myGrep- harjoitustyö

# Suunnitelma

Yleinen työn suunniteltu rakenne:

Toteutan ohjelman aliohjelmilla, jotka nimetään inkrementtien mukaan. Ensimmäisen inkrementin toiminnoista huolehtii siis aliohjelma inkrementti1, toisesta inkrementti2 jne. Tämä helpottaa työn ja kokonaistoimivuuden arviointia. Mikäli inkrementti- aliohjelmien sisällä tarvitsen aliohjelmia, luon niitä sitä mukaan. Näin main()- osio pysyy puhtaana, ja ohjelmaa on mahdollisimman helppo lukea.

1.inkrementti:

Toteutin ensimmäisen inkrementin tehtävänantopäivänä 8.2. Ohjelmani alku sijoittaa käyttäjän antaman kokonaismerkkijonon, ja merkkijonosta etsittävän lyhyemmän merkkijonon kahteen string- tyyppiseen muuttujaan. Tämän jälkeen for-loopilla merkkijono käydään läpi muistipaikka kerrallaan, ja etsitään vastaavuutta haettavalle merkkijonolle. Mikäli vastaavuus löytyy, annetaan for-loopin juosta normaalisti loppuun, ja kasvatetaan haettavan merkkijonon muistipaikkaa yhdellä. Tästä en tehnyt erillistä suunnitelmaa, koska tehtävä oli verrattain helppo.

2. inkrementti:

Toisessa inkrementissä täytyy ohjelma muokata käyttämään komentorivillä ajettavia komentoja. Lisäksi ohjelman täytyy vastaanottaa ja lukea sille annettu tiedosto, ja suorittaa käyttäjän antamalla merkkijonolla etsintä kyseiseen tiedostoon. Jos yksi tai useampi vastaavuus tiedoston riveiltä löytyy, tulee ohjelman tulostaa kyseiset rivit terminaaliin. Toteutan tämän tunneilla opittuja asioita käyttäen, eli tuodaan <fstream> mukaan ohjelmaan. Luodaan ifstream tyyppinen muuttuja, jolle annetaan argv komentosyötteeltä tiedoston nimi. Käytän mitä luultavimmin while- looppia ja getlineä tiedoston lukuun, eli luodaan toinen muuttuja, johon getline rivi kerrallaan sijoittaa tekstiä. Toinen vaihtoehto olisi tallentaa koko tiedoston sisältö string- tyyppiseen muuttujaan, mutta tämä ei olisi viisasta jos kuvitellaan että käsiteltävä tiedosto olisi isompi. Käytän myös luultavasti .find- toimintoa, mikä tekee käyttäjän syöttämän avainsanan etsimisestä helppoa.

3. inkrementti:

Tässä pystytään käyttämään toiselle inkrementille suunniteltua perustaa. Nyt lisäyksenä tulee toiminnot joita käyttäjä voi halutessaan valita. Toimintojen sisäänotto indikoidaan ”-o” komennolla, jonka perään voi valita seuraavia toimintoja:

”-ol” = Näyttää rivinumerot

”-oo” = Näyttää kuinka monta kertaa haettava sana löytyy tiedostosta

”-olo” = Kahden ensimmäisen yhdistelmä

Ensimmäinen vaisto olisi yksinkertaisesti kirjoittaa if- lausekkeilla jokaiselle kirjainyhdistelmälle määritelmä, mutta inkrementti 4:ssä mukaan tuodaan lisää toimintoja, tarkoittaen että kirjainyhdistelmiä olisi todella iso määrä. Tällä tavalla kirjainten täytyisi myös esiintyä tietyssä järjestyksessä jotta ohjelma osaisi toimia. Tämän sijaan teen ohjelman alkuun structin joka sisältää kirjaimia edustavat toiminnot, ja tarkastaa mikä toiminto milloinkin on vaadittu. Tähän saatan tarvita erillisen aliohjelman, joka tarkistaa arvot ja palauttaa ne inkrementti- aliohjelmalle, jossa säilyttäisin pelkästään toimintoja vastaavat mekanismit.

4. inkrementti

Lisään structiin loput toimintomuodot. Aikaisemmassa inkrementissä oli ongelmaa structin välittämisessä checkOptions()- aliohjelmalle, joka tarkistaa toiminnot. Paljastui että olin sijoittanut aliohjelman out-of-scopeen. Hieno muistutus kahden tunnin pähkäilyn jälkeen että järjestyksellä on väliä!

