Lennonjohtopeli

Otto Laitinen - 596530

Tietotekniikka 1. vuosikurssi

05.05.2018

Yleiskuvaus

Lennonjohtopeli. Pelissä on eri vaikeustasoja, joilla on eri määrä ”satunnaisesti” sijaitsevia kiitoratoja ja laitureita. Lentokoneiden saapumistiheys riippuu myös vaikeustasosta. Kiitoradat ovat erilaisia ja eri koneet tarvitsevat erilaisia kiitoratoja (radan minimipituus). Kiitoradat voivat olla myös ristikkäin. Lisäksi lentokenttää pystyy kiertämään “jonossa” eri korkeuksilla. Lentokoneet eivät ainoastaan saavu lentokentällä, vaan osalla koneista on jatkolentoja.

Pelissä on pisteytysjärjestelmä, joka perustuu pelaajan tehokkuuteen, mutta pisteiden saavuttaminen ei kuitenkaan ole pelin itse tarkoitus vaan pelissä tavoitteena on pystyä selviytymään kentän johdossa mahdollisimman pitkään.

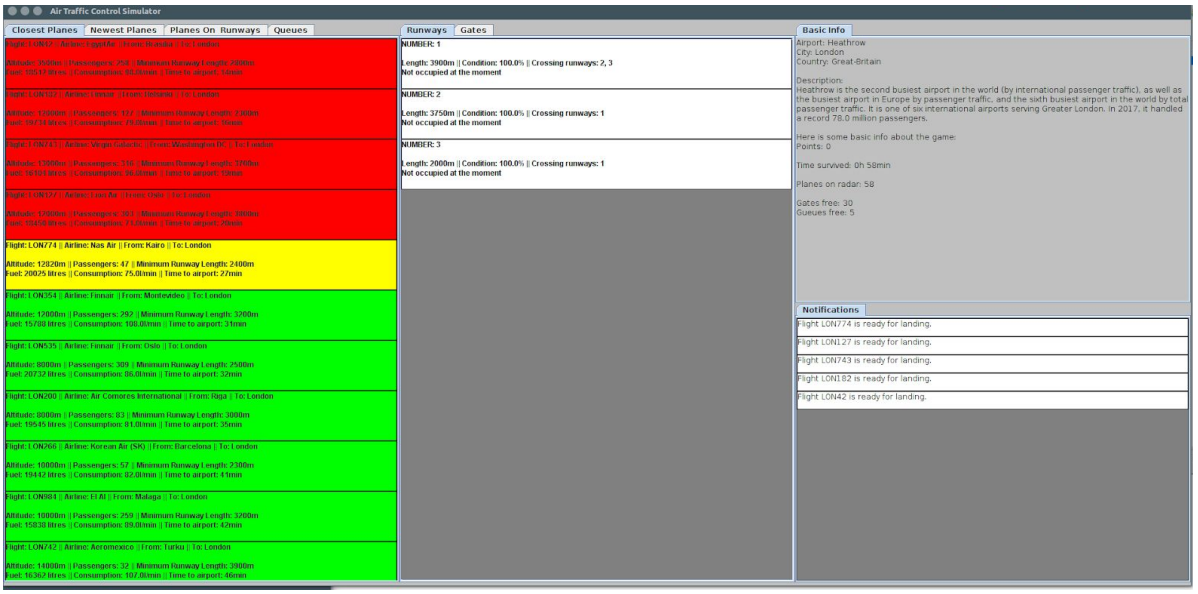
Lennonjohtopelissä ei ole keskitytty animoituun grafiikkaan, vaan GUI on tekstipohjainen. Tavoitteena on ollut antaa pelin pelaajalle samanaikaisesti mahdollisimman paljon informaatiota lentokentästä, mutta myös mahdollisuus muokata näkymää sen hetkisen tarpeeseen sopivaksi.

