Конвенции за писане на код

Основната конвенция, с която е реализирано приложението е конвенцията за писане на код в Java.

**Naming Conventions в JAVA:**

**Пакетите** се именуват с малки букви.

**Класовете** се именуват в CamelCase и да се кръщават със съществителни.

**Интерфейсите** се именуват в CamelCase и започват с "I".Името им трябва да описва операцията, която те правят.

**Методите** се именуват в CamelCase и съдържат описващ метода глагол.

**Променливите** се именуват в CamelCase и името им трябва да подсказва, за това какво представляват.

**Константите** се именуват в ANSI, думите трябва да се с главни букви, разделени с долни черти ("\_").

**Добри практики за писане на код в JAVA:**

1.Декларираме локалните променливи непосредствено преди употребата им, с което определя обхвата им и намаляваме вероятността от грешки .

2. Декларираме полетата като private с цел предпазване на данните и ги достъпваме чрез get и set методи.

3. Подреждаме отделно членовете на класа в зависимост от това дали са private, protected или public.

4.Използваме Javadoc liberally

5.Валидираме аргументите на методите

6.Файловете да съдържат по-малко от 2000 реда.

7. Дължината на редовете да е не повече от 80 символа, за да могат да се възприемат правилно от повечето терминали.

8. Ако изразът е прекалено дълъг да се разбива на по-малки части след запетая, преди оператор или с началото на израза, на същото ниво като предходния израз.9.Коментарите да са на един ред- Еnd of line.

10. Декларациите да са в началото на блока от код и да е по една на ред.

11. Инициализиране на локалните променливи там където са декларирани.

12.Класът да е деклариран като се спазват правилата за форматиране:

* Няма разстояние между името на метода и скобата, с която започва списъкът с параметри;
* Отварящата скоба “{” е в края на същия ред, където е декларацията;
* Затварящата скоба “}” е сама на следващия ред и съответства на отварящата си такава;
* Методите са разделени с един ред.

15.Празните пространства в кода трябва да са използвани :

* за разделяне на методи;
* за разделяне на локалните променливи в метод и първия statement;
* между логически раздели в метод за подобряване на четимостта;
* преди коментар;
* между ключова дума и скоба;
* след запетаи в списък с аргументи;
* Всички бинарни оператори освен „.“ са разделени с интервал;
* Между изразите във for statement.

16. Използваме скоби в изразите за да се избегнат обърквания в изразите и логиката на кода.

*Например: if ((a==b) && (c==d))*

17. Връщаме име на променлива при връщането на стойност, вместо true / false.

Линк за повече информация: <https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html>

<https://www.oracle.com/technetwork/java/codeconventions-150003.pdf>

Oписание на отделните класове, атрибути и методи

1. FrequacyTableRow

Този клас служи за да представя редовете на виртуална честотна таблица, като има следните функции и променливи:

* Променливи за името, абсолютната и релативната честота
* Конструктор с гореспоменатите параметри

1. FrequacyTable

Този клас служи като виртуална честотна таблица, като има следните функции и променливи:

* Променлива *file*, която пази файлът подаден при създаване на обект
* Променлива *exercises*, която пази данните за честотната таблица на упражненията
* Променлива *dates*, която пази данните за честотната таблица на месеците на качване на упражненията
* Getter-и и setter-и на последните две
* Функция, която да изчиства данните
* Функции, които да четат файла и да попълват абсолютната честота и името на реда(точка 1) на всяка от двете променливи за данни
* Функция, която слята релативната честота, използвайки RelativeCalculator класа

1. RelativeCalculator

Този клас служи като калкулатор за релативната стойност на всички редове в таблицата, като има следните функции и променливи:

* Функция, която смята релативните стойности на подадена честотна таблица

1. GUIFrequencyTable

Този клас служи за да предостави на потребителя избор коя таблица да види графично, като има следните функции и променливи:

* Два бутона, всеки от който отваря едно от изброените след тази точка менюта

1. GUIExercisesTable

Този клас служи за да представи графично на потребителя честотна таблица спрямо качените упражнения, като има следните функции и променливи:

* Вътрешна променлива и функция за дебъгване
* Променлива съдържаща данните, които ще се попълнят в таблицата(А)
* Променлива от тип FrequencyTable(Б)
* Функция, която попълва гореспоменатата променлива(А) спрямо данните от честотната таблица(Б), използвайки данните за качените упражнения
* Конструктор, който попълва данните за таблицата и името на колоните
* Функции, които създават и изкарват GUI

1. GUIDatesTable

Този клас служи за да представи графично на потребителя честотна таблица спрямо месеците на качване на упражнения, като има следните функции и променливи:

* + Вътрешна променлива и функция за дебъгване
  + Променлива съдържаща данните, които ще се попълнят в таблицата(А)
  + Променлива от тип FrequencyTable(Б)
  + Функция, която попълва гореспоменатата променлива(А) спрямо данните от честотната таблица(Б), използвайки данните за месеците на качване на упражнения
  + Конструктор, който попълва данните за таблицата и името на колоните
  + Функции, които създават и изкарват