Muokkaan koodia hieman, niin että kaikki kirjainyhdistelmät toimivat missä järjestyksessä tahansa. checkOptions()- pitää huolen että ohjelma lukee vasta ”-o” options-indikoinnin jälkeisiä muistipaikkoja. Eli yhdistelmästä ”-oilor” saadaan irti i, o, l ja r.

# Ohjelman rakenne

Työ valmistui 17.2. Ohjelma koostuu nyt main()- osiosta ja neljästä aliohjelmasta, increment1(), increment2(), increment3\_4 ja optionsCheck(). Mainissa luetaan argc- argumenttimäärä, jonka mukaan ohjataan käyttäjä oikeaan osoitteeseen.

* argc == 1 ohjaa käyttäjän myGrep- perustoimintoon.
* argc == 3 ohjaa increment2 aliohjelmaan, jossa luetaan tiedosto ja etsitään asetettua hakusanaa
* argc == 4 ohjaa inkrementti3\_4 aliohjelmaan, jossa kaksi viimeistä inkrementtiä. Eli tämä on ohjelman laajin osa, jossa käyttäjä valitsee halutessaan toimintoja.
* jos argc suurempi kuin 4, tai == 2, tulostetaan käyttäjälle käyttöohje.

Jos tiedoston luku jostain syystä epäonnistuu, palautetaan aliohjelmilta pääohjelmaan errorin merkiksi arvo -1, indikoiden virhettä.

Aliohjelmat onnistuivat pääpiirteittäin alkuperäisen suunnitelman mukaan. Alla vielä raakarunko.

int main()

{

if(argc == 1)

increment1();

if(argc == 3)

increment2();

if(argc == 4)

increment3\_4();

if(argc == 2 || argc > 4)

\*tulostetaan käyttöohje\*

}

incr1(){…}

incr2{…}

incr3\_4(){

checkOptions();

if(struct.lineNumber)

\*toiminnon määritelmä\*

etc...

}

# Itsearviointi

Toteutin työn mielestäni hyvin. Suunnittelin pääpiirteittäin työn kulun ennen koodin kirjoittamista, tämä helpotti huomattavasti työn puhtaaksi kirjoittamista. Työn modulaarisuus on myös mielestäni hyvällä tasolla, sillä työ on yksinkertaisesti jaoteltu aliohjelmiin, jotka sisältävät annetut inkrementit. Modulaarisuus tekee myös helpoksi mahdolliset korjaukset tarkastusvaiheessa. Kommentoin työtäni myös ahkerasti. Nyrkkisääntönä: jos kommenttia ei ole, on koodirivi itsensä selittävä. Koodia kertyi 231 riviä, joista n. 40 on kommenttirivejä. Työssä on myös vältetty turhaa itsensä toistoa, sekä sisäkkäisiä komentorakenteita, jotka vaikeuttaisivat lukua.

Toteutin arvosanaan 5 vaadittavat inkrementit, ja implementoin ne työhön mielestäni arvosanan 5 mukaisesti. Olen todella tyytyväinen työn lopputulokseen, kuitenkaan unohtamatta että aina voi tehdä kaiken paremmin ja siistimmin, ja siihen pyritään.

Työn kulku & työtunnit

|  |  |  |
| --- | --- | --- |
| Pvm | Työtunnit | Karkea kuvaus työnteon sisällöstä |
| 08.02.2023 | 4h | Inkrementti 1 valmistui nopeasti, aloitin inkrementin 2 tekoa. |
| 13.02.2023 | 4h | Inkrementti 2 valmiiksi, seuraavan isomman inkrementin hahmottelua. |
| 15.02.2023 | 5h | Inkrementti 3 valmistui nopeasti, mutta koska inkrementti 4 oli käytännössä inkrementti 3:n parantelua, täytyi aikaa kuluttaa tässä työnkohdassa suhteellisen paljon. Testasin myös aikaisempien inkrementtien toimivuutta ja varmistin että kaikki saumautuu yhteen. Tiedoston lukemisen virhehallinta myös ok. |
| 17.02.2023 | 6h | Viimeinen inkrementti valmis. Raportin teko + muut aineistot. |

