

Oppimispäiväkirja  
Teemu Keskinen

SISÄLLYS

[1. Tehtävät 2](#_Toc192670536)

[1.1 Tehtävä 1, viikko 3 2](#_Toc192670537)

[1.2 Tehtävä 2, viikko 4 3](#_Toc192670538)

[1.3 Tehtävä 3, viikko 5 4](#_Toc192670539)

[1.4 Tehtävä 4, viikko 6 5](#_Toc192670540)

[1.5 Tehtävä 5, viikko 7 6](#_Toc192670541)

[1.6 Tehtävä 6, viikko 8 8](#_Toc192670542)

[1.7 Tehtävä 7, viikko 10 9](#_Toc192670543)

[1.8 Tehtävä 8, viikko 11 11](#_Toc192670544)

# Tehtävät

## Tehtävä 1, viikko 3

Tehtävässä luodaan muuttuja numberToGuess käyttäen javan math-kirjaston random operaatiota. Operaatiolle määritetään minimi- ja maksimiarvot muuttujilla min ja max, jolloin voidaan helposti muokata arvauspelin haastavuutta. Käyttäjän syötettä kysytään scanner-kirjaston avulla ja tallennetaan käyttäjän antama vastaus muuttujaan. Tätä muuttujaa sen jälkeen verrataan random operaattorilla generoituun lukuun ja tämän perusteella annetaan vihje käyttäjälle, onko generoitu luku pienempi vai isompi kuin käyttäjän arvaama luku. Mikäli käyttäjä arvaa luvun kerrotaan hänelle tästä. Mikäli käyttäjä ei arvaa lukua oikein yritysmäärän sisällä niin tulostetaan oikea luku käyttäjälle näkyviin.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/tehtavat/week%203/NumberGuessingGame.java>

## Tehtävä 2, viikko 4

Tehtävässä luodaan luokka Birthday, jossa ympäristömuuttujasta puretaan pihalle käyttäjän syntymäpäivä käyttämällä System.getenv() komentoa. Syntymäpäivää käytetään tämän jälkeen laskemaan käyttäjän ikä päivinä, lasketaan kulunut aika käyttäjän viime syntymäpäivästä ja lasketaan myös aika käyttäjän seuraavaan syntymäpäivään. Ohjelmassa on myös lisättynä tarkistus, onko käyttäjän ikä jaollinen 1000 ja tästä onnitellaan käyttäjää. Päivämäärää käsitellään javan java.time.localDate ja java.time.temporal.ChronoUnit kirjastojen avulla. Ensimmäisenä puretaan ympäristömuuttujasta pihalle localDate muuttujaksi sopiva arvo localDate.parse() komennon avulla. Tämän lisäksi tarvitsemme localDate.now(), jota käytämme arvona mihin syntymäpäivää verrataan. Vertailun jälkeen kerrotaan käyttäjälle hänen ikänsä päivinä, milloin hänen seuraava syntymäpäivä on ja milloin hänen viimeisin syntymäpäivänsä oli. Mikäli käyttäjän ikä päivinä on jaollinen tuhannella, tulostetaan hänelle vielä sen lisäksi onnittelut tästä.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/tehtavat/week%204/Birthday.java>

## Tehtävä 3, viikko 5

Tehtävässä luodaan ohjelma, jossa tallennetaan taulukkoon tehtävässä annetut tiedot ja tämän jälkeen luetaan luodusta taulukosta tietoa for-loopin avulla. GitHubissa olleiden valmiiden luokkien avulla saamme taulukosta helposti tiedon käsiteltyä valmiina olevien get()-kutsujen avulla. Tekstinä olevaa kuvausta emme halua kokonaan, joten jaamme sen split()-komennolla ja poimimme sieltä vain haluamamme osat. Päivämäärän saamme tapahtumasta pihalle localDate-kirjastossa olevalla getdate()-kutsulla. Teksti kirjotetaan käyttäjälle kutsumalla printf, jossa tekstiin on asetettu placeholdereita (”%s”) ja nämä placeholderit korvataan komennon lopussa annetuilla muuttujilla (”teksti”, muuttuja, muuttuja, muuttuja), tässä tapauksessa. Viimeisenä tallennetaan kaikki tapahtumat taulukkoon ja puretaan sieltä pihalle vain version nimi getDescription() kutsulla Event-luokasta. Tämän jälkeen taulukko muokataan aakkosjärjestykseen Arrays.sort()-komennolla ja printataan tämä sama pihalle println(”teksti” + Arrays.toString())) komennolla

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/tehtavat/week%205/MacOSRelease.java>