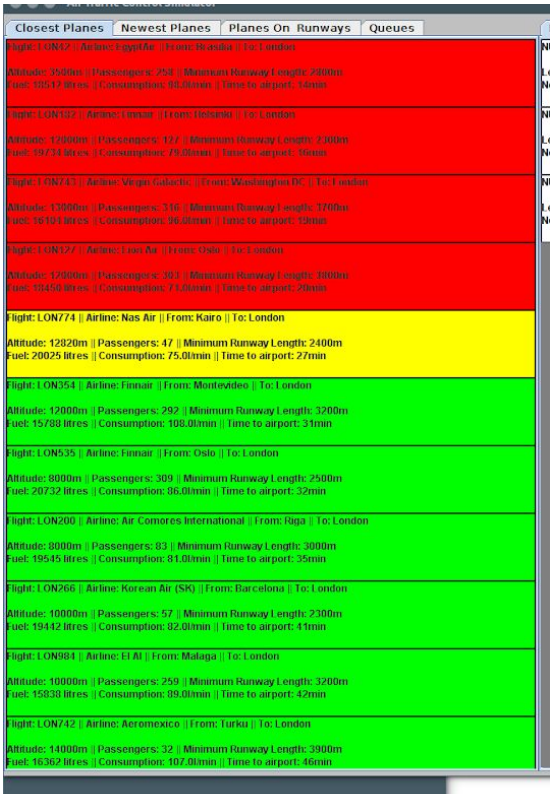
Projekti on toteutettu keskivaikean ja vaativan välimaastossa.

Käyttöohje

Kun ohjelma käynnistetään pelaaja näkee ensimmäisenä aloitusikkunan. Aloitusikkunan on tarkoitus kertoa informaatiota pelistä ja antaa ohjeita pelin pelaamiseen. Aloitusikkunassa valitaan myös vaikeustaso kolmesta mahdollisesta vaikeustasosta/kaupungista.

Kun vaikeustaso valitaan, aloitusikkuna sulkeutuu ja itse peli-ikkuna avautuu. Peli-ikkunan tarkoitus on antaa mahdollisimman paljon informaatiota pelaajalle yhdellä ruudulla ja se koostuu kolmesta vierekkäisestä kolumnista.



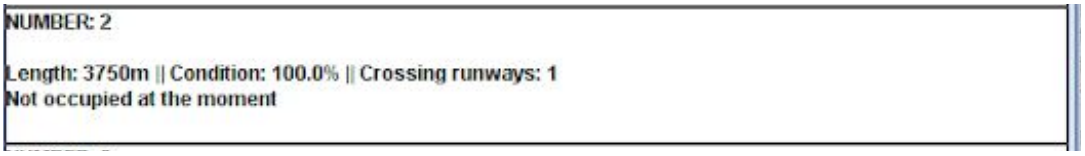


Ensimmäisessä kolumnissa näytetään informaatiota lentokentälle saapuvista tai siellä olevista lentokoneista. Saapuvat koneet voidaan järjestää saapumisjärjestyksen tai ilmoitusjärjestyksen mukaan. Lisäksi lentokonepalkista löytyvät “jonottavat” lentokoneet sekä kiitoradoilla olevat lentokoneet.

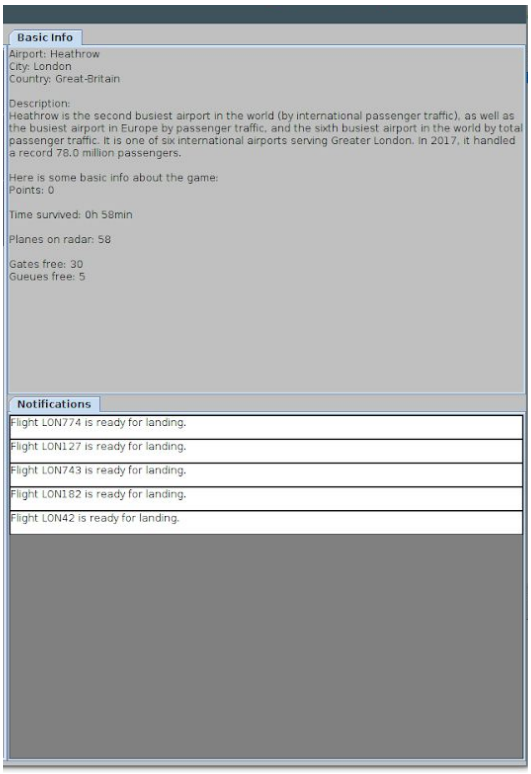
Tekstiruudut, jotka kuvaavat lentokoneita, kertovat paljon informaatiota yksittäisestä lentokoneesta. Tärkeimmät tiedot ovat aika lentokentälle ja jäljellä oleva polttoaine. Ruudut vaihtavat väriä lentokoneen tilan mukaan: mikäli lentokone on vielä kaukana, näytetään se vihreänä, jos taas lentokone vaihtamassa korkeutta, näytetään kone keltaisena ja lentokone on punainen, kun se on alle 30 minuutin päässä lentokentältä. Lentokonetta klikattaessa tulee esiin valikko, josta lentokonetta pystyy ohjeistamaan. Lentokoneelle voi ohjeistaa

