**Увод**

Ефективното разпределяне и обработване на ресурси е от решаващо значение в съвременния свят. Във връзка с това, настоящият проект цели да предостави на потребителя приложение, което дава възможност за лесна и ползотворна работа със склад, съхраняващ различни видове продукти.

Основните цели, които биват реализирани от проекта, са следните:

* Лесен за ползване и надежден интерфейс, който да позволява използването на реализираните функционалности.
* Проверки за коректността на въведените от потребителя данни и извеждане на подходящи съобщения в случай на нередност.
* Възможност за отваряне на вече съществуващи файлове, съдържащи информация за съдържанието на склад склад, и съответно запаметяване на съдържанието на даден склад под формата на файл.
* Възможност за създаване на файл, съдържащ информация за нов, празен склад, със зададени от потребителя параметри за неговия размер.
* Запаметяване на промените, направени по склада, в отделен файл, и съответно четене на направените направените промени по даден склад от съответния файл.
* Възможност да се борави с дати и да се взима директно датата, записана на изпълняващото програмата устройство.
* Ефективно и лесно за навигация представяне на склада в паметта на компютъра.
* Възможност за складиране на продукти, които изискват различно количество място за всяка своя бройка.
* Възможност за добавяне и изваждане на продукти от склада при изисканите от потребителя функционалности и условия
* Възможност за извеждане на екрана на информацията за всеки от продуктите, които се намират в склада.
* Изчисление на загубите от изхвърленото количество конкретен продукт за подаден от потребителя даден период от време.

Документацията е разделена на следните части:

1. Увод
2. Обща архитектура
3. Реализация
4. Заключение

**Обща архитектура**

Архитектурата се разделя на 3 части:

1. Помощна част
   * **class Date** - Позволява за работа с дати и взимане на записаната на устройството дата.
2. Основна част
   * **class Product** - Реализира партида от даден продукт. Пази необходимите параметри за партидата като цяло, а също така съдържа информация за броя налични бройки от съответната партида и за местоположението им в склада.
   * **class Storage** - Реализира склада като го разделя на сектции, рафтове (на всяка секция) и места (на всеки рафт), като размерите на склада не е предвидено да се промунят. Пази се и информация за всички налични продукти и за това на кои места в склада бройка от наличен кой продукт е разположена.
   * **class StorageFile** - Съдържа адреса на текущо отворения файл, който съдържа информацията за склада, а също така и на съответстващия му файл, съдържащ информацията за направените промени по него. Отговаря за прочитането и записването на съдържанието и историята на съответния склад под формата на файлове.
3. Интерфейсна част
   * **class Commands** - Реализира достъпните за потребителя команди и осъществяващи проверки за коректността на входа им.



**Реализация**

1. Помощна част
   * **class Date** - Има отделни променливи от тип **size\_t** за стойността на съответния ден, месец и година. Също така използва **struct tm** за взимане на датата, записана на съответното устройство.
2. Основна част
   * **class Product** - Информацията, която се свежда до низове, бива реализирана чрез **char\***. Дефинира се допълнителна структура **struct Position**, която да пази разположението на дадена бройка от партидата в склада. Чраз масив от тип **Position** биват запазвани “най-левите” места, заети от всяка бройка (позволено е дадена бройка да изискват по няколко последователни места в склада в зависимост от партидата си, но задължително местата трябва да са разположени на един рафт). Също така има променлива от тип **size\_t**, която пази уникален индекс за партидата и улеснява работата с нея в склада.
   * **class Storage** - Информацията за наличните партиди се пази в динамичен масив от тип **Product\*** като всеки указател сочи към налична партида. Съдържанието на склада се реализира като тримерен масив от тип **Product\*** като всеки указател е **nullptr**, ако мястото е свободно, а иначе е указател към партидата, чиято бройка заема това място. Дефинира се и допълнителна структура **struct ProductChange**, която пази информация за направените промени и всеки вид промяна си има уникален символ от тип **char**. Историята за направените промени (без записаните във файла-история по-стари промени) се записва в динамичен масив от тип **ProductChange**.
   * **class StorageFile** - Съдържа указател с тип **Storage\***, който сочи към съдържанието на сегашно отворената таблица. Реализира два низа чрез **char\***, като единият съдържа адреса на файла, имащ съдържанието на склада, а другият адреса на файла, съдържащ историята за направените промени по склада. По подразбиране името на файла, съдържащ историята на промените по склада, е същото като това на файла за съдържанието на склада, но е следвано от “**\_history**” и има разширение “**.txt**”.
3. Интерфейсна част
   * **class Commands** - Има методи, отговарящи за проверка на коректността на входа. Методът **void menu()** отговаря за разпознаване на командите, а останалите методи реализират функционалността на командите и използват само публичните методи на класовете от общата част. Също така командите за отваряне, записване и създаване на файлове позволяват на потребителя да борави само с формата по подразбиране за файла, съдържащ историята на промените по склада.

**Заключение**

Крайните резултати от изпълнението на проекта са следните:

* Всички от поставените задачи бяха успешно реализирани. Това включва следните функционалности на програмата:
  + Запамутяване на информацията за съдържанието и промените по склада в текстови файлове под подходящ формат. Възможност за прочитане и записване на файлове в дадения формат.
  + Подходяща и ефективна структура за склада.
  + Възможност да се извежда на екрана информацията за продуктите както по отделни партиди, така и по име на продукта.
  + Възможност за добавяне на продукти в диалогов режим. Алгоритъм, който да разпределя продуктите при условието продукти с еднакво име и срок да годност да са един до друг (ако е възможно) и да не са един до друг, ако са с еднакво име и различен срок на годност.
  + Изваждане на продукти от склада в диалогов режим. Изваждане първо на бройки от партидите с най-скоро изтичащ срок на годност и допитване до потребителя, ако е поискал да премахне повече от наличното количество даден продукт.
  + Извеждане на екрана на всички промени по склада за даден времеви период, включително от минали изпълнения на програмата, информацията за които е записана в съответния файл, съдържащ история за промените.
  + Възможност да разчисти склада от всички стоки, чиито срок на годност е изтекъл или ще изтича съответния ден. Записване на направените промени.
* Също така бяха реализирани и следните допълнителни функционалности:
  + В диалогов режим, по подадено име на продукт, цена на бройка от него и времеви интервал, програмата изчислява загубите за потребителя вследствие изхвърлянето на съответния продукт поради изтекъл срок на годност.
  + Възможност бройките от дадена партида да не са непременно последователно разположени в склада. Също така всяка партида може да изисква бройките да заемат различен брой последователни места в склада, но задължително всяка бройка трябва да бъде разположена върху един рафт.
  + Използване на уникален символ за различните видове действия, които могат да се извършат върху продуктите. Това се използва при отпечатването на промените на екрана и записването им във файла. Също така се записва и информация за местоположението на всяка променена бройка от партидата.
  + Възможност за създаване на нов склад с подадени от потребителя размери и записване на информацията за него в два файла, имащи подходящия формат, по подаден от потребителя адрес за файла със съдържанието на склада.

Окончателно, би могло да се заключи, че проектът е успешно реализиран.