



Mygrep projekti

Kristian Kähkönen

RAPORTTI
Maaliskuu 2022

Tietotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikka

KÄHKÖNEN, KRISTIAN

Mygrep projekti

Raportti 29 sivua, joista liitteitä 5 sivua
Maaliskuu 2022

Tämä on raportti "Ohjelmoinnin edistyneet piirteet" -kurssin ensimmäisen osan lopputyöstä. Tässä raportissa tarkastellaan, kuinka prosessi sujui ja tarkastellaan projektin lopputulokset.

Asiasanat: C++, ohjelmointi, grep, raportointi,

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering

KÄHKÖNEN, KRISTIAN

Mygrep projekti
Mygrep project

Report 29 pages, appendices 5 pages
March 2022

This is a report of project made for “Ohjelmoinnin edistyneet piirteet” -course first part finale. In this report, the process and the results are reviewed.

Key words: C++, programming, grep, report

SISÄLLYS

1	JOHDANTO	6
2	SUUNNITELMA	7
2.1	Alustava suunnitelma	7
2.2	Tiedon hankinta.....	7
3	TOTEUTUS	8
3.1	Ensimmäinen osuus.....	8
3.1.1	String vertailu.....	8
3.1.2	Ongelmat.....	8
3.2	Toinen osuus.....	8
3.2.1	Tiedoston lukeminen	9
3.2.2	Ongelmat.....	9
3.3	Kolmas osuus.....	9
3.3.1	Komentojen prosessointi	11
3.3.2	Ongelmat.....	12
3.4	Neljäs osuus	12
3.4.1	Try Catch käyttäminen.....	15
3.4.2	Ongelmat.....	15
4	TULOKSET	16
4.1	Ensimmäinen osuus.....	16
4.2	Toinen osuus.....	17
4.3	Kolmas osuus.....	17
4.4	Neljäs osuus	19
5	POHDINTA	22
	LÄHTEET.....	24
	LIITTEET	25
	Liite 1. Lähdekoodi.....	25

LYHENTEET JA TERMIT

TAMK	Tampereen ammattikorkeakoulu
C++	ohjelmointikieli
while silmukka	ohjelmoinnissa käytettävä while silmukka
for looppi	ohjelmoinnissa käytettävä for silmukka
VS	Visual Studio
Git	Tässä dokumentissa GitLab, versionhallinta
If lause	Jos lause

1 JOHDANTO

Tämä raportti kertoo, kuinka tein Mygrep projektin. Käydään prosessi ja tulokset läpi.

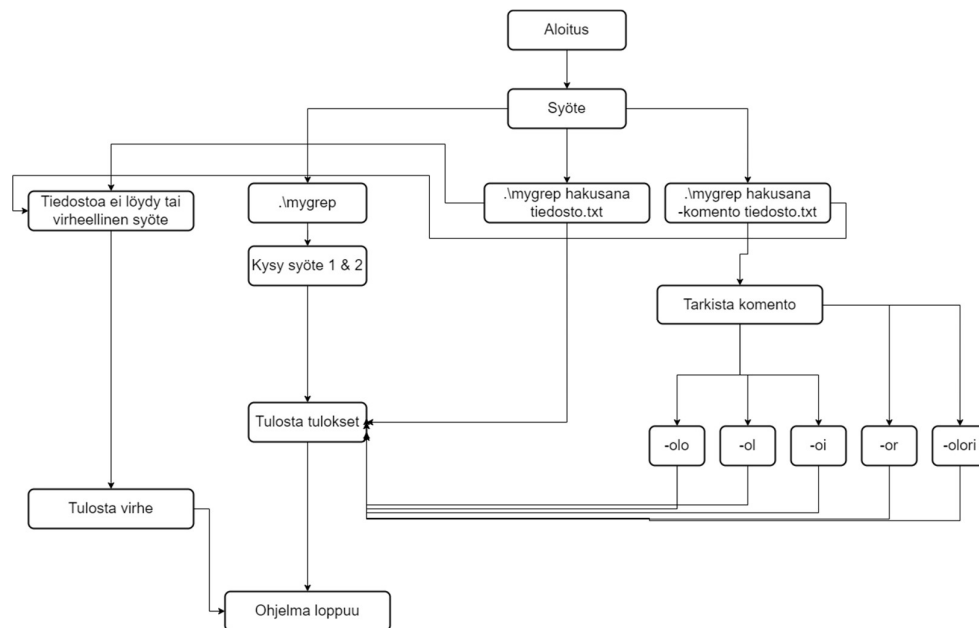
Projekti on tehty itsenäisesti ilman isoja apuja muilta.

Pohjana on edellä mainitun kurssin teorian ja harjoitukset, sekä "C++-ohjelmoinnin perusteet" -kurssilla opitut taidot.

Tiedonhankinnassa käytettiin erilaisia ohjelmointiin liittyviä sivuja, kuten "Stack Overflow" ja "GeeksforGeeks".

2 SUUNNITELMA

2.1 Alustava suunnitelma



KUVA 1. Alustava vuokaavio ohjelmasta.

Ohjelma tehdään paloittain. Aloitetaan ensimmäisestä palasta, jonka jälkeen tulevat muut osuudet.

2.2 Tiedon hankinta

Projektissa oli paljon uutta, johon en ollut perehtynyt. Opin esimerkiksi seuraavat asiat:

Komentoriviargumentit, *stringien* vertailu *find()* komennolla, *stringin* muuttaminen pienille kirjaimille, *try catch* virheen käsittely ja *filesystem* kirjaston perusteita kuten tiedoston koon lukeminen.

Käytin monia erilaisia sivuja tiedonhankinnassa, kuten Stackoverflow. Lähde-sivuilla ovat lähteet, jota hyödynsin tässä projektissa.

3 TOTEUTUS

3.1 Ensimmäinen osuus

Ensimmäinen osuus oli helpoin ja nopein. Ohjelma selitettynä:

tarkistetaan että syötteitä ei ole ohjelman kutsun jälkeen, eli `argc == 1`

anna syöte 1

anna syöte 2

verrataan syötteitä, jos syöte 2 on syöte 1., tulosta että löytyi. jos ei, tulosta ettei löytynyt.

3.1.1 String vertailu

Vertailin tekstisyötteitä *find()* komennolla.

```
int found = userInput.find(userInput2);
```

Found arvo on -1 tai jotain muuta. -1 tarkoittaa, että tulosta ei löytynyt.

Std::string::npos tarkoittaa -1, eli tämän jälkeen *if* lause, jossa tulostetaan, että löytyi:

```
if (found != std::string::npos) {
    // tulosta teksti
}
```

3.1.2 Ongelmat

En kokenut isompia ongelmia tässä kohtaa vielä. Suurin ongelmani oli ymmärtää miten *find()* toimii ja kuinka laitan ”-merkin tulostukseen.

3.2 Toinen osuus

Toinen osuus oli todella samanlainen, joten siinä kului vain hetki.

Avataan käyttäjän haluama tiedosto, ja tulostetaan kaikki halutut rivit *while* silmuksessa.

3.2.1 Tiedoston lukeminen

Esimerkki syöte komentorivillä:

```
.\mygrep following man_grep_plain_ASCII.txt
```

Ensimmäisen sana on ohjelman nimi, toinen on haettava sana, kolmas on avattava tiedoston nimi.

```
std::string searchItem = argv[1];  
file.open(argv[2]);
```

3.2.2 Ongelmat

En kokenut ongelmia tässä osassa.

3.3 Kolmas osuus

Kolmannessa osuudessa tulee nyt syötteeseen komennot. Periaate on sama kuin aiemmassa osuudessa, mutta käyttäjä voi nyt syöttää asetuksia, -oo, -ol ja -olo komentoja. Voimme uudelleen käyttää aiempaa koodia tässä.