laskeutumiskiitoradan, uuden lentokorkeuden tai korkeuden lentokentän kiertämiseen.

Kiitorata- ja porttinäkymä löytyvät keskimmäisestä palkista. Jokaisesta kiitoradasta näkee sen numeron, pituuden ja mahdollisesti risteävät kiitoradat.



Mikäli kiitoradalla on lentokone, näyttää kiitorata kyseisen lentokoneen lennon numeron. Mikäli kiitoradalla on lentokone, on kiitorata maalattu punaiseksi. Mikäli taas risteävällä kiitoradalla on lentokone, maalataan kiitorata keltaiseksi.



Portti näyttää itsestään oman numeronsa ja portin varaustilanteen. Porttia klikkaamalla tulee esiin valikko, jossa on erilaisia vaihto-ehtoja riippuen siitä, onko portilla lentokonetta ja millainen lentokone portilla on. Mikäli lentokoneella on jatkolento, voi lentokoneen lähettää valikosta takaisin kiitoradalle nousua varten, mikäli lentokoneella ei ole jatkolentoa, se voi lähettää hangaariin.

Oikean puoleisesta palkista löytyy perusinfoa kentästä sekä ilmoituslista, joka näyttää viimeisimmät 15 ilmoitusta lentokoneilta.

Yleisesti pelin voi hävitä vain aiheuttamalla koneiden törmäyksen. Tämä voi tapahtua, mikäli koneet törmäävät kiitoradalla tai jos lentokenttää kiertää samalla korkeudella liian monta lentokonetta samanaikaisesti. Myös mikäli lentokone ei

ole saanut ohjeistusta lentokentälle päästyään tai siltä loppuu polttoaine, pelin häviää. Pelin loputtua tulee esiin lopetusikkuna, joka näyttää syyn pelin loppumiseen ja pelaajan pisteet.

Pisteitä pelaaja saa ainoastaan onnistuneista koneiden laskeutumisista, mutta koneiden nouseminen ja jonojen ym. hallitseminen on tärkeää, jotta koneet pystyvät rauhassa laskeutumaan.

Ohjelman rakenne

Pääongelma on lentokentän toiminta ja sen mallintaminen. Tätä kuvaa graafisessa luokkakaaviossa luokka Airport. Pääongelma voidaan jakaa pienempiin osiin, kuten lentokone (Airplane), kiitorata (Runway), portti (Gate) ja jono (Queue) Luokkien väliset suhteet selviävät paremmin luokkakaaviosta.

Toisin kuin alkuperäisessä suunnitelmassa, Airplane luokka tietää, millä kiitoradalla tai portilla ja missä jonossa se on. Tämä on tehty sen takia, että ohjelman painopiste siirtyi kehityksen aikana selkeästi Airplane-luokkaa kohti. Ei ollut järkevää pakottaa kaikkia ohjelman toimintoja ja hakuja Airport-luokan kautta, vaikka näinkin olisi ollut mahdollista tehdä. Toisin sanoen ohjelmassa lentokone ohjaa lentokoneen toimintoja, mikä eroaa suunnitellusta “lentokenttä ohjaa kaikkea”- mallista.

Lentokoneet eivät tällä hetkellä myöskään ole rakenteellisesti erilaisia (trait-rakennetta ei käytetä), koska koneiden erilaisuus muodostuu lähinnä tarvittavasta kiitoradan pituudesta. Yhdeksi kehityskohdaksi jääkin, ettei pelissä ole muita kuin matkustajakoneita.

