



Projekti Mygrep

Juho Tiainen

Raportti
Maaliskuu 2023

Tietotekniikan tutkinto-ohjelma
22TIETOB

juho.tiainen@tuni.fi
045 1711995

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma
22TIETOB

TIAINEN JUHO
Projekti mygrep

Raportti 15 sivua, joista liitteitä 5 sivua.
Maaliskuu 2023

Raportista on tarkoitus saada selville C++ -kielellä tehty Mygrep-ohjelman toiminta ja toiminnot. Toiminnot perustuvat Linux:sta löytyvään grep-työkaluun, jolla voi etsiä pyytämäsi "merkkijonon" rivit ja määrän kuinka useasti se tiedostossa esiintyy. Mygrep-ohjelman on tarkoitus implementoida edellä mainittuja grep-työkaluja ja sisältää sen toimintoja.

Asiasanat: Mygrep, tiivistelmä

Sisällys

1. JOHDANTO.....	5
2. INKREMENTIT	6
1. Ensimmäinen Inkrementti	6
2. Toinen Inkrementti.....	7
3. Kolmas Inkrementti	8
4. Aikataulu.....	9
5. Oppiminen	10
3. LIITTEET.....	11

1. JOHDANTO

Mygrep -ohjelma on tehty inkrementteinä, joista jokainen vastaa tehtävänannon pisteystystä. Tekemäni ohjelma vastaa ja on toteutettu 3. inkrementin kriteerien mukaan eli pisteityksen mukaan 4 pisteen arvoinen. Inkrementit ovat tehty järjestyksessä ja toimii 3. saakka.

-Ohjelma on tehty C++-kielellä mukautumaan Linuxin grep-työkalua ja sen ominaisuuksia. Sen tarkoituksena on hakea mm. merkkijonon kohdan teksistä sekä rivin josta haettu merkkijono löytyy.

Ongelma kohdissa ohjelma ilmoittaa mikäli tiedostoa ei löydy tai syöttö on viallinen. Ohjelma ilmoittaa myös jos etsimääsi merkkijonoa ei löydy. Ohjelmassa voit valita haluatko etsiä vain merkkijonon esiintymismäärän tai vain rivit tai molemmat.

2. INKREMENTIT

1. Ensimmäinen Inkrementti

Ensimmäisen inkrementin toiminta perustuu käyttäjän syötettyihin stringeihin. Käyttäjä syöttää ensin sanan/lauseen josta hän haluaa etsiä tietyn ”merkkijonon”. Jonka jälkeen hän syöttää merkkijonon jota hän etsii edellisestä syötöstä. Ohjelma tulostaa löytyneen merkkijonon position lauseesta (kuva 2.1.0).

Ohjelma ilmoittaa myös mikäli merkkijonoa ei lauseesta löydy (kuva 2.1.1)

Ohjelma on toteutettu koodissa käyttäen argc-toimintoa, joka lukee argumenttien määrän syötteestä. Jos syötteessä on vain yksi komentoriviargumentti, tässä tapauksessa ”mygrep.exe” nimi, ohjelma pääättelee että käytetään ensimmäisen inkrementin hakutoimintoa (kuva 2.1.2).

```
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe
Give a string to search from: Erkki Hietalahti
Give a string to search for : lahti

"lahti" found in "Erkki Hietalahti" in position 11
```

kuva 2.1.0

```
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe
Give a string to search from: Erkki Hietalahti
Give a string to search for : fr

"fr" NOT found in "Erkki Hietalahti "
```