## Tehtävä 4, viikko 6

Tehtävässä luodaan ohjelma, joka pystyy näyttämään annettujen kriteerien perusteella valitut tapahtumat. Tapahtumia edustavat aiemmin tehdyn Event-luokan oliot. Tapahtumat tulevat varastosta, joka on toteutettu esimerkissä. Ohjelma käyttää komentoriviparametreja suodattaakseen tapahtumia päivämäärän ja kategorian perusteella. Komentoriviltä pyydetään kaksi parametria, päivämäärä ja kuukausi muodossa –mm-dd ja tapahtuman primary/secondary kuvaus. Myös pelkästään primary kuvauksella on mahdollista hakea tapahtumaa. Mikäli parametrit ovat virheellisiä palautetaan käyttäjälle virhe, jossa häntä ohjeistetaan oikeasta muotoilusta argumentteja varten. Mikäli tiedot on annettu oikein, suodatetaan tapahtumat annetun päivämäärän ja kategorian mukaan. Tapauksessa, jossa pelkästään primäärikategorian arvo on annettu, asetetaan toinen kategoria wildcard-arvoksi (\*). Tehtävän toteuttamista varten on Today.java tiedostoon pitänyt lisätä metodit: report(LocalDate filterDate, Category filterCategory): joka suodattaa ja tulostaa tapahtumat ja

isSameDate(LocalDate someDate, LocalDate otherDate): joka tarkistaa, ovatko päivämäärät samat (ilman vuotta). Näiden metodien lisäksi on myös category.java tiedostoon tehty muutoksia lisäämällä sinne matches-metodi, joka tarkistaa vastaavatko kategoriat toisiaan. Jos sekundaariarvo on wildcard, se hyväksytään mös.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti



Lähdekoodi:

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/tehtavat/week%206/Today.java>

## Tehtävä 5, viikko 7

Tehtävässä muokataan githubissa olevia valmiita tiedostoja lisäämällä mavenin avulla projektiin ulkoisen kirjaston avulla Apache Commons CSV. Tämä tapahtuu lisäämällä projektin pom.xml tiedostoon dependency.

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus

Tekoälyn generoima sisältö voi olla virheellistä.  
  
  
Tämän lisäksi täytyy tiedostoissa valmiiksi annettua CSV-tiedoston lukemista ja käsittelyä muokata käyttämään uutta lisättyä kirjastoa. Tärkeimmät lisäykset ovat import org.apache.commons.csv importit, joilla ulkoiset CSV-tiedoston käsittelyyn käytetyt kirjastot lisätään.

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus, viiva

Tekoälyn generoima sisältö voi olla virheellistä.  
  
Tämän lisäksi on tiedostoon CSVEventProvider.java nyt lisätty CSVParser lisätyistä kirjastoista, joka koittaa lukea CSV-tiedoston path-muuttujasta, joka sisältää tiedoston sijainnin. Mikäli tiedosto löytyy ja siihen on lukemisoikeus, luetaan jokainen rivi ja kutsutaan event-funktiota ja muokataan rivi tapahtumaksi. Tämän jälkeen tallennetaan juuri luotu tapahtuma listaan ja lopuksi tulostetaan tämä lista pihalle. Myös private event makeEvent funktiota on täytynyt muokkaa, jotta CSVParser toimii oikein. Tänne on tarvinnut muokata syötteen tyyppi CSVRecord tyypiksi, jotta tallennettu CSVRecord-muuttuja voidaan käsitellä String muuttujan sijasta. Tämän jälkeen poistettiin ei tarpeellinen string-split koodista tässä kohtaa, koska record-muuttujassa ne on purettu jo valmiiksi. Today.java tiedostossa on lisätty tiedostoon dynaaminen sijainnin haku ja käytetään Path muuttujaa, jolla voidaan hakea käyttäjän kotikansiosta today kansio ja tämän kansion sisältä events.csv tiedosto. Tämän lisäksi lisättiin tarkistus, tiedoston löytämiseen. Mikäli tiedostoa ei löydy tai sitä ei saada avattua palautetaan käyttäjälle virhe tästä ja suljetaan ohjelma hallitusti.

Kuva, joka sisältää kohteen teksti, kuvakaappaus

Tekoälyn generoima sisältö voi olla virheellistä.