Flight-luokka, vaikkakin pieni, omaa ison roolin. Se määrittää lentokoneen matkustajamäärän ja matka-ajan sekä mahdollistaa jatkolentojen määrittämisen. Flight-luokkaa käytetään myös yksittäisen lentokoneen tunnistamisessa.

Queue-luokka on edelleen toteutettuna abstraktina, vaikka jonoja onkin lopullisessa toteutuksessa vain yhden tyyppisiä. “Maajonot” jäävät jatkokehityksen tehtäväksi, jos tulevat ajankohtaisiksi ikinä, sillä niiden tarve pelissä on mielestäni pieni.

Käyttöliittymä ja samalla ohjelma aloitetaan Launcher-objektilla. Launcher-objekti vastaa siitä että oikea ikkuna näytetään oikeaan aikaan ja, että Creator-luokka saa oikean tiedoston, josta se luo lentokentän.

Creatorilla on ohjelmassa tärkeä rooli. Se sisältää JSON-parserin ja metodit lentokentän, lentokoneiden ja lentojen luomiseen. Creator vastaa myös randomatisoinnista, joka on tärkeä osa ohjelmaa, sillä kaikki lentokoneiden polttoaineen kulutuksesta lentojen määränpäähän on satunnaista.

Launcherin lisäksi tärkeimmät GUI-luokat ovat StartScreen (aloitusikkuna), MainGame (peli-ikkuna) ja EndScreen(lopetusikkuna). StartScreen määrittää vaikeustason levelChosen muuttujan avulla ja tämän jälkeen MainGame ottaa luodun Airport-olion parametrikseen ja

peli käynnistyy. MainGamen Timer-olio kutsuu Airport-olion onTick-metodia, mikä saa pelin pyörimään. Lopulta EndScreen ottaa jo lopullisessa tilassa olevan Airport-olion parametriksi ja näyttää pelin lopputuloksen.

Muut GUI-luokat ovat erilaisia ponnahdusvalikoita ja tekstiruutuja tai paneeleita, jotka ovat omina luokkinaan vain koodin selkeyttämiseksi.

Algoritmit

Lennonjohtopeli ei sisällä monimutkaisia algoritmeja. Tärkeimmät algoritmit ovat kuitenkin ehkä pistelaskualgoritmi ja korkeudenmuutosalgoritmi.

Pistelaskenta algoritmi on hyvin yksinkertainen: jokaisesta myöhästymisminuutista annetaan 10 pistettä miinusta ja jokainen ylimääräinen polttoainelitra on pluspiste. Eli kaavana

```
minutes * (-10) + fuelLitres = pointsPerLanding
```

Myös korkeudenmuutosalgoritmi on yksinkertainen. Mikäli koneen tulee nousta lisätään jokaisella tikillä koneen korkeuteen 10 metriä ja taas vähennetään 10 metriä, jos koneen tulee laskeutua.

Myös koneen polttoaineen kulutus voidaan tulkita algoritmiksi. Joka minuutti lentokoneen polttoaineesta vähennetään lentokoneelle uniikin polttoainetta/minuutti-arvon verran polttoainetta.

