5G00EI63-3002 Ohjelmoinnin perusteet

Tekijä: Lari Liuhamo, 19TIETOB
Aihe: Brainfuck-ohjelmointikielen tulkki
Palautettu: 15.12.2019
Lähdekoodi: https://gitlab.tamk.cloud/Diapolo10/BF-Interpreter/tree/master

Esipuhe

Tämä on luultavasti jo aiemmin tullut selväksi, mutta ohjelmointi ei ole minulle mikään uusi asia. Suurin osa tämän alkeiskurssin materiaalista oli asiaa, jonka tiesin jo entuudestaan juurta jaksaen, ja loput saatoin olettaa muiden kielten kokemuksen perusteella, joten en kieltämättä paljoakaan uutta lopulla tunneilla oppinut. Mutta tämä ei ole opettajien syytä!

Tästä syystä palauttamani tehtävät ovat usein olleet astetta kehittyneempiä kuin mitä niiden tarvitsisi olla. Esimerkiksi piilaskuri oli mielestäni aivan liian yksinkertainen, joten koska en ole vielä kovin tottunut monisäikeisten ohjelmien luomiseen, päätin kokeilla, jos saisin sen kirjoitettua käyttämään kaikkia prosessorin säikeitä – ja onnistuihan tämä, vieläpä yllättävän helposti ilman selkeää huolta datakisatilanteista.

Mutta nyt onkin lopputyö kyseessä, joten halusin ottaa tilaisuudesta vaarin ja yrittää luoda jotakin sen verran haastavaa, että minullakin olisi jotakin uutta opittavaa.

Ensimmäinen yritys

Ideoiden keksiminen oli kieltämättä vaikeaa. En halunnut käyttää annettuja esimerkkiohjelmia, koska ne olivat aivan liian yksinkertaisia, ja monet muut vähemmän kokeneet luultavasti käyttäisivät niitä pohjina (joskin on toki mahdollista, että kaikilla muilla oli sama ajatus ja lopulta kukaan ei ole niitä käyttänyt). Minulta meni noin viikko ennen kuin aloin työstämään ensimmäistä ideaani; Tetris-peli komentorivillä. Pääsin noin puoleenväliin projektia, kun ongelmaksi muodostui käyttöliittymän piirtäminen komentorivillä. Microsoftin dokumentaatio Win32 API:lle ei ollut loppujen lopuksi kovinkaan hyödyllinen esimerkkien puutteen vuoksi, ja en halunnut yrittää käyttää konsolikomentoa ”cls” joka kerta kun konsoli piti päivittää. Loppujen lopuksi hylkäsin idean, kun lopputyön deadline alkoi lähestyä ja vakuutuin, että tämä projekti vaatisi aivan liikaa aikaa.

Toinen, ja onnistunut, yritys

Siispä palasin takaisin piirustuspöydän ääreen. Tarvitsin nyt pikapikaa uuden idean, jossa olisi riittävästi haastetta, mutta joka olisi myös ajan puitteissa toteutettavissa. Ideoita ei meinannut aluksi syntyä lainkaan, mutta ottaessani osaa Advent of Code 2019 -projektiin ja törmätessäni tehtävään tulkista, joka muistutti minua assemblykielistä, sain päähäni yrittää luoda tulkin Brainfuck-ohjelmointikielelle. Miksi BF? Sitä on erittäin yksinkertainen parsia, ja uskoin sen yksinkertaisuuden riittävän juuri ja juuri, jotta ehtisin saamaan jotakin aikaan.

Otin tavoitteekseni saada ohjelma suorittamaan Wikipedia Brainfuck-kielen sivulla löytyneen ”Hello World”-ohjelman, joka sisälsi ylimääräisen kommenttisilmukan ohjelman alussa. Jos ohjelmani pystyisi suorittamaan sen, se pystyisi suorittamaan minkä tahansa standardin BF-ohjelman.