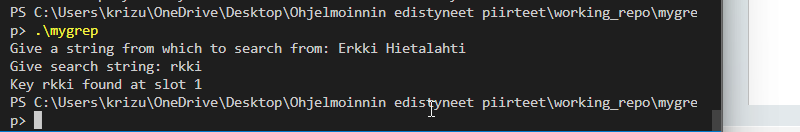
Yhteystiedot

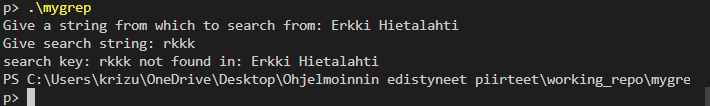
Pyydetyt yhteystiedot: [kristian.sikanen@tuni.fi](mailto:kristian.sikanen@tuni.fi), 0442060112

github repo: https://github.com/Krisbis/mygrep.git

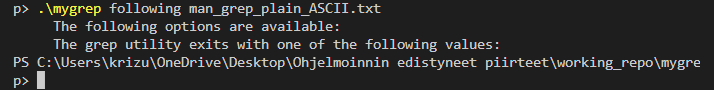
Todisteet ohjelman toimivuudesta

Inkrementti 1

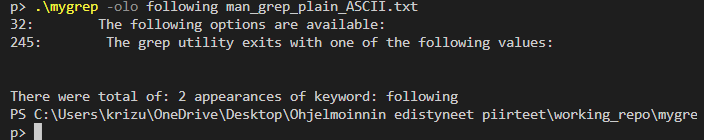




Inkrementti 2



Inkrementti 3



Inkrementti 4

kaikki toiminnot yhdessä. Following- sanan sisältävä rivi 32 on pyyhitty pois

Text

Description automatically generated

case-insensitiveness

Text

Description automatically generated

Virheenkäsittely:

Text

Description automatically generated

reverse-toiminto, kaikki e- kirjainta sisältävät rivit pyyhitty pois.

Graphical user interface, text, application

Description automatically generated

Lähdekoodi

#include <iostream>

#include <fstream>

#include <string>

#include <algorithm>

using namespace std;

struct optionsStruct

{

    bool occurence = false;  // Number of occurences

    bool lineNumber = false; // Linenumbers

    bool reverse = false;    // deletion of the keyword containing line

    bool insens = false;     // case insensitiveness

};

void increment1();

int increment2(int, char \*[]);

int increment3\_4(int, char \*[], optionsStruct);

void optionsCheck(string, optionsStruct &);

int main(int argc, char \*argv[])

{

    optionsStruct method{false, false, false, false};

    if (argc == 1)

    {

        increment1();

    }

    if (argc == 3)

    {

        increment2(argc, argv);

    }

    if (argc == 4)

    {

        increment3\_4(argc, argv, method);

    }

    // Lukee argc, jos cmd- syote yli 4 (tai == 2) argumenttia pitkä, tulostetaan ohje

    if (argc > 4 || argc == 2)

    {

        cerr << "\nUsage: mygrep keyword file\n";

        cerr << "\noptional: mygrep ('options') keyword file";

        cerr << "\n-ol  -> Displays only line numbers";

        cerr << "\n-oo  -> Displays only number of occurences";

        cerr << "\n-oi  -> Displays only number of occurences (case-insensitive)";

        cerr << "\n-or  -> Displays every line that doesn't include the keyword";

    }

    return 0;

}

void increment1()

{

    //\*--------inkrementti 1----------\*//

    string inString;

    string searchStr;

    int lapcount = 0;

    cout << "Give a string from which to search from: ";

    getline(cin, inString);

    cout << "Give search string: ";

    getline(cin, searchStr);

    // For- luuppi, joka luuppaa syotetyn string- muuttujan sisaltaman merkkimaaran verran

    // Jos ensimmaisen merkin vastaavuus loytyy, alkaa muuttuja lapcount kaymaan lapi seuraavia tulevia merkkeja

    // Kun taysi vastaavuus loydetty, nollataan lapcount tulevien vastaavuuksien varalle

    for (int i = 0; i < inString.size(); i++)

    {

        if (inString[i] != searchStr[lapcount])

        {

            lapcount = 0;

        }

        if (inString[i] == searchStr[lapcount])

        {

            lapcount++;

        }

        if (lapcount == searchStr.size())

        {

            cout << "Key " << searchStr << " found at slot " << (i - searchStr.size() + 1) << "\n";

            lapcount = 0;

        }

    }

}

int increment2(int argc, char \*argv[])

{

    //\*--------inkrementti 2----------\*//

    // Luodaan muuttuja jolle sijoitetaan hakusana, avataan tiedosto ja luetaan sisalto