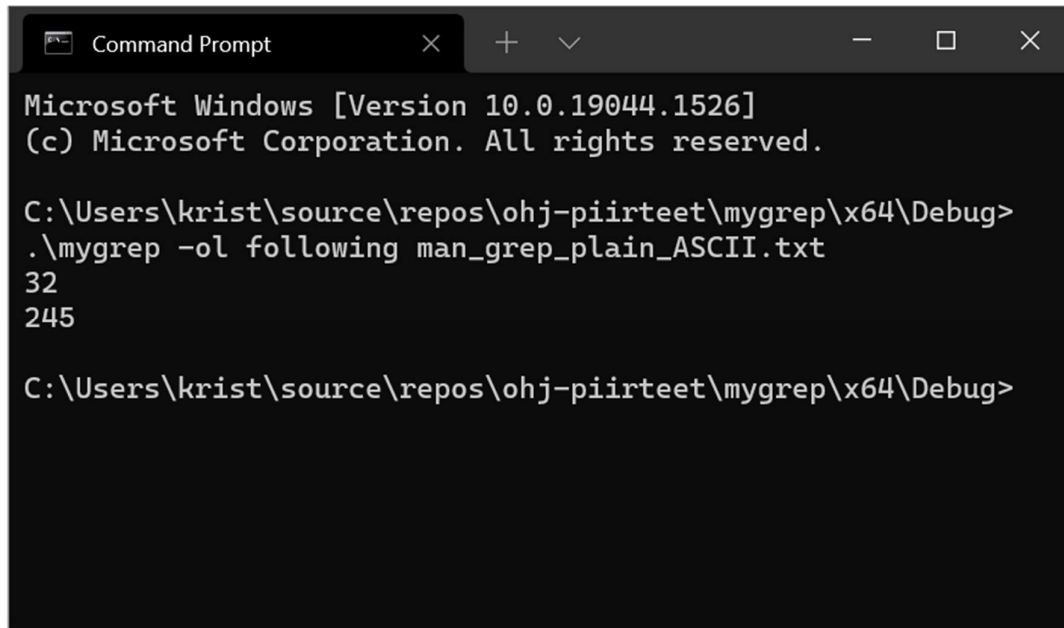
Esimerkki syöte olisi:

```
.\mygrep -olo following man_grep_plain_ASCII.txt
```

Ensimmäisen sana syötteessä on ohjelman nimi, toinen on lisäkomento, kolmas on haettava sana, neljäs on avattava tiedoston nimi.

Lisäkomentojen alussa -o tarkoittaa *options* eli asetukset.

Jos komento on *-ol* eli *line numbering* saa ohjelman tulostavan rivien edessä rivinumerot.



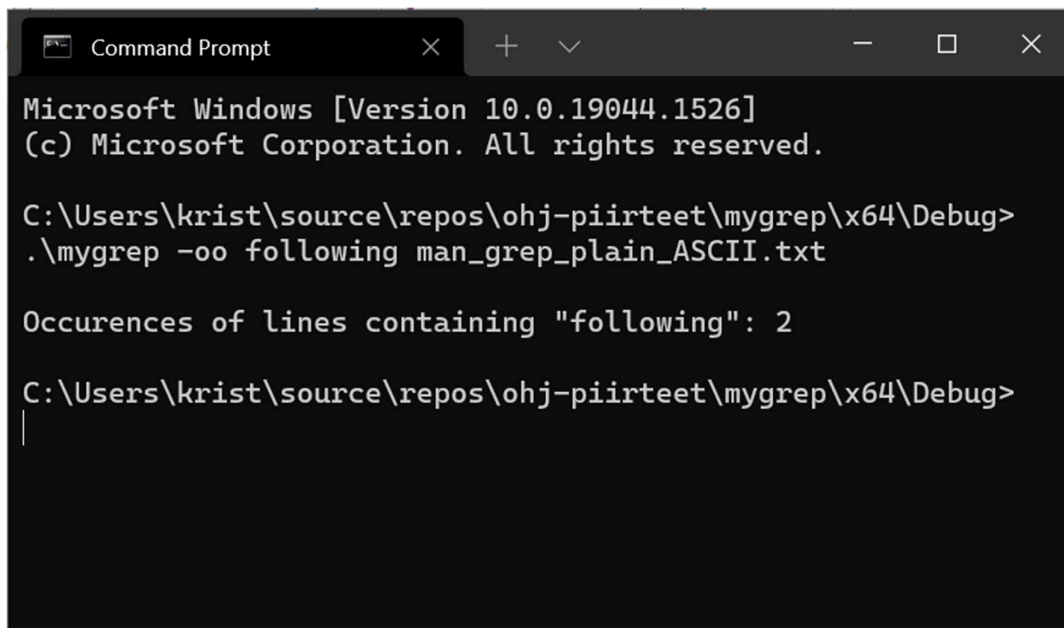
```
Command Prompt
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
.\mygrep -ol following man_grep_plain_ASCII.txt
32
245

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 2. Esimerkkiajo *-ol* -komennolla.

Jos komento on *-oo* eli *occurrences* tulostaa hakusanan ilmestyneiden rivien lukumäärät.



```
Command Prompt
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

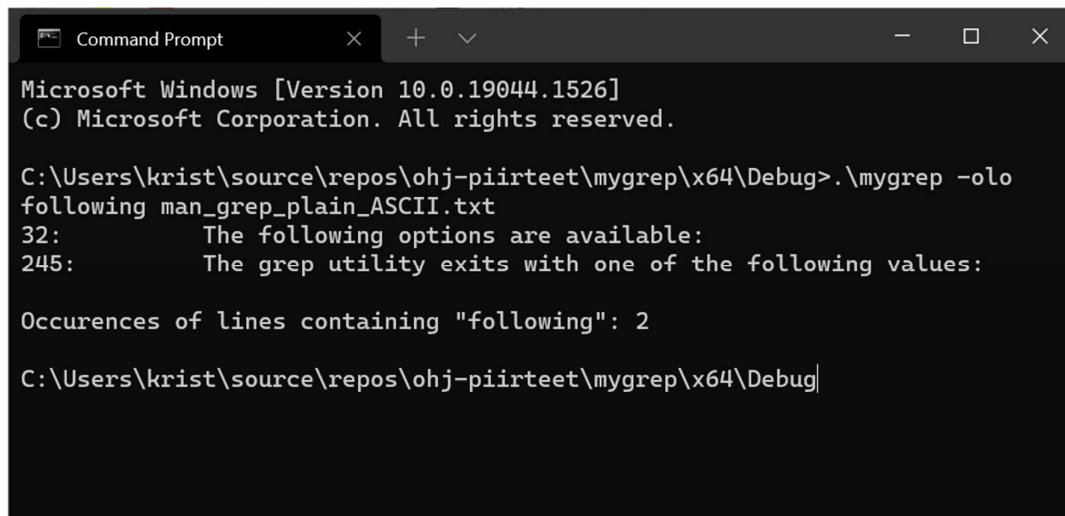
C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
.\mygrep -oo following man_grep_plain_ASCII.txt

Occurences of lines containing "following": 2

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 3. Esimerkkiajo *-oo* -komennolla.

Jos yhdistää komennot eli `-o/o`, tulostetaan rivien numerot ja alhaalla kerrotaan miltä riviltä hakusana löytyi.



```

Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -olo
following man_grep_plain_ASCII.txt
32:      The following options are available:
245:     The grep utility exits with one of the following values:

Occurrences of lines containing "following": 2

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug|

```

KUVA 4. Esimerkkiajo `-o/o` -komennolla.

3.3.1 Komentojen prosessointi

Tehdään jokaisesta syötteestä *string*. Esimerkiksi `std::string olo = "-olo";`
Voimme uudelleen käyttää aiempaa *while* silmukkaa tässä. Lisään silmukkaan kaksi *if* lausetta ja rivilaskija muuttuja `x:n`:

```

x++;
if (argv[1] == olo) {
    std::cout << lineNumber << ":" << '\t' << temp << '\n';
    y = 1;
}
else if (argv[1] == ol) {
    std::cout << lineNumber << '\n';
}

```

Silmukan ulkopuolella on vielä rivitulostus:

```

if (argv[1] == oo || y == 1) {
    std::cout << "\nOccurrences of lines containing \"" <<
searchItem << "\": " << stringMatch << "\n";
}

```

3.3.2 Ongelmat

Koin vaikeuksia, kuinka tulostan rivin vain kerran enkä joka kerta silmukassa. Koin ongelmia myös *if (string)* -lauseissa. Muuten en kokenut isompia ongelmia.

