString. В самите команди динамичната памет винаги се освобождава, когато не е нужна. Като оптимизация може да се изведе минималният брой отваряне на файлове – при възможност след като бъде отворен файл елементите от него се записват в масив, така че да се избегне сравнително тежката операция на четене от файл.

Програмата е тествана много пъти в процеса на писане, както и със тестовия сценарии, включен в zip архива.

1. Заключение

Проектът „Личен календар“ има за цел да опрости организирането на събития. Въвеждането и извеждането на срещи се постига с диалоговия режим на работа.

Като бъдещи подобрения може да се визира по-добро илюстриране на срещите, като се показва изображение на 24-часов ден и кои от часовете са заети. Друга идея е приложението да има връзка с интернет, така че да може графиците да се качват в страница, която може да бъде отваряна от всички хора, които срещите засягат. Може и да се направи мобилна версия, която да действа в синхрон с компютърната, така че да може потребителят да променя срещите си от всякакъв уред.