

Käyttöjärjestelmät ja systeemiohjelmointi kesä 2024

Harjoitustyöprojekti

Ella Salo, 000668578

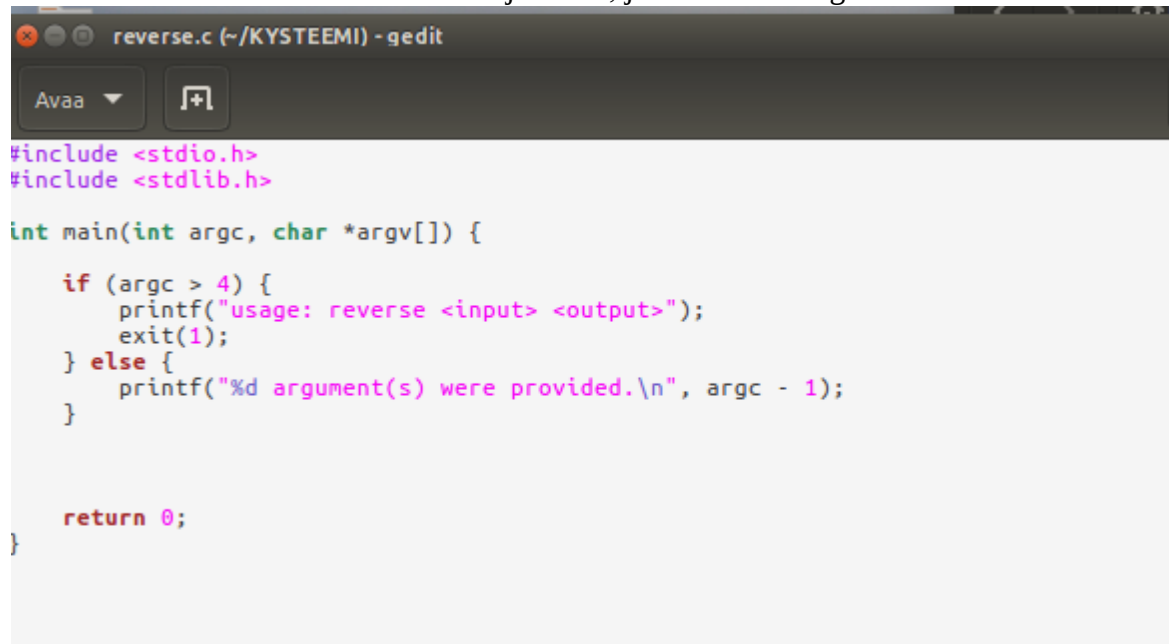
25.7.2024

Kaikki koodit löytyvät osoitteesta: <https://github.com/ella-salo/kysteemi>.

Pahoittelen tiedoston huonolaatuisia kuvakaappauksia. Näytöillä näkyvät laatikot eivät ole vika vaan ominaisuus vanhassa Ubuntussani. Sain loppujen lopuksi ongelman korjattua ja viimeisissä projekteissa ongelmaa ei enää ole. Lisäksi kuvakaappauksissa, jossa ajan koodejani, näkyy isäni käyttäjänimi, koska aloin vahingossa isäni käyttäjällä näitä tekemään enkä jaksanut siirtyä enää omalleni. Vakuutan kuitenkin, että olen itse kaiken tämän tehnyt ja isäni ei olisi osannut näitä koodata.

Project 1: Warmup to C and Unix programming

Aloitetaan luomalla toimiva alku C-ohjelmalle, joka tarkistaa argumenttien määrän:



```
reverse.c (~ /KYTEEMI) - gedit
Avaa ▾ [icon]

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc > 4) {
        printf("usage: reverse <input> <output>");
        exit(1);
    } else {
        printf("%d argument(s) were provided.\n", argc - 1);
    }

    return 0;
}
```

Sitten teemme ohjelman, joka input-tiedoston saadessaan onnistuu lukemaan sen stdoutiin:

```

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc > 4) {
        printf("usage: reverse <input> <output>");
        exit(1);
    } else if (argc == 2) {
        // This is if only inputfile is given

        FILE *inputFile = fopen(argv[1], "r");
        if (inputFile == NULL) {
            perror("Error opening input file");
            exit(1);
        }

        char line[256];
        while (fgets(line, sizeof(line), inputFile)) {
            printf("%s", line);
        }

        fclose(inputFile);
    } else {
        printf("%d argument(s) were provided.\n", argc - 1);
    }

    return 0;
}