3.4 Neljäs osuus

Neljäs ja viimeinen osuus. Tässä osuudessa idea sama kuin edellisessä, mutta enemmän komentoja:

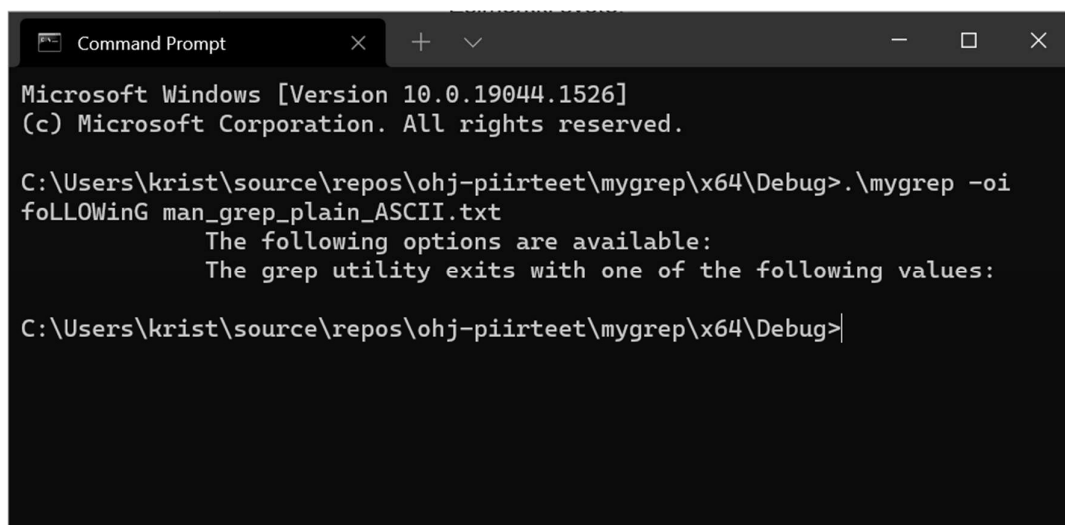
Käyttäjä voi syöttää nyt *-oi*, *-or* ja *-olori* komentoja. Voimme uudelleen käyttää aiempaa koodia tässä, mutta sitä pitää muokata paljon.

Esimerkki syöte:

```
.\mygrep -olori following man_grep_plain_ASCII.txt
```

Ensimmäisen sana syötteessä on ohjelman nimi, toinen on lisäkomento, kolmas on haettava sana, neljäs on avattava tiedoston nimi.

Jos komento on *-oi*, niin pieni- ja isokirjaimet ovat saman arvoisia ja tulostetaan ne rivit.



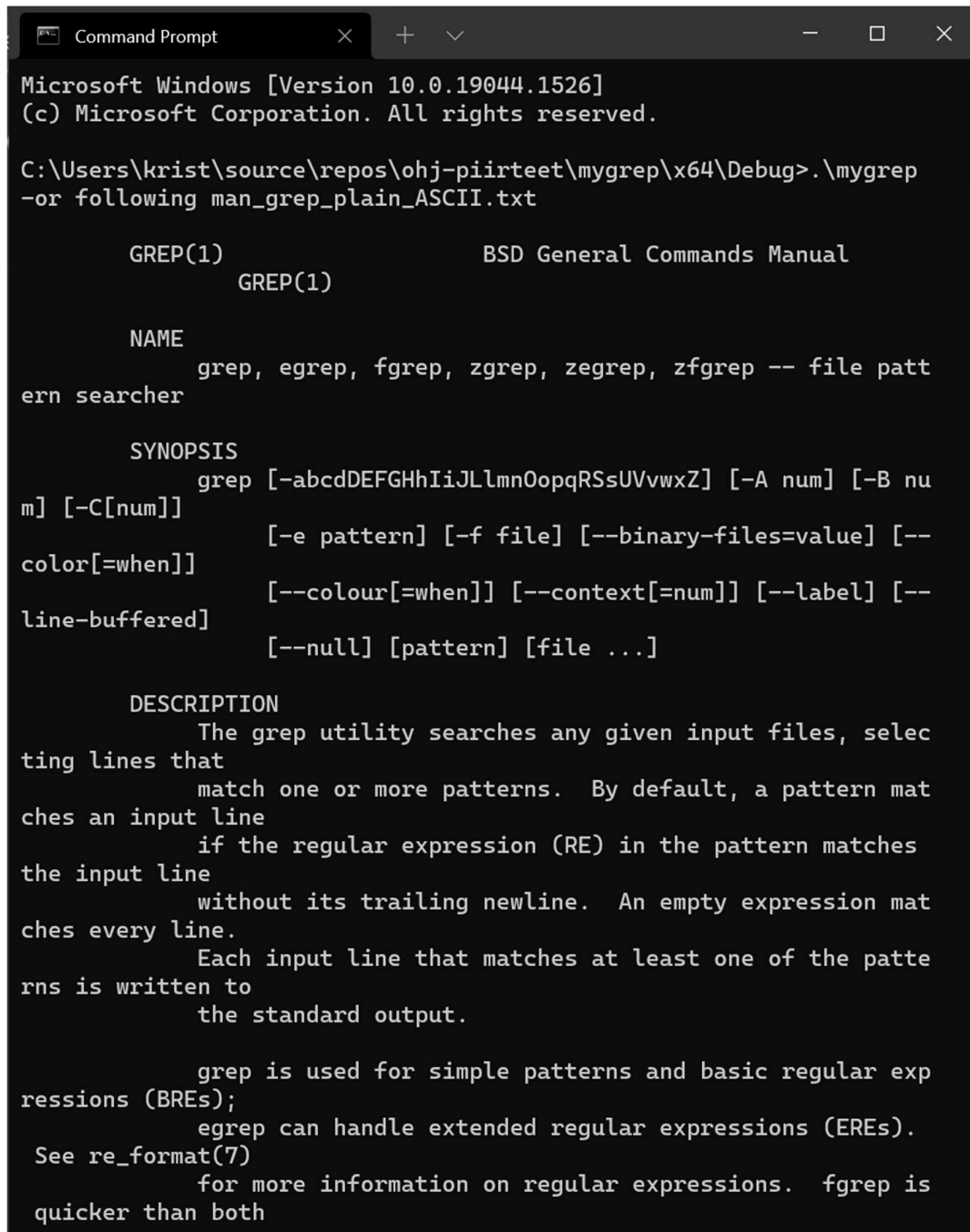
```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -oi
foLLOWinG man_grep_plain_ASCII.txt
    The following options are available:
    The grep utility exits with one of the following values:

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 5. Esimerkkiajo *-oi* -komennolla.

Jos komento on `-or`, etsitään lauseet missä ei ole käyttäjän syötettä.