Tietorakenteet

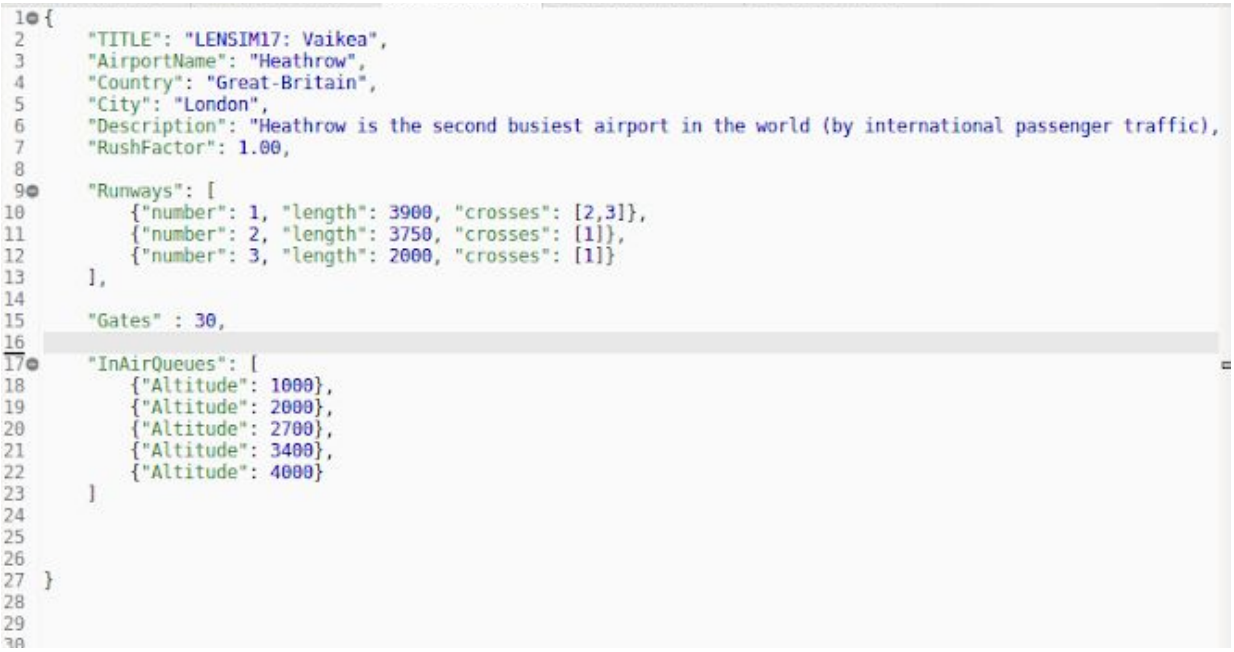
Ohjelmassa käytetään sekaisin muuttuvatilaisia ja muuttumattomia tietorakenteita. Bufferia käytetään esimerkiksi lentokoneiden varastoimiseen Airport-luokassa, koska koneiden määrä muuttuu jatkuvasti ohjelma suorittamisen aikana. Toisaalta ohjelmassa käytetään paljon muuttamattomia tietorakenteita, kuten Vectoreita ja Optioneja.

Vectoreita käytetään aina, kun varastoidaan muuttumattomia listoja, kuten kiitoratoja tai portteja. Optionit ovat käytössä, koska ohjelma sisältää paljon olioita, jotka varastoivat tietoa toisista oliosta vain hetkellisesti. Tällaisia ovat esimerkiksi Runway- ja Gate-luokat, jotka pitävät lentokonetta Optionissa, mikäli lentokone on varannut Runwayn tai Gaten.

Tiedostot

Ohjelma käsittelee kahdentyyppisiä tiedostoja: tekstitiedostoja ja JSON-tiedostoja. Tekstitiedostot sisältävät joka rivillä vain yhden sanan, joka on joko kaupungin tai lentoyhtiön nimi riippuen tiedostosta. JSON-tiedosto on hieman monimutkaisempi.

JSON-tiedosto määrittää lentokentän ominaisuudet.



```
{
    "TITLE": "LENSIM17: Vaikea",
    "AirportName": "Heathrow",
    "Country": "Great-Britain",
    "City": "London",
    "Description": "Heathrow is the second busiest airport in the world (by international passenger traffic),
    "RushFactor": 1.00,

    "Runways": [
        {"number": 1, "length": 3900, "crosses": [2,3]},
        {"number": 2, "length": 3750, "crosses": [1]},
        {"number": 3, "length": 2000, "crosses": [1]}
    ],

    "Gates" : 30,

    "InAirQueues": [
        {"Altitude": 1000},
        {"Altitude": 2000},
        {"Altitude": 2700},
        {"Altitude": 3400},
        {"Altitude": 4000}
    ]


}
```

- RushFactor määrittää, millä todennäköisyydellä lentokone luodaan joka minuutti.
- Runways-lista sisältää tiedot kiitoradoista olioina ja uuden runwayn pystyy lisäämään helposti lisäämällä uuden olion.
    - Määrittelyjen kanssa on oltava tarkka: mikäli numero 1 on ristissä numero 2:n kanssa täytyy numero 2:n olla ristissä numero 1:n kanssa
- Gates kertoo porttien määrän.
- InAirQueues-lista sisältää jonot ja uuden jonon voi lisätä lisäämällä olion listan sisään

Muutoin JSON-tiedosto on suhteellisen selkeä.