Otin lisähaasteeksi tavoitteen käyttää pelkkää standardikirjastoa, ja käyttäen ainoastaan kurssilla opittuja asioita (plus nimiavaruuksia luettavuuden parantamiseksi). Suurin osa ohjelman kirjoittamisesta sujui hyvin,

vaikka minulla ei aiempaa kokemusta tulkkien ja kääntäjien luomisesta olekaan, mutta silmukoiden kanssa kärsin pitkään erinäisten bugien kanssa. Sain vasta eilen käytyä koko ohjelman läpi Visual Studion debuggerilla ja ymmärsin ongelmien lähteen, sain sen korjattua ja sain lopulta odottamani tulosteen. Tämän jälkeen siivosin ohjelmaa, lisäsin kommentteja ja valmistelin komentoriviparametrien tuen.

Ohjelman käyttö

Brainfuck-tulkkini pitäisi olla käännettävissä millä tahansa C++17-standardia tukevilla kääntäjillä, mutta olen testannut sen vain Visual Studion MSVC-kääntäjällä. Käännöksen jälkeen ohjelmalle syötetään haluttu Brainfuck-kielinen lähdekooditiedosto komentorivillä. Valinnaisesti voidaan syöttää toinen tiedostonimi, johon ohjelman ulostulo kirjoitetaan.

```
bf_interpreter.exe -i ”hello_world.bf” -o ”hello_world.txt”
```

Ohjelman mukana tulee erilaisia Brainfuck-ohjelmia, joita voidaan suorittaa tulkilla. Lisäksi ohjelma tukee käytännössä kaikkia ohjelmia, joita voi luoda osoitteessa: https://copy.sh/brainfuck/text.html

Ohjelman toiminta

Brainfuck-kielen standardi on hieman hatara, ja monet eri ohjelmat vaativat hieman eri asioita, mutta pääpiirteittäin kieli on määritelty näin (ja ohjelmani käyttäytyy näin):

Oletetaan, että meillä on ääretön nauha soluja, joka alkaa nollasta. Tämän nauhan solut voivat sisältää lukuja 0-255, ja lukujen ylivuoto on odotettua käytöstä (0-1 == 255, 255+1 == 0). Meillä on myös osoitin, joka osoittaa senhetkiseen soluun ja on alussa nolla. Ohjelmakoodia luetaan vasemmalta oikealle, ja se voi sisältää kahdeksaa eri operaattoria:

1. ’+’ lisää tämänhetkiseen soluun luvun 1
2. ’-’ vähentää tämänhetkisestä solusta luvun 1
3. ’>’ siirtää osoittimen seuraavaan soluun nauhalla, ja luo uuden solun, jos sitä ei vielä ole olemassa
4. ’<’ siirtää osoittimen edelliseen soluun nauhalla; standardi ei määrittele, mitä tapahtuu, jos osoitin menee negatiiviseksi, joten tämä on määrittelemätöntä käytöstä ja saattaa kaataa ohjelman
5. ’[’ aloittaa silmukan, jos osoittimen osoittama solu on muuta kuin nolla. Muussa tapauksessa silmukan sisältö jätetään huomiotta
6. ’]’ päättää silmukan, ja siirtää osoittimen vastaavaan silmukan alkuun, jos sisältö on huomioitu. Muussa tapauksessa silmukka päättyy
7. ’.’ tulostaa senhetkisen osoittimen osoittaman solun arvon merkkinä
8. ’,’ ottaa käyttäjältä kirjaimen tai merkin, ja varastoi sen senhetkiseen soluun

Aikaa säästääkseen, tulkkini parsii ensin Brainfuck-ohjelmakoodista kommentit pois, jolloin jäljelle jää vain ohjelman operaattorit. Tämän jälkeen ohjelmani luo sisäisen kuvauksen tästä Turingin koneesta, ja alkaa muokkaamaan koneen tilaa perustuen ohjelman käskyihin. Lopulta tulkki palauttaa koneen viimeisen tilan, jonka nauhan tiedot tulostetaan näytölle.