```

Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
-or following man_grep_plain_ASCII.txt

      GREP(1)                                BSD General Commands Manual
      GREP(1)

      NAME
      grep, egrep, fgrep, zgrep, zegrep, zfgrep -- file pattern
      searcher

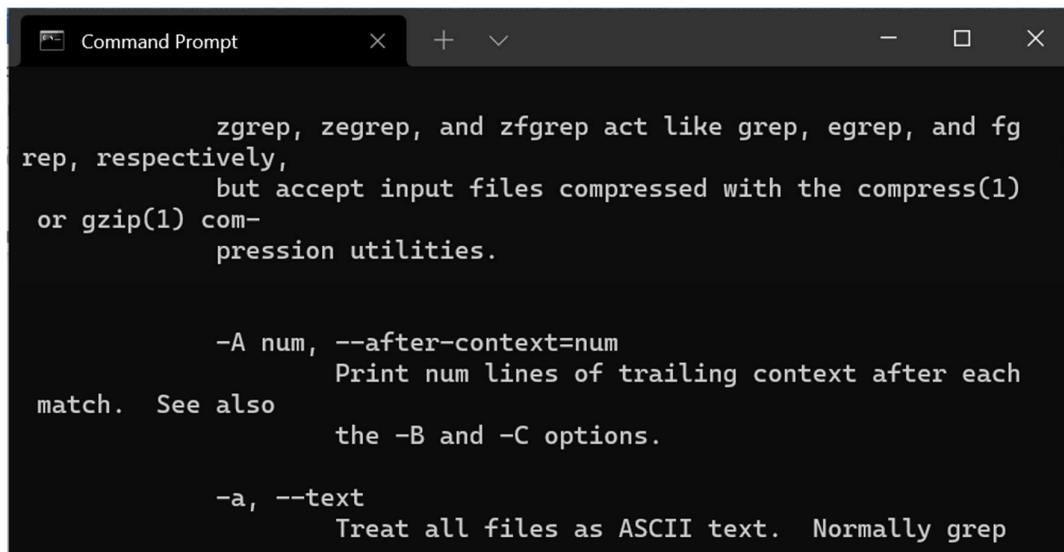
      SYNOPSIS
      grep [-abcdDEFGHhIiJLlmnOopqRSsUVvwXZ] [-A num] [-B num]
      [-C[num]] [-e pattern] [-f file] [--binary-files=value] [--
      color[=when]] [--colour[=when]] [--context[=num]] [--label]
      [--line-buffered] [--null] [pattern] [file ...]

      DESCRIPTION
      The grep utility searches any given input files, selecting
      lines that match one or more patterns. By default, a pattern
      matches an input line if the regular expression (RE) in the
      pattern matches the input line without its trailing newline.
      An empty expression matches every line. Each input line that
      matches at least one of the patterns is written to the standard
      output.

      grep is used for simple patterns and basic regular expressions
      (BREs); egrep can handle extended regular expressions (EREs).
      See re_format(7) for more information on regular expressions.
      fgrep is quicker than both
  
```

KUVA 6. Esimerkkiäjo `-or` -komennolla, osa 1.

...



```

      zgrep, zegrep, and zfgrep act like grep, egrep, and fgrep, respectively,
      but accept input files compressed with the compress(1) or gzip(1) compression utilities.

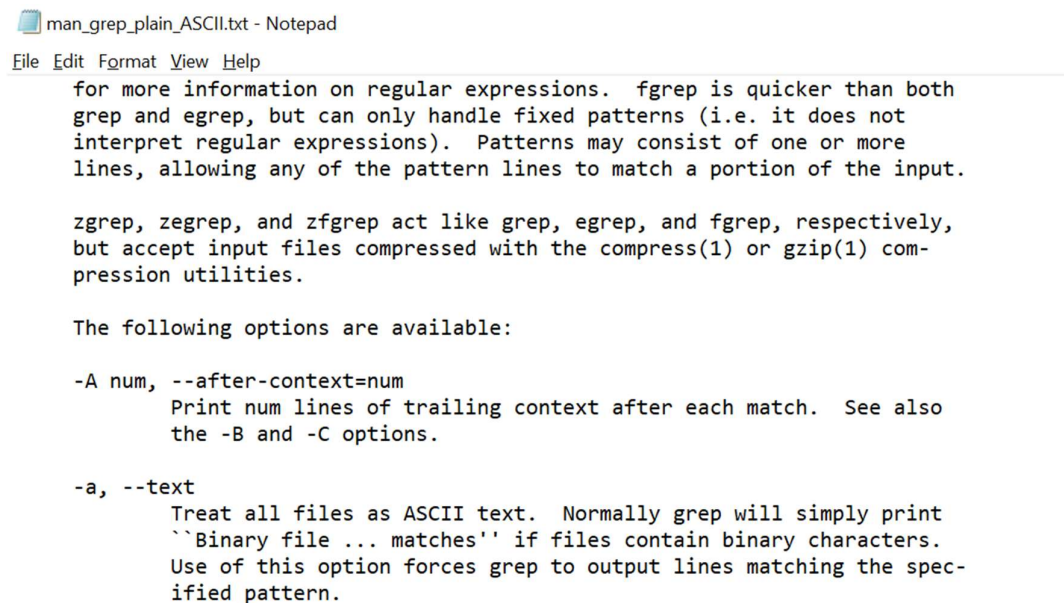
      -A num, --after-context=num
      Print num lines of trailing context after each match. See also the -B and -C options.

      -a, --text
      Treat all files as ASCII text. Normally grep

```

KUVA 7. Esimerkkiajo `-or` -komennolla, osa 2.

Kohdassa `"-A num, --after-context==num"` päällä pitäisi lukea *"The following options are available"*, mutta sitä ei lue `-or` komennon takia.



```

man_grep_plain_ASCII.txt - Notepad
File Edit Format View Help
for more information on regular expressions. fgrep is quicker than both
grep and egrep, but can only handle fixed patterns (i.e. it does not
interpret regular expressions). Patterns may consist of one or more
lines, allowing any of the pattern lines to match a portion of the input.

zgrep, zegrep, and zfgrep act like grep, egrep, and fgrep, respectively,
but accept input files compressed with the compress(1) or gzip(1) compression utilities.

The following options are available:

-A num, --after-context=num
  Print num lines of trailing context after each match. See also
  the -B and -C options.

-a, --text
  Treat all files as ASCII text. Normally grep will simply print
  ``Binary file ... matches'' if files contain binary characters.
  Use of this option forces grep to output lines matching the specified pattern.

```

KUVA 8. `man_grep_plain_ASCII.txt` -tiedosto, jota luimme

3.4.1 Try Catch käyttäminen

Neljännessä osuudessa käytin try catch virheiden prosessointiin. Alla koodi virheen käsittelystä, jos tiedostoa ei voi avata tai tiedostoa ei ole olemassa.

```
try {  
    // koodia  
    throw 100;  
}  
catch (int error) {  
    if (error == 100) {  
        std::cout << "An exception occurred. Exception Nr.-1\n";  
        std::cout << "Could not find size of file \"" << argv[3] << "\"";  
    }  
}
```

3.4.2 Ongelmat

En saanut toimimaan virhetulostusta, jos tiedosto on tyhjä.