```

Joka näyttää toimivan:

```

markus@markus-X550CL:~/KYSTEEI$ ./rev input.txt
hello
i
am
an
engineer

```

Nyt muokataan luku käyttämään linkitettyä listaa niin, että tiedosto voi olla kuinka pitkä tahansa. Lisätään myös jokainen alkio aina listan alkuun niin se saadaan tulostumaan käänteisenä.

Linkitettyä listaa varten luodaan seuraavat:

```

typedef struct Node {
    char line[256];
    struct Node *next;
} Node;

// Makes a new node
Node* createNode(char *line) {
    Node *newNode = (Node*)malloc(sizeof(Node));
    strcpy(newNode->line, line);
    newNode->next = NULL;
    return newNode;
}

// Adds the new node to the beginning of the linked list
void addNodeToBeginning(Node **head, char *line) {
    Node *newNode = createNode(line);
    newNode->next = *head;
    *head = newNode;
}

// Frees memory
void freeMemory(Node *head) {
    Node *temp;
    while (head != NULL) {
        temp = head;
        head = head->next;
        free(temp);
    }
}

```

Asetettiin riville kumminkin staattinen maksimimitta, koska linkitetyn listan tekeminen linkitetyn listan sisälle ei tunnu järkevältä.

```

markus@markus-X550CL:~/KYTEEMI$ gcc reverse.c -o rev
markus@markus-X550CL:~/KYTEEMI$ ./rev input.txt
engineer
an
an
i
hello

```

Näyttää toimivan.

Seuraavaksi tehtävänannosta löytyi hyvä vinkki käyttää getline() komentoa, jotta voidaan lukea minkä tahansa mittainen rivi. Poistetaan se staattinen maksimimitta siis.

```

// For dynamic reading of the inputfile using getline s.t. the line can be any length
Node *head = NULL;
char *line = NULL;
size_t len = 0;
ssize_t read;

while ((read = getline(&line, &len, inputFile)) != -1) {
    addNodeToBeginning(&head, line);
}

free(line);
fclose(inputFile);

```

Node-structia piti vähän muuttaa, rivistä tehdään pointeri:

```

typedef struct Node {
    char *line;
    struct Node *next;
} Node;

```

Tämän jälkeen koodin rakennetta muutettiin niin, että inputFile ja outputFile alustetaan NULL:ksi ja annettujen parametrien mukaisesti määritetään joko annetuksi tiedostoksi tai stdin/stdout. Tämän lisäksi käydään läpi tehtävänannon tarkempia vaatimuksia errorista yms formaliteeteista.

```

int main(int argc, char *argv[]) {
    // If too many arguments
    if (argc > 3) {
        fprintf(stderr, "usage: reverse <input> <output>\n");
        exit(1);
    } else if (argc == 3) { // If the input and output files are the same
        if (strcmp(argv[1], argv[2]) == 0) {
            fprintf(stderr, "Input and output file must differ\n");
        }
    }

    // Initialize files to be NULL
    FILE *inputFile = NULL;
    FILE *outputFile = NULL;

    if (argc > 1) { // If arguments are given, the first one is the input file
        inputFile = fopen(argv[1], "r");
        if (inputFile == NULL) {
            fprintf(stderr, "error: cannot open file '%s'\n", argv[1]);
            exit(1);
        }
    } else { // If no arguments, reads from stdin
        inputFile = stdin;
    }

    if (argc == 3) { // If 3 arguments, the third one is the output file
        outputFile = fopen(argv[2], "w");
        if (outputFile == NULL) {
            fprintf(stderr, "error: cannot open file '%s'\n", argv[2]);
            fclose(inputFile);
            exit(1);
        }
    } else { // Otherwise print to stdout
        outputFile = stdout;
    }

    // Outputting
    Node *current = head;
    while (current != NULL) {
        fprintf(outputFile, "%s", current->line);
        current = current->next;
    }
}