Testaus

Suunniteltaessa ohjelmaa oli tarkoitus testata yksikkötesteillä. Kuitenkin suurin osa ohjelman kehityksestä keskittyi siihen, että ohjelman käyttöliittymä toimisi. Tämä johti siihen, että ohjelmaa testattiin lähinnä käyttöliittymän kautta ja sen ehdoin. Ennen käyttöliittymän ohjelmointia peliä testattiin tulostamalla konsoliin tarvittavat tiedot. Näin tehtiin, koska se oli nopeaa ja antoi nopean palautteen. Tässä vaiheessa testaaminen tosin keskittyi pitkälti vain JSON-parserin testaamiseen.

Voidaankin sanoa, että testauksen suunnittelun suurin puute oli kokemattomuus käyttöliittymien rakentamisessa ja testaamisessa.

Ohjelman tunnetut puutteet ja viat

Mitä tulee tehtävänantoon, ei ohjelmassa oikeastaan ole puutteita. Alkuperäiseen suunnitelmaan verrattaessa puutteita kuitenkin on:

- Kaikki lentokoneet ovat matkustuslentokoneita (rahtikoneet puuttuvat)
- Sääolosuhteet eivät vaihdu
- Maajonot puuttuvat
- Käyttöliittymästä ei tullut täysin suunnitelman mukainen, koska ainakaan omilla taidoilla en saanut ScrollPanelia toimimaan päivitysten kanssa tarpeeksi sulavasti
    - Tämä on johtanut siihen, että käyttöliittymä pystyy näyttämään vain tietyn määrän koneita kerralla.
    - Puutetta on yritetty korjata tarjoamalla useita lentokonenäkymiä
- Vaikka en alunperin halunnut keskittyä animoituun graafiseen käyttöliittymään voidaan sellaisen puuttumista pitää puutteena
- Yleisesti pienenä puuttena voidaan pitää myös sitä etten saanut pelin kulusta sulavampaa kuin se on. Koneet laskeutuvat ja nousevat hieman kankeasti.

Vikoja ohjelmasta löytyy muutama:

- Klikatessa esiin tuleva ponnahdusikkuna ei joka klikillä tule esiin. Tämä todennäköisesti on jonkinlainen päivitysongelma, mutta en ole sitä saanut korjattua

Plussat ja miinukset

+ Olen mielestäni onnistunut luomaan käyttöliittymän joka antaa pelaajalle yhdellä ruudulla paljon käytännöllistä informaatiota
+ Lentokoneiden ja lentojen satunnaisuus toimii mielestäni hyvin. Kahta samanlaista lentoa ei lentokentälle kovinkaan todennäköisesti saavu ja arvot ovat tästä huolimatta rajoitettu realistisiksi. (Kuopion ja Helsingin tapauksissa kiitoratojen tiedot on otettu suoraan esikuvistaan)
+ Peli on pelaajalle suhteellisen reilu. Tappion säännöt ovat selkeät ja vaikka peli ei täysin tekstipohjaisena välttämättä innosta kovinkaan montaa ihmistä, yrittää se olla uskollinen tekstipohjaisten simulaatiopelien genrelle.
- Pelin graafinen käyttöliittymä on hieman kömpelö. Vaikka olenkin sitä mieltä, että GUI näyttääkin paljon informaatiota, on siinä liikkuminen hieman kömpelöä. Tähän vaikuttaa myös ScrollPanelin sopimattomuus jatkuvasti päivittyvän sisällön kanssa käytettäväksi
- Koneiden laskeutuminen ja nouseminen on hieman kömpelöä.

Poikkeamat suunnitelmasta

Alkuperäisessä suunnitelmassa oli mukana luokat Notification ja LandQueue. Näitä ei kuitenkaan toteutettu ajanpuutteen ja vaihtoehtoisten ratkaisujen vuoksi.

Vaikka toteutusjärjestys meni pitkälti alkuperäisen suunnitelman mukaan oli ajankäyttösuunnitelma aivan liian optimistinen. Graafisen käyttöliittymän ohjelmoimiseen meni noin tuplasti se aika, mitä olin suunnitellut. Tämä tarkoittaa kymmeniä työtunteja.