```
if (file.peek() == std::ifstream::traits_type::eof()) {  
    std::cout << "File is empty";  
}
```

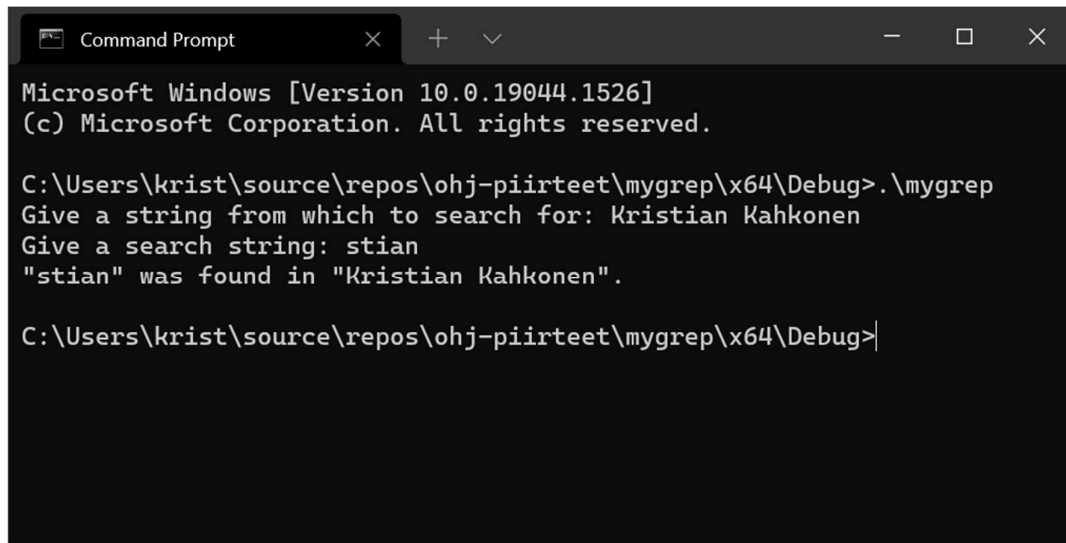
Mutta jos tiedostoa ei ole olemassa, virheenkäsittely toimii oikein eikä ohjelma kaadu.

4 TULOKSET

Ohjelma toimii kuten halutusti, paitsi yksi pieni ongelma. Ohjelma ei tulosta *"File is empty"* jos tiedosto on tyhjä, mutta jos tiedostoa ei ole olemassa virhetulostus toimii oikein.

Alla kappaleissa kuvakaappaukset tuloksista.

4.1 Ensimmäinen osuus

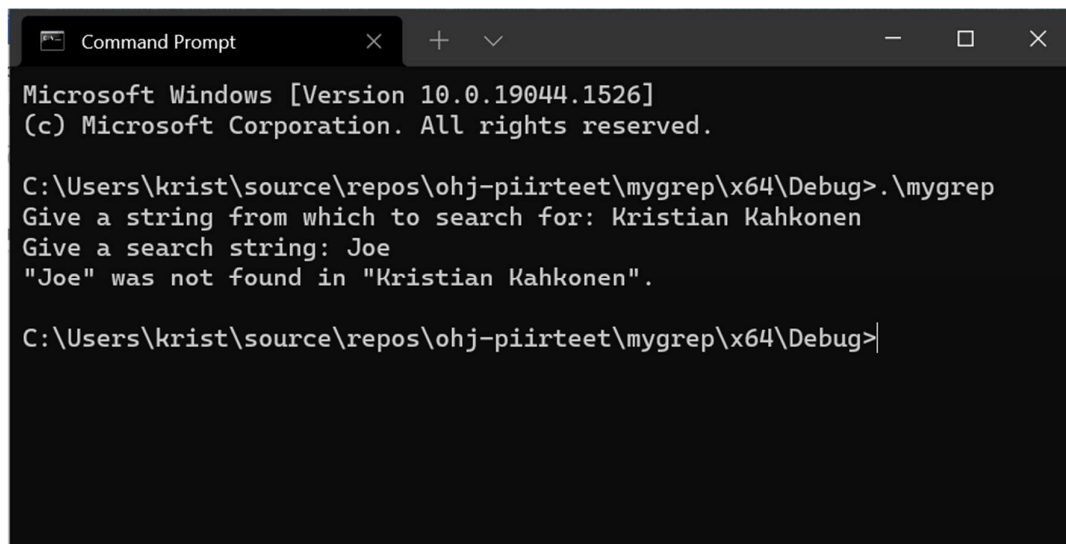


```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
Give a string from which to search for: Kristian Kahkonen
Give a search string: stian
"stian" was found in "Kristian Kahkonen".

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>|
```

KUVA 8. Ensimmäinen osuus demonstraatio jos teksti löytyy



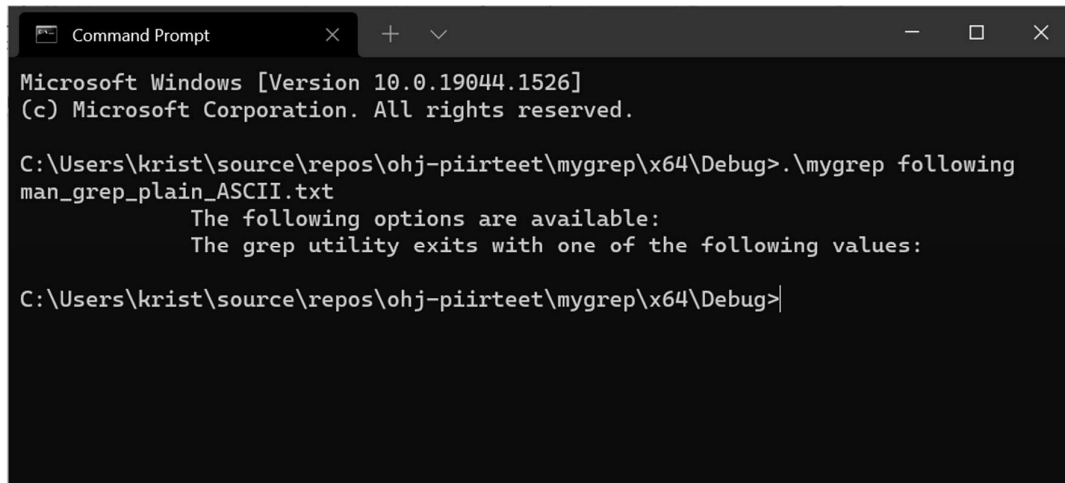
```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
Give a string from which to search for: Kristian Kahkonen
Give a search string: Joe
"Joe" was not found in "Kristian Kahkonen".

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>|
```

KUVA 9. Ensimmäinen osuus demonstraatio, jos teksti ei löydy

4.2 Toinen osuus

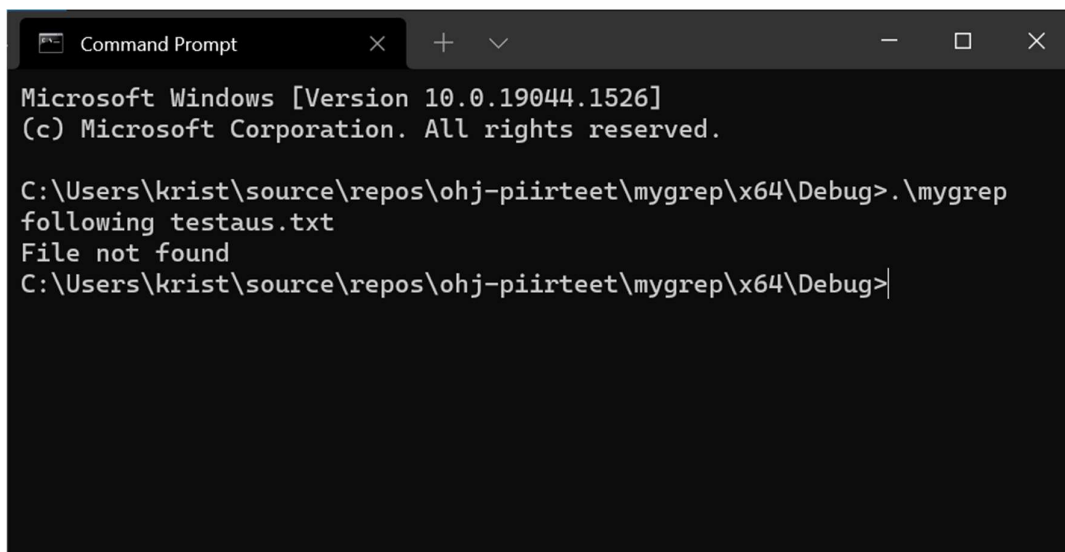


```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep following
man_grep_plain_ASCII.txt
    The following options are available:
    The grep utility exits with one of the following values:

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 10. Toinen osuus demonstraatio, etsitään sanaa "following" tiedostosta