kuva 2.1.1

```
fstream file1; // Variable to collect data from a file
string string1, string2;

string line;

if (argc == 1) { // 1. increment - Search if string is in another string
    cout << "Give a string to search from: ";
    getline(cin >> ws, string1); //User input for string ( sentence or a word ) from where to search

    cout << "Give a string to search for : ";

    getline(cin >> ws, string2); // User input ( a part of a word or a word) for search

    size_t found = string1.find(string2);

    if (found != string::npos) { // Find search string position
        cout << endl << "\"" << string2 << "\" found in \"" << string1 << "\" in position " << found << endl;
    }

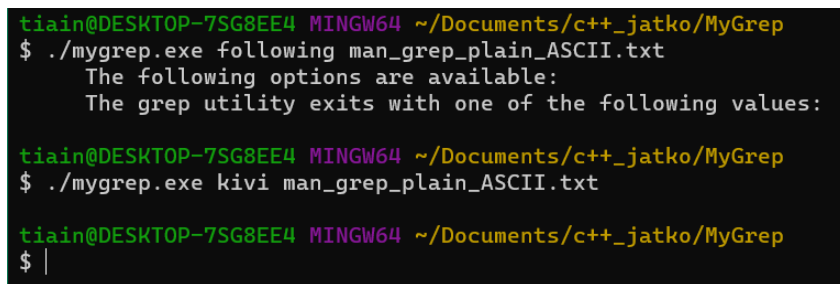
    else { //Error message if string is not found
        cout << endl << "\"" << string2 << "\" NOT found in \"" << string1 << "\"<< endl;
    }
}
```

kuva 2.1.2

2. Toinen Inkrementti

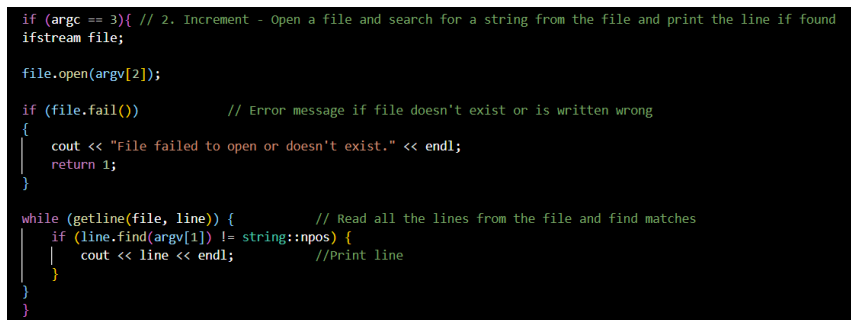
Toisessa inkrementissä toimii ensimmäisen inkrementin toiminnallisuudet ja sen toimintaperiaatteena toimii kolmen komentoriviargumentin käyttäminen. Ensimmäisen argumentti on ohjelmistotiedoston nimi, toinen on hakusana ja kolmas on ASCII-muotoinen tiedosto.

Komento lukee syötetystä tiedostosta, mikäli se on olemassa, sen sisältämät rivit. Jos tiedostosta löytyy hakemasi merkkijono, tulostaa ohjelma kyseiset rivit ulostulossa. Esimerkissä tiedostonimi on "mygrep.exe", hakusana on "following" ja tiedosto on (Linux grep:in käyttöohje) "man_grep_plain_ASCII.txt". Mikäli ohjelma ei löydä haettavaa merkkijonoa, ohjelma ei tulosta mitään (kuva 2.2.0), koodi (kuva 2.2.1.).



```
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe following man_grep_plain_ASCII.txt
The following options are available:
The grep utility exits with one of the following values:
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe kivi man_grep_plain_ASCII.txt
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ |
```

kuva 2.2.0



```
if (argc == 3){ // 2. Increment - Open a file and search for a string from the file and print the line if found
ifstream file;

file.open(argv[2]);

if (file.fail()) // Error message if file doesn't exist or is written wrong
{
    cout << "File failed to open or doesn't exist." << endl;
    return 1;
}

while (getline(file, line)) { // Read all the lines from the file and find matches
    if (line.find(argv[1]) != string::npos) {
        cout << line << endl; //Print line
    }
}
}
```

kuva 2.2.1

3. Kolmas Inkrementti

Kolmannessa inkrementissä toimii ensimmäisen ja toisen inkrementin toiminnallisuudet ja sen toimintaperiaatteena toimii neljän komentoriviargumentin käyttäminen. Neljäntenä argumenttina on Linux Grep:in pohjautuvat "-olo", "-ol" ja "-oo" komennot -o = option, l = line number, o = occurrences ja lo (both line number and occurrences) . Argumentit menevät järjestyksessä "tiedostonimi", "-olo", "haettava sana", "ASCII-tiedosto" (kuva 2.3.0) ja koodi (kuva 2.3.1, 2.3.2, 2.3.3).