Swingiä ei selkeästi oltu suunniteltu siihen käyttöön, mihin minä sitä halusin käyttää ja huonosti tehty virallinen dokumentaatio vaikeutti käyttöliittymän ohjelmointia.

Toteutunut työjärjestys ja aikataulu

1. Projektin aloitus 22.02.18
2. JSON-Parser on valmis 04.03.18 aikaa käytetty 10h
3. Creator-luokka pystyy parseria käyttämällä luomaan Airport-olion 15.03.18 14h
4. Airport ja Airplane ovat lähelle valmiita /cratePlane on lähelle valmis 25.03.18 12h
5. GUI-kehityksen aloitus 25.03.18
6. Ponnahdusvalikot lisätty 09.04.18 16h
7. ScrollPanel ei toimi oikein. 15.04.18 12h
8. Tehdään vaihtohtoisratkaisu ScrollPaneleille 16.04.18 8h
9. Koneiden ja lentojen randomisaatio toimii. Koneet voivat törmätä 25.04.18 16h
10. Projektin deadline 26.04.18
11. Alkuvalikon ja loppuikkunan lisääminen. Launcherin muokkaaminen useaa ikkunaa tukevaksi. NextFlights toimii 27.04.18 12h
12. Projekti on pieniä korjauksia ja dokumenttia vailla valmis 04.05.18 12h

Huom: Ajat ovat erittäin karkeita arvioita. GUI käytetty turha 20h olisi voitu käyttää järkevämpiin asioihin ja tällöin jopa alkuperäiseen kurssin aikatauluun olisi saatettu ehtiä.

Arvio lopputuloksesta

Olen loppujen lopuksi suhteellisen tyytyväinen lopputulokseen, vaikka sovellus valmistuukin reilun viikon myöhässä annetusta aikarajasta. Koodi voisi olla paljon selkeämpää ja ohjelmointityyli voisi olla konsistentimpi, mutta työssä ei ole ohjeistukseen verratessa huomattavia puutteita.

Ohjelmaan jäi kuitenkin erittäin paljon parannettavaa. Peliin pystyisi lisäämään esimerkiksi animoitua graafiikkaa, mikä tekisi pelistä heti mukaansatempaavamman. Lisäksi pelin graafinen käyttöliittymä kannattaisi kokonaan siirtää Swingiltä käyttämään joitain toista kirjastoa, sillä Swing ei voi olla paras mahdollinen grafiikkakirjasto tämäntyyppisen sovelluksen mallintamiseen.

Peliin voisi lisätä myös sääolosuhteiden vaihtumisen tai muita erityistilanteita. Nämä mainittiinkin jo alkuperäisessä tehtävänannossa ja ohjelman luokkajako ja muu rakenne on tehty siten, että nämä lisäykset olisi suhteellisen helppoa laittaa ohjelmaan. Tämä johtuu siitä, että Runway-oliota pystyy ohjaamaan sen ulkopuolelta ts. vain Airport-luokkaa muokkaamalla voidaan lisäominaisuudet saada toimimaan. Toisaalta taas graafinen käyttöliittymä on jo nykyisessä muodossaan paisunut peli-ikkunan osalta erittäin epäselväksi verkoksi, jossa lisäyksien tekeminen saattaa olla vaikeaa ja joskus jopa mahdotonta.

Valitsin tämän projektin kuitenkin juuri siksi, että halusin haastaa itseäni ja pakottaa itseni oppimaan lisää graafisen käyttöliittymän luomisesta. Siihen tavoitteeseen olen päässyt ja siitä voin olla tyytyväinen.

Viitteet

- **Scala Docs** http://docs.scala-lang.org/
    - **Scala API** https://www.scala-lang.org/api/current/
    - **Scala Swing API** https://www.scala-lang.org/api/current/scala-swing/scala/swing/index.html
- **Java Swing Tutorial** https://docs.oracle.com/javase/tutorial/uiswing/
- **Scala Parser** http://central.maven.org/maven2/org/scala-lang/modules/scala-parser-combinators_2.12/1.1.0/
- **Scala OTFried** http://otfried.org/scala/index.html
- **Various** https://stackoverflow.com/
- **Wikipedia lentokenttien ja lentokoneiden tietoja varten** https://en.wikipedia.org/

Liitteet

- **UML kaavio**
- **Projektin koodi**