```

Sitten testataan, toimiiko ilman annettuja parametreja:

```

markus@markus-X550CL:~/KYTEEMI$ ./rev
oh
yeah
i
think
this
actually
works
works
actually
this
think
i
yeah
oh
markus@markus-X550CL:~/KYTEEMI$

```

Sitten testataan, toimiiko pelkällä input-tiedostolla:

```

markus@markus-X550CL:~/KYTEEMI$ ./rev input.txt
engineer
an
an
i
hello
markus@markus-X550CL:~/KYTEEMI$

```

Sitten testataan kahdella eri tiedostolla:

```
markus@markus-X550CL:~/KYTEEMI$ touch output.txt
markus@markus-X550CL:~/KYTEEMI$ ./rev input.txt output.txt
markus@markus-X550CL:~/KYTEEMI$ cat output.txt
engineer
an
am
i
hello
markus@markus-X550CL:~/KYTEEMI$
```

Kaikki näyttävät toimivan. Tarkistetaan vielä, että koodi saa kiinni virheet väärästä määrästä parametreja ja siitä, jos input-ja output-tiedostot ovat samat tai jos tiedoston avaaminen ei onnistu.

```
markus@markus-X550CL:~/KYTEEMI$ ./rev input.txt output.txt input.txt
usage: reverse <input> <output>
markus@markus-X550CL:~/KYTEEMI$ ./rev input.txt input.txt
Input and output file must differ
markus@markus-X550CL:~/KYTEEMI$ ./rev inp.txt
error: cannot open file 'inp.txt'
markus@markus-X550CL:~/KYTEEMI$
```

Näyttää toimivan juuri niin kuin pitääkin.

Project 2: Unix Utilities

my-cat.c

Aloitetaan luomalla ohjelma, joka vain lukee yhden annetun tiedoston tehtävänannossa määritellyllä tavalla:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    FILE *fp = fopen(argv[1], "r");
    if (fp == NULL) {
        printf("my-cat: cannot open file\n"); // Error in opening a file
        exit(1);
    }

    char buffer[1024]; // Buffer for the line
    while (fgets(buffer, sizeof(buffer), fp) != NULL) {
        printf("%s", buffer);
    }

    fclose(fp);
    return 0;
}
```

Tämä toimii:

```
markus@markus-X550CL:~/KYSTEEI/Project2$ ./my-cat file.txt
Susikoira Roi oli nuorena innostunut rumpujen soittamisesta, ja soittipa hän ihan bändissäkin. Roin
bändin lupaava nousukiito katkesi kuitenkin Roin päihdeongelmiin ja lopulta kun Roi ei pystynyt
edes omin jaloin kiipeämään esiintymislavalle, bändi päätti laittaa pillit pussiin.

Roi tajusi, että hänen oli ryhdistäydyttävä, ja hän pestautui matruusiksi laivalle. Raikas
meri-ilma selvittäisi hänen päänsä ja hän pääsisi aloittamaan elämänsä puhtaalta pöydältä. Vanhat
rumpunsa Roi myi pois yhtä bongorumpua lukuunottamatta. Sitä olisi mukava soittaa laivan kannella
auringonlaskua katsellessa.

Laivalla kaikki sujuikin aluksi hyvin, ja Roi viihdytti muuta henkilökuntaa jokailtaisilla
rumpusooloillaan. Mutta sitten laivalla alkoi levitä kumma tauti. Yhä useampi laivallaolija alkoi
kärsiä ensin merisairauden kaltaisesta pahoinvoinnista, ja sitten vuorokauden kuluessa vaipua
koomaan. Taudin leviämisen hillitsemiseksi sairastuneet vietiin ruumaan perustetulle
karanteeniosastolle.