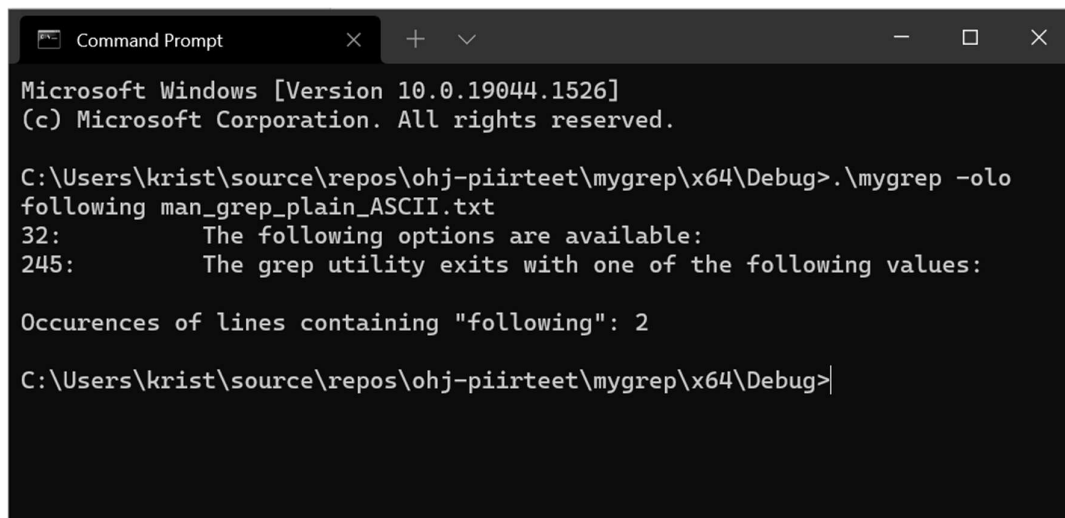


```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
following testaus.txt
File not found
C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 11. Toinen osuus demonstraatio, jos tiedostoa ei ole olemassa tai tiedostoa ei voi lukea.

4.3 Kolmas osuus

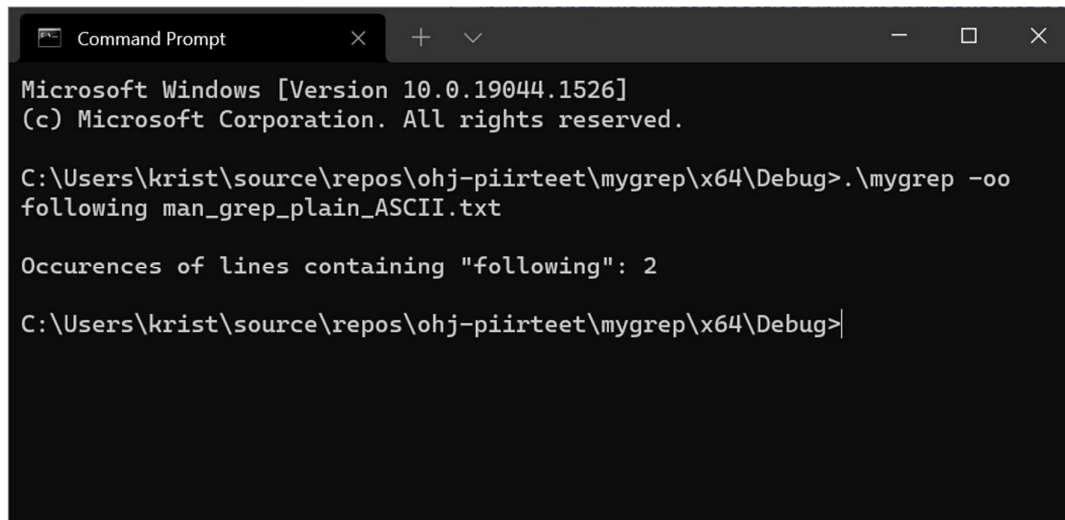


```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -olo
following man_grep_plain_ASCII.txt
32:      The following options are available:
245:     The grep utility exits with one of the following values:

Occurrences of lines containing "following": 2

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 12. Kolmas osuus demonstraatio komennolla `-olo`

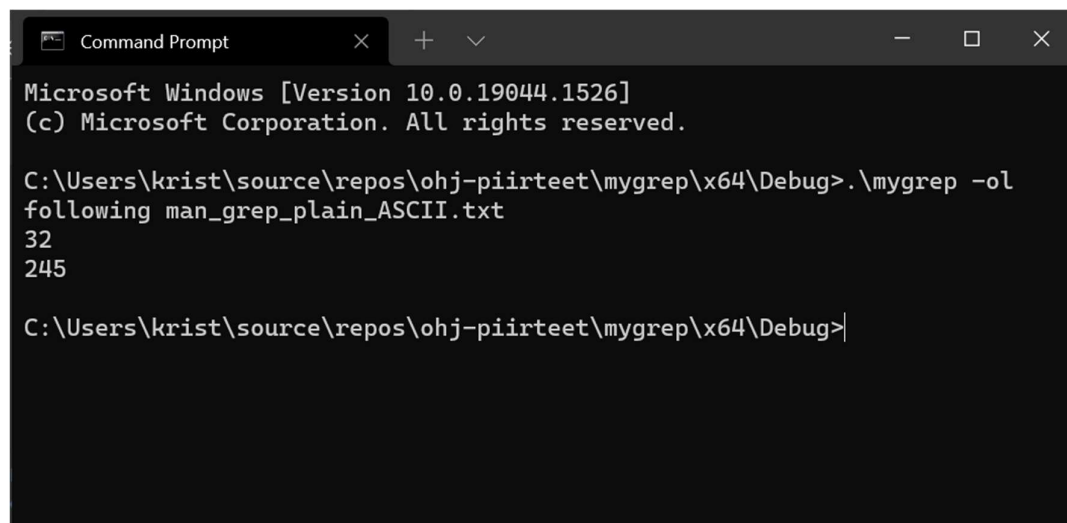
```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -oo
following man_grep_plain_ASCII.txt

Occurrences of lines containing "following": 2

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

KUVA 13. Kolmas osuus demonstraatio komennolla `-oo`

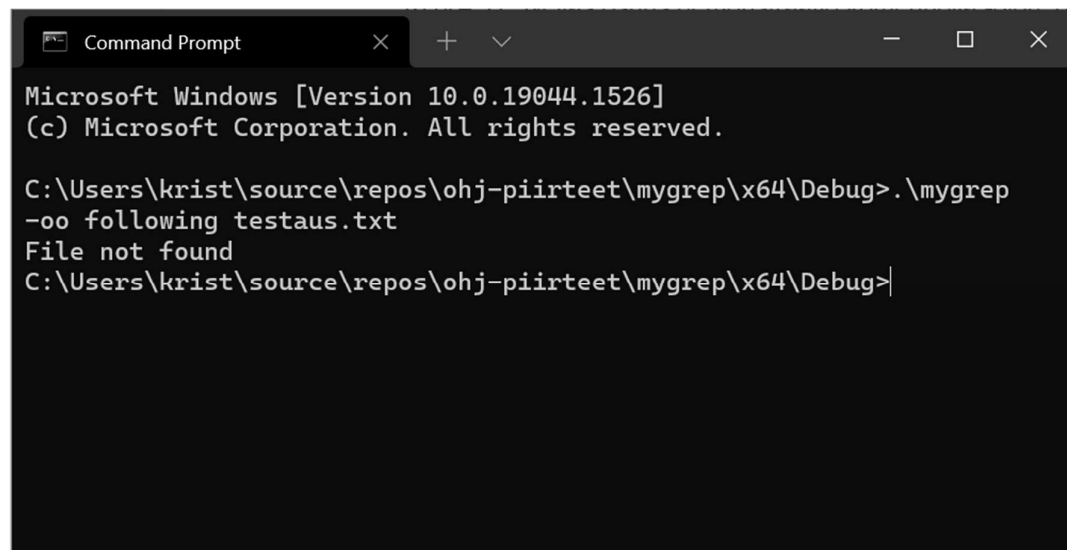


```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -ol
following man_grep_plain_ASCII.txt
32
245

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>|
```

KUVA 14. Kolmas osuus demonstraatio komennolla -o/



```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
-oo following testaus.txt
File not found
C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>|
```

KUVA 15. Kolmas osuus demonstraatio jos tiedostoa ei ole olemassa tai sitä ei voi lukea

4.4 Neljäs osuus

```

Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -or following man_grep_plain_ASCII.txt

      GREP(1)                  BSD General Commands Manual                  GREP(1)

NAME
    grep, egrep, fgrep, zgrep, zegrep, zfgrep -- file pattern searcher

SYNOPSIS
    grep [-abcdDEFGHhIiJLlmnOopqRSsUVvwXZ] [-A num] [-B num] [-C[num]]
        [-e pattern] [-f file] [--binary-files=value] [--color[=when]]
        [--colour[=when]] [--context[=num]] [--label] [--line-buffered]

```

KUVA 16. Neljäs osuus demonstraatio komennolla *-or*