Lähdekoodi:  
<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/Today/src/main/java/tamk/ohsyte/CSVEventProvider.java>

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/Today/src/main/java/tamk/ohsyte/Today.java>

## Tehtävä 6, viikko 8

Tämän viikon tehtävässä oli tarkoitus lisätä EventManager-luokka ja toteuttaa sen sisällä olevat add- ja removeEventProvider funktiot. Päätin toteuttaa tämän ratkaisun viime viikolla tekemäni CSVEventprovider päälle, koska tämä koodi oli minulle tuttu verrattuna githubissa tarjottuun openCSV versioon. Lisäsin projektiini valmiiksi tarjotut tiedostot EventManager ja FirstEventProvider sekä SecondEventProvider. EventManager tiedostossa toteutin add- ja removeEventProvider funktiot try-catch avulla, jolloin virheellisessä suorituksessa tulisi edelleen virheilmoitus. Lisäämisen tein this.eventProviders.add(provider) koodilla, joka lisää kutsuttaessa tapahtuman järjestäjän listaan järjestäjistä. Poistamista varten tein boolean muuttujan, jossa removeIf kutsutaan ja tarkistetaan mikäli annettu provider on yhtäsuuri kuin listassa jo olemassa oleva providerId. Tätä testiä toteutetaan provider.getIdentifier().equals(providerId) koodilla, joka vertaa provider muuttujaan tallennettua arvoa getIdentifier kutsulla saatuun providerId. Mikäli nämä ovat yhtäsuuret poistetaan annettu provider listasta. Today.javaan lisäsin vain githubissa olevalla mallilla kutsut, joilla testataan, toimiiko koodi oikein.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Tekoälyn generoima sisältö voi olla virheellistä.

Lähdekoodit:

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/tree/main/Today/src/main/java/tamk/ohsyte>

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/Today/src/main/java/tamk/ohsyte/Today.java>

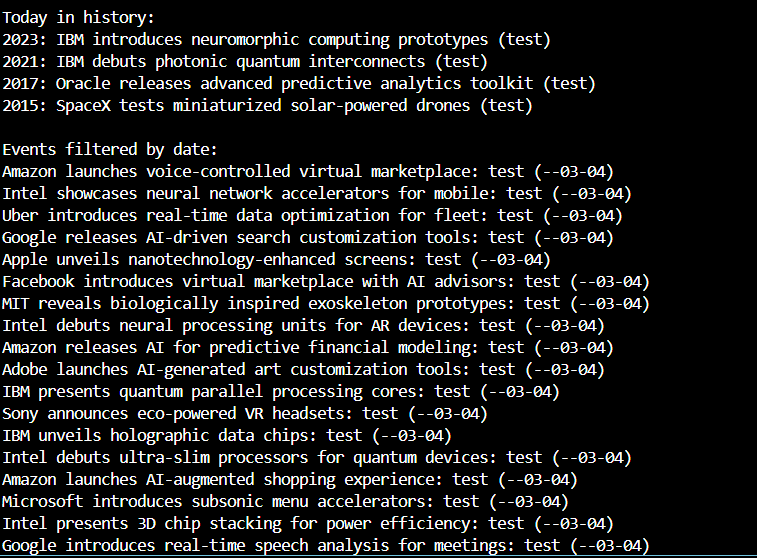
<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/Today/src/main/java/tamk/ohsyte/EventManager.java>

## Tehtävä 7, viikko 10

Tässä tehtävässä laajennettiin ohjelman suodatusmahdollisuuksia lisäämällä EventFilter-abstrakti luokka ja sen aliluokat. Näiden avulla voidaan suodattaa tapahtumia päivämäärän ja kategorian perusteella.

Ensimmäiseksi luotiin EventFilter-abstrakti luokka, joka määrittää metodin accepts(Event event). Tämä metodi tarkistaa, täyttääkö annettu tapahtuma tietyt suodatuskriteerit.

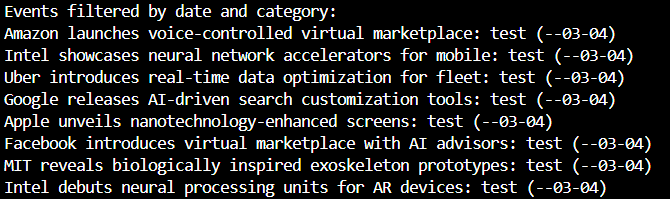
Tämän jälkeen toteutettiin suodatusluokat DateFilter, joka suodattaa tapahtumat päivämäärän perusteella ja tarvittaessa ottaa huomioon myös vuoden riippuen halutaanko tarkistaa, onko kyseessä singular vai annual tapahtuma. CategoryFilter suodattaa tapahtumat niiden kategorian perusteella. DateCategoryFilter, joka yhdistää molemmat edellä mainitut suodatusehdot ja suodattaa tapahtumat sekä päivämäärän että kategorian mukaan. Näiden lisäksi EventManager-luokkaan lisättiin metodi getFilteredEvents(EventFilter filter), joka palauttaa listan tapahtumista, jotka vastaavat annettua suodatinta. Lopuksi Today.java-tiedostossa testattiin uudet suodatusmetodit lisäämällä esimerkkikutsut DateFilter, CategoryFilter ja DateCategoryFilter -luokkien avulla.