Roin harmiksi tauti tarttui myös häneen. Roikin vaipui koomaan ja hänet kannettiin ruumaan. Yöllä
tarkastuskierroksella ollut laivan lääkäri ihmetteli Roin huoneesta kantautuvaa meteliä. Lääkäri
kurkisti ovesta olevasta aukosta ja ihmetyksekseen näki Roin hoippuvan ja kaatuilevan ympäri
huonetta. Paikalle saapunut perämies tiedusteli, mitä ihmettä huoneessa tapahtuu. Lääkäri totesi
jopa hieman huvittuneesti: "Harva ruumassa kompuroi, kuin koomassa rumpu-Roi."
markus@markus-X550CL:~/KYSTEEI/Project2$
```

Sitten laajennetaan se lukemaan loopissa kaikki annetut argumentit ja lukemaan kaikki tiedostot. Jos tiedostoa ei anneta, poistutaan palauttaen 0.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc == 1) { // If no files are given, we just return
        return 0;
    }

    for (int i = 1; i < argc; i++) { // Looping through all files given as arguments
        FILE *fp = fopen(argv[i], "r");
        if (fp == NULL) {
            printf("my-cat: cannot open file\n"); // Error in opening a file
            exit(1);
        }

        char buffer[1024]; // Buffer for the line
        while (fgets(buffer, sizeof(buffer), fp) != NULL) {
            printf("%s", buffer);
        }

        fclose(fp);
    }

    return 0;
}
```

Koodi näyttää toimivan ja tulostaa peräkkäin ensin annetun Susikoira Roi -vitsin ja pätkän Hamletia.

koomaan. Taudin leviämisen hillitsemiseksi sairastuneet vietiin ruumaan perustetulle karanteeniosastolle.

Roin harmiksi tauti tarttui myös häneen. Roikin vaipui koomaan ja hänet kannettiin ruumaan. Yöllä tarkastuskierroksella ollut laivan lääkäri ihmetteli Roin huoneesta kantautuvaa meteliä. Lääkäri kurkisti ovesta olevasta aukosta ja ihmetyksekseen näki Roin hoippuvan ja kaatuilevan ympäri huonetta. Paikalle saapunut perämies tiedusteli, mitä ihmettä huoneessa tapahtuu. Lääkäri totesi jopa hieman huvittuneesti: "Harva ruumassa kompuroi, kuin koomassa rumpu-Roi."

Enter Claudius King of Denmarke, Gertrude the
Queene, Hamlet, Polonius,
Laertes, and his Sister Ophelia,
Lords Attendant.
King.

Though yet of Hamlet our deere Brothers death
The memory be greene: and that it vs befitted
To beare our hearts in greefe, and our whole Kingdome
To be contracted in one brow of woe:
Yet so farre hath Discretion fought with Nature,
That we with wisest sorrow thinke on him,
Together with remembrance of our selues.
Therefore our sometimes Sister, now our Queen,
Th'Imperiall Ioyntresse of this warlike State,
Haue we, as 'twere, with a defeated ioy,
With one Auspicious, and one Dropping eye,
With mirth in Funerall, and with Dirge in Marriage,
In equall Scale weighing Delight and Dole
Taken to Wife; nor haue we heerein barr'd
Your better Wisedomes, which haue freely gone
With this affaire along, for all our Thanks.
Now followes, that you know young Fortinbras,
Holding a weake supposall of our worth;
Or thinking by our late deere Brothers death,
Our State to be disioynt, and out of Frame,
Collegued with the dreame of his Aduantage;
He hath not fayl'd to pester vs with Message,

Ilman argumentteja koodi vain lopettaa ajamisen:

```
markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-cat  
markus@markus-X550CL:~/KYTEEMI/Project2$
```

my-grep.c

Käytetään tämän ominaisuuden tekemisessä string kirjaston funktiota strstr() jonne annettaessa kaksi merkkijonoa, se etsii ensimmäisestä jälkimmäisen substringin ensimmäisen esiintymisindeksin ja palauttaa sen pointerin. Jos pointer on NULL, tiedämme ettei jälkimmäistä stringiä ole ensimmäisessä.