```

Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -oi
following man_grep_plain_ASCII.txt
    The following options are available:
    The grep utility exits with one of the following values:

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>|

```

KUVA 17. Neljäs osuus demonstraatio komennolla *-oi*

```

Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

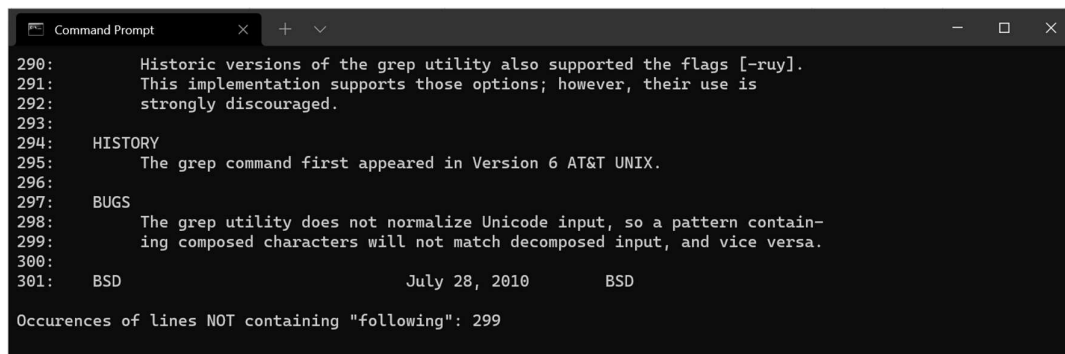
C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep -olori following man_grep_plain_ASCII.txt

1:
2:      GREP(1)                  BSD General Commands Manual                  GREP(1)
3:
4:      NAME
5:          grep, egrep, fgrep, zgrep, zegrep, zfgrep -- file pattern searcher
6:
7:      SYNOPSIS
8:          grep [-abcdDEFGHhIiJLlmnOopqRSsUVvwXZ] [-A num] [-B num] [-C[num]]
9:              [-e pattern] [-f file] [--binary-files=value] [--color[=when]]
10:             [--colour[=when]] [--context[=num]] [--label] [--line-buffered]

```

KUVA 18. Neljäs osuus demonstraatio komennolla *-olori*, osa 1

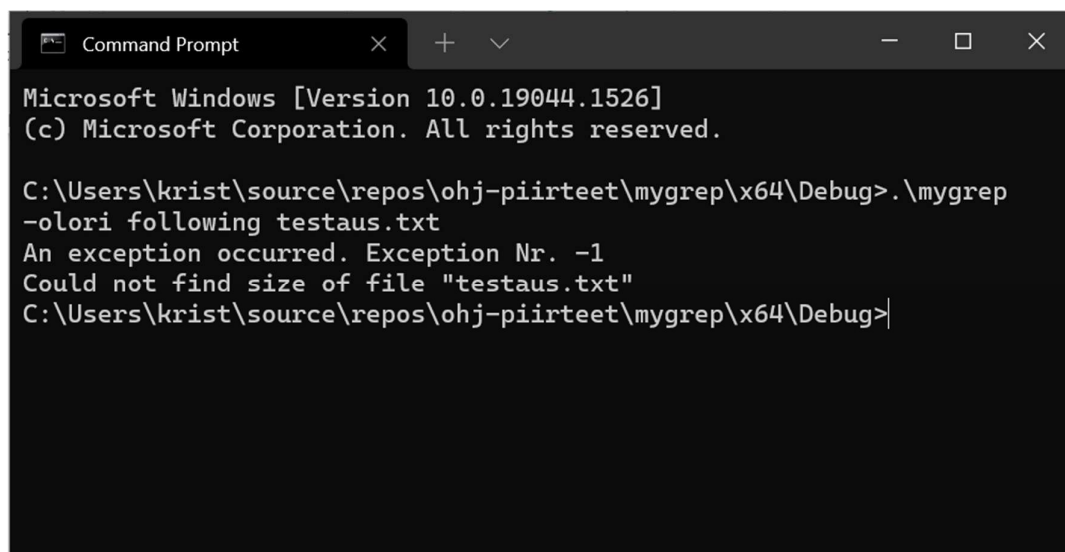
...



```
Command Prompt
290:      Historic versions of the grep utility also supported the flags [-ruy].
291:      This implementation supports those options; however, their use is
292:      strongly discouraged.
293:
294:      HISTORY
295:      The grep command first appeared in Version 6 AT&T UNIX.
296:
297:      BUGS
298:      The grep utility does not normalize Unicode input, so a pattern contain-
299:      ing composed characters will not match decomposed input, and vice versa.
300:
301:      BSD                                July 28, 2010                                BSD

Occurrences of lines NOT containing "following": 299
```

KUVA 19. Neljäs osuus demonstraatio komennolla *-olori*, osa 2



```
Command Prompt
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>.\mygrep
-olori following testaus.txt
An exception occurred. Exception Nr. -1
Could not find size of file "testaus.txt"
C:\Users\krist\source\repos\ohj-piirteet\mygrep\x64\Debug>
```

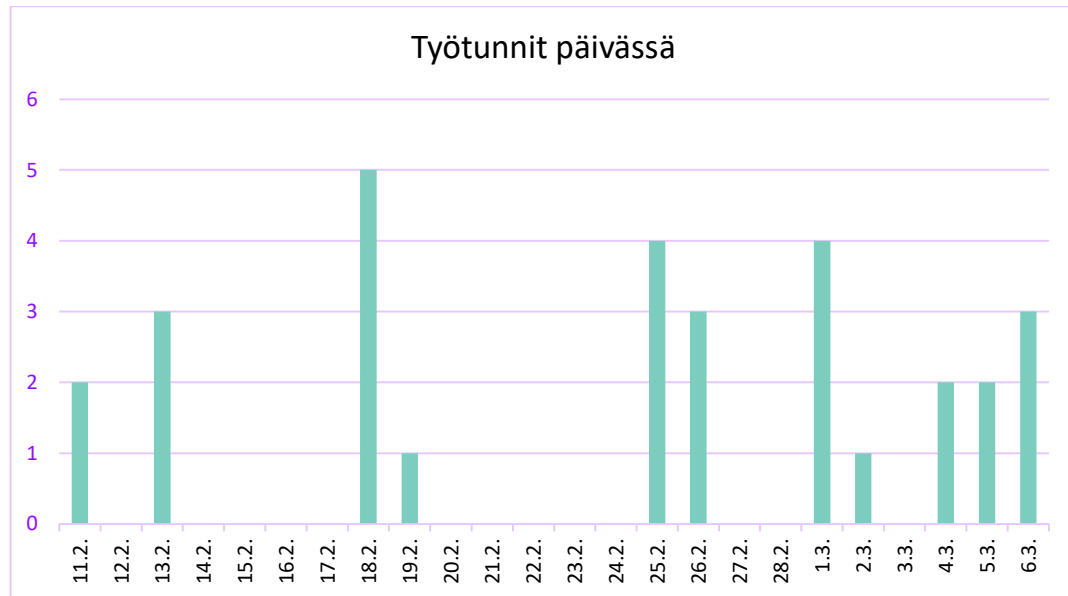
KUVA 20. Neljäs osuus demonstraatio jos tiedostoa ei ole olemassa tai voi lukea.

5 POHDINTA

Projekti kokonaisuudessaan sujui hyvin. Olisin voinut tehdä jostain osista omia aliohjelmia ja tiivistää koodia. En keksinyt mitään käyttöä osoittimille paitsi *stringin* muuntaminen pieniin kirjaimiin. Sain hyödynnettyä kuitenkin muita kurssilla käytyjä asioita, kuten *fstream* ja *try catch*.

Projektiin kului enemmän aikaa kuin odotin, noin 30 tuntia yhteensä.

TAULUKKO 1. Työtunnit päivässä



TAULUKKO 2. Työtunnit tiedot

11.2.	2	Gitin ja Excelin luonti, kommentojen etsiminen
13.2.	3	Taistelu Gitin kanssa ja 1. inkrementin ohjelma mahdollisesti valmis
18.2.	5	Yhden inkrementin työ rakennettu ja seuraava osio aloitettu
19.2.	1	Toisen inkrementin työ valmis
25.2.	4	Kolmannen inkrementin työ valmis
26.2.	3	Neljännän inkrementin aloitus
1.3.	4	Debuggaaminen, viimeistelyä vaille valmis!
2.3.	1	Viimeistely ja kommentoitu koodi
4.3.	2	Raportointi
5.3.	2	Raportointi
6.3.	3	Raportointi ja viimeiset virheenkorjaukset

Tavoittelen projektista arvosanaa 5, sillä tein kaikki osuudet tehtävänannossa.

Linkki GitLabiin: <https://gitlab.tamk.cloud/kristiank/ohj-piirteet>

LÄHTEET

Kartik Ahuja & Avadhut Patade. Command line arguments in C/C++.
<https://www.geeksforgeeks.org/command-line-arguments-in-c-cpp/>

Corob-msft, Taojunshen, DCtheGeek. 08.02.2022. Main function and command-line arguments
<https://docs.microsoft.com/en-us/cpp/cpp/main-function-command-line-args?view=msvc-170>

Alex. 11.6 — Command line arguments
<https://www.learncpp.com/cpp-tutorial/command-line-arguments/>

W3 Schools — C++ Exceptions
https://www.w3schools.com/cpp/cpp_exceptions.asp

LIITTEET

Liite 1. Lähdekoodi

1 (5)

main.cpp:

```

// mygrep.cpp : This file contains the 'main' function. Program execution
// begins and ends there.
// github.com/kristianka

#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include <filesystem>

void toLowercase(std::string& text);

int main(int argc, char* argv[])
{
    std::setlocale(LC_ALL, "fi");
    std::string userInput = "";
    std::string userInput2 = "";
    std::string temp = "";
    std::ifstream file;
    std::string oo = "-oo";
    std::string ol = "-ol";
    std::string olo = "-olo";
    std::string oi = "-oi";
    std::string o_r = "-or";
    std::string olori = "-olori";

    int lineCounter = 0;
    int stringMatch = 0;

    // just an integer, when changed to 1 print lines
    int y = 0;

    // --- 1. INCREMENT PART STARTS ---
    if (argc == 1)
    {
        std::cout << "Give a string from which to search for: ";
        std::getline(std::cin, userInput);
        std::cout << "Give a search string: ";
        std::getline(std::cin, userInput2);

        int found = userInput.find(userInput2);
        // if found is not -1, which means it's found
        if (found != std::string::npos) {
            std::cout << "\"" << userInput2 << "\"" << " was found in " <<
            "\"" << userInput << "\"." << std::endl;
        }
        else {
            std::cout << "\"" << userInput2 << "\"" << " was not found in "
            << "\"" << userInput << "\"." << std::endl;
        }
    }
    // --- 1. INCREMENT PART ENDS ---

```

2 (5)

```

// --- 2. INCREMENT PART STARTS ---
else if (argc == 3) {

    std::string searchItem = argv[1];
    file.open(argv[2]);

    // if file has opened successfully
    if (file)
    {
        // let's read the file
        while (std::getline(file, temp)) {

            //If the string is found in the other string, find() will
            // return -1. string::npos means -1, which means no matches.
            // so if temp has searchItem, code will be executed and
            print temp line
            if (temp.find(searchItem) != std::string::npos) {
                std::cout << '\t' << temp << '\n';
            }

        }
        file.close();
    }
    else {
        std::cout << "File not found";
    }
}
// --- 2. INCREMENT PART ENDS ---

// --- 3. INCREMENT PART STARTS ---

else if (argc == 4 && argv[1] != oi && argv[1] != o_r && argv[1] != ol-
ori) {

    std::string searchItem = argv[2];
    file.open(argv[3]);

    // if file has opened successfully
    if (file)
    {
        // let's read the file
        while (std::getline(file, temp)) {

            // line counter ++
            lineCounter++;

            //If the string is found in the other string, find() will
            // return -1. string::npos means -1, which means no matches.
            // so if temp has searchItem, code will be executed and
            print temp line
            if (temp.find(searchItem) != std::string::npos) {
                stringMatch++;
                if (argv[1] == olo) {
                    std::cout << lineCounter << ":" << '\t' << temp <<
'\n';

                    // let's change y = 1, so occurrences will print only
once.

                    y = 1;
                }
                else if (argv[1] == ol) {
                    std::cout << lineCounter << '\n';

```

```

    }

    }

    if (argv[1] == oo || y == 1) {
        std::cout << "\nOccurrences of lines containing \"" <<
searchItem << "\": " << stringMatch << "\n";
    }
    file.close();
}
else {
    std::cout << "File not found";
}
}
// --- 3. INCREMENT PART ENDS ---

// --- 4. INCREMENT PART STARTS ---
else if (argc == 4 && !(argv[1] == oo || argv[1] == ol || argv[1] ==
olo)) {

    std::string searchItem = argv[2];
    file.open(argv[3]);
    std::string str = searchItem;

    if (argv[1] == oi)
    {
        toLowercase(str);
        // if file has opened successfully
        if (file)
        {
            // let's read the file
            while (std::getline(file, temp)) {

                std::string undercaseTemp = temp;
                toLowercase(undercaseTemp);

                //If the string is found in the other string, find()
will
                // return -1. string::npos means -1, which means no
matches.
                // so if undercaseTemp has searchItem, code will be exe-
cuted and print temp line
                if (undercaseTemp.find(str) != std::string::npos) {
                    std::cout << '\t' << temp << '\n';
                }
            }
            file.close();
        }
        else {
            std::cout << "File not found";
        }
    }

    if (argv[1] == o_r)
    {
        // if file has opened successfully
        if (file)
        {
            // let's read the file
            while (std::getline(file, temp)) {

```

4 (5)

```

//If the string is found in the other string, find() will
// return -1. string::npos means -1, which means no
matches.
// so if temp has searchItem, code will be executed and
print temp line
    if (temp.find(searchItem) == std::string::npos) {
        std::cout << '\t' << temp << '\n';
    }
}
file.close();
}
else {
    std::cout << "File not found";
}
}

if (argv[1] == olori)
{
    try {
        if (file)
        {
            toLowercase(str);
            // let's read the file
            while (std::getline(file, temp)) {
                std::string undercaseTemp = temp;
                toLowercase(undercaseTemp);

                lineCounter++;

                //If the string is found in the other string, find()
                // return -1. string::npos means -1, which means no
                // so if undercaseTemp has searchItem, code will be
                // executed and print temp line
                if (undercaseTemp.find(str) != std::string::npos) {
                    stringMatch++;
                }

                // print lines NOT containing string
                if (undercaseTemp.find(str) == std::string::npos) {
                    std::cout << lineCounter << ":" << '\t' << temp
<< '\n';

                    // let's change y = 1, so occurrences will print
                    // only once.
                    y = 1;
                }
            }
            // print this only once
            if (y == 1) {
                std::cout << "\nOccurrences of lines NOT containing
\"\" << searchItem << "\": " << lineCounter - stringMatch << "\n";
            }
            file.close();
        }
    }
    else {
        try {
            // try to see file's size. if file size is 0 program
            // will crash, but we counter this by using try catch
            std::uintmax_t size = std::filesystem-
            tem::file_size(argv[3]);

```

5 (5)

```

        catch (const std::filesystem::filesystem_error error) {
            throw 100;
        }
    }
    catch (int error) {
        if (error == 100)
        {
            std::cout << "An exception occurred. Exception Nr. -1
\n";
            std::cout << "Could not find size of file \"" << argv[3]
<< "\"";
        }
    }
    }
    // if first line of file is same as end of the file
    if (file.peek() == std::ifstream::traits_type::eof()) {
        std::cout << "File is empty";
    }
    }
    // --- 4. INCREMENT PART ENDS ---

    return 0;
}

void toLowercase(std::string& text) {
    // convert string to lowercase
    std::transform(text.begin(), text.end(), text.begin(), ::tolower);
}

-----

```