TDD-DAGBOK, Kristin Kullberg

Måndag 2025-01-13, dag 6 (em)

Inser att jag har börjar i helt fel ände. Scrappar allt jag gjort och börjar om för att möta uppgiftskrav från start. Ny dagbok börjar nu!

Börjar med en ToDo-lista i en fil i projektet. Först av allt behöver jag en klass för löprunda (Record), och skriver tester för att testa att det går att skapa en, och sätta distans, tid och datum. Jag skriver tester för tre konstruktorer, en utan argument, en med tid, distans och datum, och en utan datum där jag förväntar mig att dagens datum ska sättas. ID ska jag hantera senare. Reflektion 2025-01-16: Här planerade jag att ID skulle sättas automatiskt. Det har jag gjort om ID varit en int i stället för en String.

Används tid bara i bara sekunder, eller kan det behövas en TimeConversion-klass för att underlätta för användaren att se/mata in tid för löprundor? Jag skriver tester för TimeConversion (HHMMSS till sekunder och vice versa) och tillhörande kod. Funktionen verkar dock inte behövas i denna uppgift? Nu finns det i alla fall.

Behöver en Runner-klass som ska ha egenskaper age, height, weight och fitnessScore. Skriver test som testar konstruktor för en "tom" löpare samt en konstruktor där age, height och weight sätts.

Note: Getters och setters skapar jag för alla klassvariabler. Dessa testas inte.

Nu ger jag mig på FileStorage. Mockar FileStorage-klassen och skriver tester för de, i FileStorage.java, redan skapade metoderna createRecord, readRecord och getRecordIDs, samt ett test för recordNotFound. Använder mig av flera when..then-statements eftersom filen inte finns.

Behöver en klass för att göra olika beräkningar, medelvärden m.m. Detta är morgondagens fokus.

Tisdag 2025-01-14, dag 7

Dags att testa Calculations. Börjar med tester för km/h och min/km där jag använder värden på tid och distans. Använder i koden en metod för formatering för att double ska ha en decimal. Märker också, när koden är skriven men testerna inte går igenom, att decimaltecken för double i det returnerade värdet är att kommatecken. Använder mig av DecimalFormatSymbols(US.locale) för att få en punkt. Kvällsreflektion: Den särskilda formateringen används i flera metoder. Ska jag snygga till och bryta ut till egen metod? Svar ja. Fixar imorgon.

Inser också att man i medelvärde-metoderna kanske vill skicka en hel löptur och få tid och distans “gratis” istället för att skicka tid och distans som argument. Jag gör fler tester för det, och overloadar de redan skrivna metoderna.

Tester skrivs också för totalDistance samt avgDistanceAllRecords, där jag skapar en List<Double> med påhittade distanser metoderna att räkna på.

Tester för att räkna ut dagar mellan löpturer skrivs (daysBetween). En där man skickar med två datum, men även här kanske man vill underlätta och bara jämföra två löpturer, så jag skriver en test för det samt en overload-metod som bara hämter ut datumen från de två senaste löpturerna, och sedan använder sig av original-metoden. Reflektion: Det ska inte gå att få ett negativt värde, så jag returnerar ett absolutvärde. Reflektion 2: Utgår från att datumen i listan är sorterad i ordning med senaste datum sist.

Nu då. Fitness score!

Vad ska jag skicka in? Dagar mellan löpturer ställer till det. Jag utgår här från att man skickar med nuvarande fitnessScore, aktuell löprunda samt dagar mellan löprundor (i implementationen får man använda metoden daysBetween för att få fram det). Slutet-av-dagen-reflektion: Ska man kunna skicka in sin Runner-instans och sätta Runner.fitnessScore direkt i metoden? Kanske i nästa projekt.

I övrigt har jag bråkat med jacoco-rapporter som inte finns, projektets mapp-struktur och pom-fil, 100% missad kod i jacoco-rapport (löstes med ett mvn-kommando) och har nu ett fungerande sätt att generera kodtäckning-rapport.

Imorgon: ID-hantering, se gamla löprundor, radera löprunda.

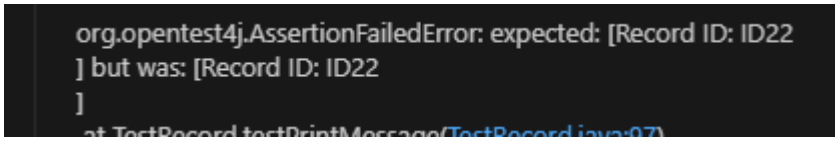
Onsdag 2025-01-15, dag 8

Skriver ett test för formatering av en double till en decimal + kod. Kan då bryta ut denna formatering ur totalt fem metoder. Mycket snyggare. Tester visar fortfarande grönt efter denna refaktorering. Tumme-upp-emoji på det.

Går vidare med att kunna se äldre löprundor. Behöver då lägga till ID på mina Records. Skriver test + kod för ny konstruktor med ID, tid, distans, datum.

Skapar en mock-array med records med ID.

Angående printa info med hjälp av ID. Testar först att använda ByteStreamOutputInfo för att läsa av det som skrivs till terminal med sout. Får felmeddelande och kan tycka att detta meddelande är yttepyttelite ohjälpsamt:



Får det inte att funka och tar ett annat spår. Skickar i stället in ArrayList och ID i metoden getRecordByID.

Det slår mig nu att mina mockRecord etc kanske inte ska heta något med “mock” eftersom de är riktiga records och inte mockade!? Kanske i nästa projekt.

Skriver tester för att printa info om löptur, både statiskt och icke-statiskt.

Nu funkar det att ta bort en record med hjälp av ID. Behöver också test för att inte ta bort om ett felaktigt ID används samt att kontrollera att exception kastas. Said and done. Testar exception även för GetRecordByID.

Testar också metod för att kontrollera om ID redan finns i lista.

Testar att lägga till löprunde-ArrayList som en klassvariabel för runner.

Testar att lägga till en löprunda: funkar. Ska nu testa att det inte går att lägga till en runda med redan existerande ID.

F.n. kan man skapa en ny löprunda, men inte lägga till den i lista om ID redan finns.

Print Record info skrivs om till att även visa medelhastighet km/h och min/km.

Tar bort test och konstruktorer som skapar “tom” löprunda, det ska inte gå att skapa utan ID.

Testar nu att skapa och lägga till löprunda i runner.runlist både med ledigt ID, och även där ID saknas. Om ID inte är ledigt ska IllegalArgumentException kastas. Funkar.

Refaktorerar: Tar bort metod AddRecordToRunList eftersom record automatiskt läggs till i runList när den skapas via createAndAddRecord.

Skriver test för att kunna filtera löpturer längre än en viss distans ur en lista. Funkar.

Torsdag 2025-01-16, dag 9, sista dag innan tenta

Idag är det filhantering och exceptions på agendan!

Forskar på hur man använder when..then när man kallar på en void-metod. Det gör man inte. Man använder doThrow..when. Har tidigare kommenterat bort "Throws exception”-delen i FileStorage.java men nu kör vi på exceptions här.

Tester skrivs för att testa att exception kastas när fel ID används i createRecord, readRecord och deleteRecord. Funkar.

Hmm. Klart?

Klart!

Jag vill tacka min man, inte mina barn, chatGPT och handledare Max för att detta nu kan lämnas in med den mentala hälsan i behåll.

Som en sista sak lade jag till tester för att daysBetween nu även tar emot en ArrayList, sorterar datum i ordning och jämför de två nyaste rundorna, samt returnerar 0 om listan har mindre än 2 löprundor.

Nu lämnar jag in.