    // Jos tiedostoa ei saada auki syystä tai toisesta, palauttaa errorin

    string keyword = argv[1];

    ifstream file(argv[2]);

    if (!file)

    {

        cerr << "Error opening file\n";

        return -1;

    }

    // Luodaan muuttuja johon sijoitetaan tiedoston tekstirivi

    // Luupataan lopetusehdolla -> 'niin pitkaan kun tekstia loytyy'

    // Jos ohjelma loytaa vastaavuuden, tulostetaan se naytolle

    // Npos -> maksimikoko size\_t:eelle -> size\_t on sizeof- operaattorin palauttama tyyppi

    //- toimii siis tarkastuksena tulostukselle

    string line;

    bool notFound = true;

    while (getline(file, line))

    {

        if (line.find(keyword) != string::npos)

        {

            cout << line << endl;

            notFound = false;

        }

    }

    if (notFound)

    {

        cout << "\nRequested keyword not found in this file 1\n";

    }

    return 0;

}

void optionsCheck(string option, optionsStruct &method)

{

    // Tama aliohjelma passaa increment3\_4 aliohjelmalle tiedon millä metodeilla (options -o) kayttaja haluaa tulosteen

    // Sijoittaa siis structiin referenssillä totuusarvon, josta incement3\_4 aliohjelma tarkastaa mitä tulostetta halutaan

    for (int i = 2; i < option.size(); i++)

    {

        if (option[i] == 'l')

        {

            method.lineNumber = true;

        }

        else if (option[i] == 'o')

        {

            method.occurence = true;

        }

        else if (option[i] == 'i')

        {

            method.insens = true;

        }

        else if (option[i] == 'r')

        {

            method.reverse = true;

        }

    }

}

int increment3\_4(int argc, char \*argv[], optionsStruct method)

{

    //\*--------inkrementti 3----------\*//

    //\*--------inkrementti 4----------\*//

    // Tata kommentoin vahemman, koska kayttaa samoja evaita kuin inkrementti kaksi

    // Tahan aliohjelmaan on myös sisallytetty viimeinen inkrementti

    // Viimeistä inkrementtia koskevat patkat ovat merkittynä ohjelman arviointia helpottamiseksi

    string option = argv[1];

    string keyword = argv[2];

    ifstream file(argv[3]);

    if (!file)

    {

        cerr << "Error opening file\n";

        return -1;

    }

    string line;

    int occurences = 0;   // options: -oo

    int row = 0;          // options: -ol

    bool notFound = true; // Jos haettu keyword ei löydy

    optionsCheck(option, method);

    while (getline(file, line))

    {

        row++; // Muuttuja tallentamaan läpikäydyt rivit.

        // ↓↓-inkrementti4 -- case insenitiveness if chosen--↓↓

        /\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

        if (method.insens)

        {

            transform(line.begin(), line.end(), line.begin(), ::tolower);

            transform(keyword.begin(), keyword.end(), keyword.begin(), ::tolower);

        }

        // ↓↓-inkrementti4 -- reverse if chosen--↓↓

        /\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

        if (method.reverse)

        {

            if (line.find(keyword) != string::npos)

            {

                notFound = false;

                occurences++;

                continue;

            }

            method.lineNumber ? cout << row << ":   " : cout << "";

            cout << line << endl;

        }

        // ↓↓-inkrementti3&4 -- normal flow of options-↓↓

        /\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

        if (line.find(keyword) != string::npos && !method.reverse)

        {

            // options: line numbers

            method.lineNumber ? cout << row << ":   " : cout << "";

            cout << line << endl;

            occurences++;

            notFound = false;

        }

    }

    // Tulostetaan tulos luupin ulkopuolella vaihtoehdoille joissa occurence valittuna

    if (!notFound && method.occurence)

    {

        cout << "\n\nThere were total of: " << occurences << " appearances of keyword: " << keyword << endl;

    }

    // Jos vastaavuuksia ei loydy, niin yksiselitteisesti notFound = true

    // Tarkistaa boolean muuttujan, ja luettujen rivien maaran

    // Jos luetut rivit > 1, tiedetaan että ohjelma pääsi käsiksi tiedostoon ja luki-

    //  sen sisallon loytamatta vastaavuuksia

    if (notFound == true && row > 1)

    {

        cout << "\nRequested keyword not found in this file\n";

    }

    return 0;

}