(tässä ei riittänyt visual studio coden sisäisessä terminaalissa muistia näyttää alkua)

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Tekoälyn generoima sisältö voi olla virheellistä.



Lähdekoodit:

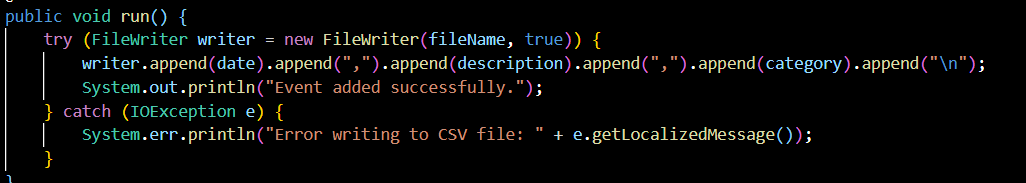
<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/week8/src/main/java/tamk/ohsyte/CategoryFilter.java> [https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/week8/src/main/java/tamk/ohsyte/DateCategoryFilter.java](https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/week8/src/main/java/tamk/ohsyte/DateCategoryFilter.jav)

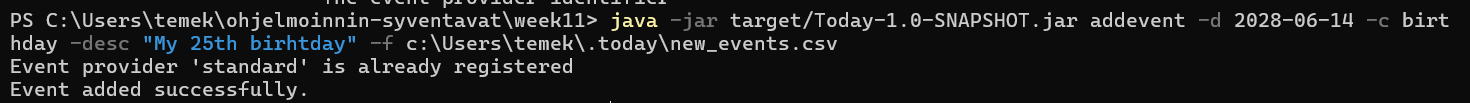
<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/week8/src/main/java/tamk/ohsyte/DateFilter.java>

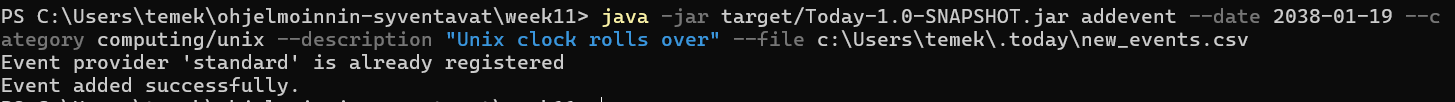
<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/blob/main/week8/src/main/java/tamk/ohsyte/EventFilter.java>

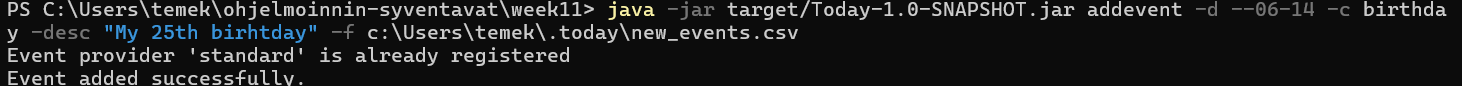
## Tehtävä 8, viikko 11

Tehtävässä lisättiin toiminnallisuus manuaaliseen tapahtuman lisäämiseen. Tätä varten lisätään CSVEventProvider tiedostoon uusi komento ”addevent” joka lisää tapahtuman käyttäjän antamaan .csv tiedostoon. Tämä toteutettiin @command annotaatiolla. @command kutsutaan ensimmäisen kerran CSVEventProviderin alussa, jolla nimetään tulevat komennot. seuraavalla @command annotaatiolla nimetään addevent ja määritetään tarvittavat argumentit. Voidaan nimetä tavat, joilla voidaan kirjoittaa komentorivillä komento. Käytän itse tässä yhtä kokonaan kirjoitettua ja yhtä aliasta, jossa on lyhenteet näille. Komentoriviltä tarvitaan tapahtuman päivämäärä, tapahtuman kategoria, tapahtuman kuvaus ja tiedosto, johon tapahtuma lisätään. Nämä on määritelty vaadittavaksi koodissa lisäämällä required = true @option perään, jolloin nämä tiedot vaaditaan, jotta koodi toimisi oikein. Provider tietokenttä ei ole pakollinen vaan sen voi jättää auki, tällöin se täytetään standard vakionimikkeellä. Tämän jälkeen try-catch avulla yritetään kirjoittaa tietoja csv-tedostoon FileWriter objektin avulla ja mikäli tämä ei toteudu oikein palautetaan virhe käyttäjälle.









Kuva, joka sisältää kohteen teksti, kuvakaappaus, numero, Fontti

Tekoälyn generoima sisältö voi olla virheellistä.

Lähdekoodit:

<https://github.com/TeemuKeskinen/ohjelmoinnin-syventavat/tree/main/week11/src/main/java/tamk/ohsyte>