*Nino Salonen, 521493, niaams@utu.fi*

*18.12.2020 | Ohjelmoinnin perusteet, Erno Lokkila*

*Hirsipuu*

Harjoitustyö

Sisällysluettelo

[1. Tehtävän kuvaus 1](#_Toc59378889)

[2. Ratkaisuperiaate 1](#_Toc59378890)

[3. Moduulien kuvaus 2](#_Toc59378891)

[4. Testausjärjestely 2](#_Toc59378892)

[5. Liitteet 3](#_Toc59378893)

[5.1 Vaadittavat ohjelmat 3](#_Toc59378894)

[5.2 Käyttöohje 3](#_Toc59378895)

# Tehtävän kuvaus

Vaatimuksina oli mm. 3 metodia, listan ja hajautustaulun käyttö, tiedostoista luku ja kirjoitus, sekä vähintään 100 lausetta koodia.

Valitsin tehtäväksi luoda tutun pelin nimeltä hirsipuu. Pelissä käyttäjän täytyy ratkaista tuntematon sana arvaamalla sanaan sisältyviä kirjaimia syöttämällä niitä yksitellen. Peli toimii täysin tekstipohjalla, ja tulevissa laajennuksissa voisi peliin sisällyttää graafisen käyttöliittymän. Alun perin suunnitelmassa oli tarkoitus ilmaista väärien arvausten jäljellä oleva määrä perinteisellä hirsipuu-mallilla ASCII-merkkejä käyttäen. Tämä osoittautui kuitenkin ongelmalliseksi, joten hirsipuun kuvitus jää mahdolliseksi laajennukseksi.

Kun pelaaja arvaa kirjaimen oikein tai väärin, tulostaa ohjelma tilanteen mukaisesti aina saman viestin pelaajalle. Näitä viestejä voisi olla useampi, ja ohjelma voisi valita niistä vuorotellen jonkin. Tämäkin voisi olla yksi laajennuksen kohde.

# Ratkaisuperiaate

Ohjelman käynnistyttyä, kysyy se pelaajalta, haluaako tämä jatkaa edellisestä kerrasta tai aloittaa uuden pelin. Jos pelaaja haluaa jatkaa, lataa ohjelma save.json tiedostosta pelitilanne-hajautustauluun pelin tilan. Jos pelaaja aloittaa uuden pelin, saa hajautustaulu tiedot alkutilanne.json tiedostosta.

Kun peli on alkanut, ohjelma arpoo satunnaisen sanan käyttäen sananvalinta() -metodia. Tämän jälkeen pelaajalta odotetaan syötettä. Syöte sallitaan vasta, kun ohjelma tunnistaa sen kirjaimeksi, ja sen pituus on yksi. Kun syöte on annettu, ohjelma tarkistaa kokeilin() -metodilla oliko kirjain sanassa vai ei. Jos kirjain oli sanassa, ohjelma tulostaa onnitteluviestin käyttäjälle ja oikeiden ”\_” -merkkien paikoille ilmestyy arvattu kirjain ja mahdollisten väärien arvausten määrä pysyy samana. Jos kirjain ei ollut sanassa, ohjelma tulostaa harmitteluviestin käyttäjälle, jäljellä olevien väärien arvausten määrän piirraUkko()metodilla, listan näytölle annetuista vääristä kirjaimista, ja mahdollisten väärien vastausten määrä kasvaa yhdellä. Tällä hetkellä maksimimäärä vääriä vastauksia on 7.

Ohjelma tarkistaa jatkuvasti, onko pelaaja ylittänyt väärien vastauksien maksimimäärää, ja heti kun näin tapahtuu, pelaaja häviää. Ohjelma tarkistaa myös jatkuvasti checkVoitto() -metodilla, onko kaikki kirjaimet arvattu, ja jos on, pelaaja voittaa. Kummankin tapauksen jälkeen käyttäjältä kysytään, haluaako tämä aloittaa uuden pelin.

# Moduulien kuvaus

sananvalinta() -moduuli avaa tiedoston sanat.csv ja valitsee sieltä random.choice() moduulilla satunnaisen sanan, ja palauttaa sen.

piirraUkko(vaarVast) saa parametrinaan pelitilanne-hajautustaulusta väärien vastausten määrän kokonaislukuna, muuttaa sen merkkijonoksi. Avaa tiedoston ukko.json ja tulostaa sieltä numeroa vastaavan arvon.

save(tilanne) saa parametrinaan pelitilanne-hajautustaulun, ja kirjoittaa sen tiedostoon save.json.

kokeilin(sana, arvaus) saa parametrinaan sanan, mikä on pelitilanne-hajautustaulusta pelissä arvattava sana, sekä arvaus, mikä on pelaajan antama syöte. Palauttaa True, jos arvaus on sanassa, tai False, jos ei ole.

lataaAlusta() muuttaa pelitilanne-hajautustaulun vastaamaan lähtötilannetta alkutilanne.json tiedostosta. Se myös arpoo uuden sanan pelitilanne-hajautustauluun.

lataaSavesta() avaa save.json tiedoston ja muuttaa pelitilanne-hajautustaulun vastaamaan save.json tiedostoon tallennettua hajautustaulua.

peli() metodi on yksi vuoro. Tulostaa arvatut väärät kirjaimet sekä arvausten jäljellä olevan määrän piirraUkko() -metodilla. Ottaa käyttäjältä syötteenä yhden kirjaimen, ja käyttää tähän kokeilin() metodia. Jos arvattu kirjain on jo arvattu, ohjelma ilmoittaa tästä ja kutsuu uudelleen peli() -metodia. Nyt ohjelma käyttää kokeilin() -metodia. Jos metodi palauttaa True, sanasta paljastuu kirjaimia ja pelitilanne-hajautustaulun oikein arvattuihin kirjaimiin lisätään kyseinen kirjain. Jos False, väärien vastausten määrä kasvaa ja väärin arvattujen kirjainten listaan pelitilanne-hajautustaulussa lisätään tämä kirjain. Nyt checkVoitto() metodia kutsutaan, jos ei ole voittanut, kutsuu uudelleen peli() metodia uudelleen. Useissa kohdissa kutsutaan myös save() metodia automaattisesti pelin tallentamiseksi. Metodista poistutaan palauttamalla False, esim tilanteessa, jossa pelaaja häviää tai voittaa.

peliKoko() metodi sisältää alkuun peli() metodin. Kun pelimetodista poistutaan, kysytään käyttäjältä haluaako tämä uuteen peliin. Jos haluaa, kutsutaan peliKoko() metodia uudelleen, jos ei, exit().

# Testausjärjestely

Testausjärjestelyina olen antanut ohjelman useammalle perheenjäsenelle kokeiltavaksi. Olen ohjeistanut ilmoittamaan bugeista välittömästi, eli siitä, miltä bugi näyttää ja miten se aiheutui. Moduulien erillisen testauksen olen suorittanut itse, missä ei tullut ilmi ongelmia.

20.12 Paljastunut bugi, salli pelaajan syöttämään numeroita. -Korjattu

26.12 Ilmeni bugi, jossa iso kirjain tulkittiin eri kirjaimena kuin vastaava pieni kirjain. -Korjattu

# Liitteet

## Vaadittavat ohjelmat

Python 3.x

## Käyttöohje

1. Avaa harkkatyö-kansio
2. Avaa hirsipuu.py -tiedosto jollakin Python3 tukevalla ohjelmalla
3. Seuraa ohjelman antamia ohjeita