jos tiedoston nimi tai komento on kirjoitettu väärin tai se puuttuu, ohjelma tulostaa virhetekstin (kuva 2.3.4)

```
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe -oo following man_grep_plain_ASCII.txt
The following options are available:
The grep utility exits with one of the following values:

Occurrences of line containing "following": 2
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe -ol following man_grep_plain_ASCII.txt
31: The following options are available:
244: The grep utility exits with one of the following values:

tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe -olo following man_grep_plain_ASCII.txt
31: The following options are available:
244: The grep utility exits with one of the following values:

Occurrences of line containing "following": 2
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$
```

kuva 2.3.0

```
if (argc == 4) { // 3. Increment - Add commands to reveal line number, occurrences or both of strings that are searched.
    ifstream file;
    int line_counter;
    if (strcmp(argv[1], "-olo") == 0) // Add -olo command to reveal line number and occurrences of strings that are searched.
    {
        file.open(argv[3]);

        int n_count = 0;

        if (file.fail()) // Error message if file doesn't exist
        {
            cout << "File failed to open or doesn't exist." << endl;
            return 1;
        }

        while (getline(file, line)) { // count line numbers
            line_counter++;
            if (line.find(argv[2]) != string::npos) {
                cout << line_counter << ":\t" << line << endl;
                n_count++;
            }
        }

        // Show how many occurrences there are in the file with specific string
        cout << endl << "Occurrences of line containing \" << argv[2] << "\": " << n_count;
    }
}
```

kuva 2.3.1


```

else if (strcmp(argv[1], "-ol") == 0) // Add -ol command to reveal line number of strings that is searched.
{
    file.open(argv[3]);

    int n_count = 0;

    if (file.fail()) // Error message
    {
        cout << "File failed to open or doesn't exist." << endl;
        return 1;
    }

    while (getline(file, line)) {
        line_counter++; // Count line numbers
        if (line.find(argv[2]) != string::npos) {
            cout << line_counter << ":\\t" << line << endl;
            n_count++;
        }
    }
}

```

kuva 2.3.2

```

}
else if (strcmp(argv[1], "-oo") == 0) // Add -oo command to reveal occurrences of strings that are searched.
{
    file.open(argv[3]);

    int n_count = 0;

    if (file.fail()) // Error message if file doesn't exist
    {
        cout << "File failed to open." << endl;
        return 1;
    }

    while (getline(file, line)) {
        line_counter++; // Count line numbers
        if (line.find(argv[2]) != string::npos) {
            cout << line << endl;
            n_count++;
        }
    }

    // Show how many occurrences there are in the file with specific string
    cout << endl << "Occurrences of line containing \" " << argv[2] << "\": " << n_count;
}

else { // Error message if the command is written wrong or isn't any of the options
    cout << "Error. Command not found!";
}
}

```

kuva 2.3.3

```

tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe -olo following vaaratiedostoascii.txt
File failed to open or doesn't exist.

tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$ ./mygrep.exe -ollkio following man_grep_plain_ASCII.txt
Error. Command not found!
tiain@DESKTOP-7SG8EE4 MINGW64 ~/Documents/c++_jatko/MyGrep
$

```

kuva 2.3.4

4. Aikataulu

2.3.2023	klo 15–22	Koodin rakentamista
3.3.2023	klo 16-20	Koodin tarkastus/korjaus
4.3.2023	klo 18-20	Hiomista ja testit
5.3.2023	klo 14-18	Raportin tekeminen
yhteensä	17 h	

juho.tiainen@tuni.fi

045 1711995

5. Oppiminen

Projektia tehdessä opin käyttämään päätettä ohjelman testaukseen sekä argc ja argv komennot tulivat uutena.

Projekti on tehty 4p arvoisena, koen että 4p on täten järkevä arviointi projektille.

3. LIITTEET

mygrep.cpp:

```
#include <iostream>
```

```
#include <string>
```

```
#include<string.h>
```

```
#include<fstream> //Library to collect data from another file
```

```
using namespace std;
```

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

```
{
```

```
    fstream file1;    // Variable to collect data from a file
```

```
    string string1, string2;
```

```
    string line;
```

```
    if (argc == 1) {    // 1. increment - Search if string is in another string
```

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

```
        getline(cin >> ws, string1); //User input for string ( sentence or a word ) from where to search
```

```
        cout << "Give a string to search for : ";
```

```
        getline(cin >> ws, string2); // User input ( a part of a word or a word) for search
```

```
        size_t found = string1.find(string2);
```

```

if (found != string::npos) {    // Find search string position
    cout << endl << "\"" << string2 << "\" found in \"" << string1 << "\" in position " << found
<< endl;
}

```

```

else {    //Error message if string is not found
    cout << endl << "\"" << string2 << "\" NOT found in \"" << string1 << "\"<< endl;
}
}

```

if (argc == 3){ // 2. Increment - Open a file and search for a string from the file and print the line if found

```

ifstream file;

```

```

file.open(argv[2]);

```

```

if (file.fail())    // Error message if file doesn't exist or is written wrong
{
    cout << "File failed to open or doesn't exist." << endl;
    return 1;
}

```

```

while (getline(file, line)) {    // Read all the lines from the file and find matches
    if (line.find(argv[1]) != string::npos) {
        cout << line << endl;    //Print line
    }
}
}

```

if (argc == 4){ // 3. Increment - Add commands to reveal line number, occurrences or both of strings that are searched.

```

ifstream file;

```

```

int line_counter;

```

```
if(strcmp(argv[1], "-olo") == 0)    // Add -olo command to reveal line number and
occurrences of strings that are searched.
```

```
{
    file.open(argv[3]);

    int n_count = 0;

    if (file.fail())                // Error message if file doesn't exist
    {
        cout << "File failed to open or doesn't exist." << endl;
        return 1;
    }

    while (getline(file, line)) {
        line_counter++;            // Count line numbers
        if (line.find(argv[2]) != string::npos) {
            cout << line_counter << ":\t" << line << endl;
            n_count++;
        }
    }

} // Show how many occurrences there are in the file with specific string
    cout << endl << "Occurrences of line containing \"" << argv[2] << "\": " << n_count;

}
```

```
else if (strcmp(argv[1], "-ol") == 0)    // Add -ol command to reveal line number of
strings that is searched.
```

```
{
    file.open(argv[3]);

    int n_count = 0;
```

```

if (file.fail())          // Error message
{
    cout << "File failed to open or doesn't exist." << endl;
    return 1;
}

while (getline(file, line)) {
    line_counter++;        // Count line numbers
    if (line.find(argv[2]) != string::npos) {
        cout << line_counter << ":\t" << line << endl;
        n_count++;
    }
}

}

else if(strcmp(argv[1], "-oo") == 0)    // Add -oo command to reveal occurrences of
strings that are searched.
{
    file.open(argv[3]);

    int n_count = 0;

    if (file.fail())        // Error message if file doesn't exist
    {
        cout << "File failed to open." << endl;
        return 1;
    }

    while (getline(file, line)) {

```

```

        line_counter++;          // Count line numbers
        if (line.find(argv[2]) != string::npos) {
            cout << line << endl;
            n_count++;

        }

    } // Show how many occurrences there are in the file with specific string
    cout << endl << "Occurrences of line containing \"" << argv[2] << "\": " <<
n_count;

    }
    else {          // Error message if the command is written wrong or isn't any of the
options
        cout << "Error. Command not found!";
    }
}

return 0;
}

```