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

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

    if (argc == 1) { // If no arguments are given
        printf("my-grep: searchterm [file ...]\n");
        exit(1);
    }

    char *searchstring = argv[1];

    if (strlen(searchstring) == 0) { // If given string is empty, will not print anything
        exit(0);
    }

    // Needed initializations for the getline()
    char *line = NULL;
    size_t len = 0;

    if (argc == 2) { // If no files are given, read from stdin
        while (getline(&line, &len, stdin) != -1) {
            if (strstr(line, searchstring) != NULL) {
                printf("%s", line);
            }
            free(line);
            return 0; // Natural exit
        }
    }

    // Loop through each file specified in the command line arguments
    for (int i = 2; i < argc; i++) {
        FILE *file = fopen(argv[i], "r");
        if (file == NULL) {
            printf("my-grep: cannot open file\n");
            exit(1);
        }

        while (getline(&line, &len, file) != -1) {
            // strstr return a pointer to the first index of a substring (searchstring) so
            // if that is not null, then that string is on the line and we print it.
            if (strstr(line, searchstring) != NULL) {
                printf("%s", line);
            }
        }

        fclose(file);
    }

    free(line);
    return 0;
}

```

Koodin tekemisen välivaiheet valitettavasti hävisivät, kun tekstinkäsittelyohjelma kaatui varoittamatta. Se näyttää nyt kuitenkin toimivan halutulla tavalla.

```

markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-grep thou file.txt file2.txt
And loose your voyce. What would'st thou beg Laertes,
What would'st thou have Laertes?
From whence, though willingly I came to Denmarke
My thoughts and wishes bend againe towards France,
markus@markus-X550CL:~/KYTEEMI/Project2$

markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-grep rumpu file.txt file2.txt
Susikoiria Roi oli nuorena innostunut rumpujen soittamisesta, ja soittipa hän ihan bändissäkin. Roin
rumpunsa Roi myi pois yhtä bongorumpua lukuunottamatta. Sitä olisi mukava soittaa laivan kannella
rumpusooloillaan. Mutta sitten laivalla alkoi levitä kumma tauti. Yhä useampi laivallaolija alkoi
jopa hieman huvittuneesti: "Harva ruumassa kompuroi, kuin koomassa rumpu-Roi."
markus@markus-X550CL:~/KYTEEMI/Project2$

```

my-zip.c ja my-unzip.c

Tässä tehtävässä luodaan oma zip- ja unzip. my-zip.c muodostaa RLE-menetelmällä pakatun datan ja tulostaa sen stdoutiin. Data tallennetaan niin, että ensin on 4-tavuinen binääriluku ja sitten merkki, jota se edustaa. 4 tavua on 32 bittiä.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc < 2) { // Check that at least one file is given
        printf("my-zip: file1 [file2 ...]\n");
        return 1;
    }

    for (int i = 1; i < argc; i++) { // Go through all given files
        FILE *file = fopen(argv[i], "r");
        if (file == NULL) {
            printf("my-zip: cannot open file\n");
            return 1;
        }

        int count = 0;
        char current, previous;

        if ((current = fgetc(file)) == EOF) { // Puts in the first value for current or finds out
            // that the file is empty and we can leave this one
            break;
        }
        count++;
        previous = current;

        // Going through the file one character at a time and we increase the counter while it's
        // the same as the previous and go back to one if its not

        while ((current = fgetc(file)) != EOF) {
            if (current == previous) {
                count++;
            } else {
                fwrite(&count, 4, 1, stdout); // Writes a 4-byte integer in binary
                printf("%s", &previous); // Writes the ASCII character
                count = 1; // Resetting counter (next will be the first occurrence)
            }
            previous = current;
        }

        fclose(file);
    }

    return 0;
}
```

Näin ollen käymme jokaisen tiedoston kaikki merkit läpi laskien koko ajan kuinka monta samaa peräkkäistä kirjainta on ja kirjoitamme stdoutiin 4-tavuisen kokonaisluvun binäärillä ja sen perään ASCII merkin.

```
markus@markus-X550CL:~/KYSTEEI/Project2$ gcc -o my-zip my-zip.c -Wall -Werror
markus@markus-X550CL:~/KYSTEEI/Project2$ ./my-zip tobezipped.txt
ah
0h
markus@markus-X550CL:~/KYSTEEI/Project2$ ./my-zip tobezipped.txt > zipped.txt
markus@markus-X550CL:~/KYSTEEI/Project2$ cat zipped.txt
ah
0h
markus@markus-X550CL:~/KYSTEEI/Project2$
```

Tästä on tässä vaiheessa vaikea sanoa, toimiiko se täysin, koska binäärilukuja se ei näytä mukavasti lukevan, mutta se nähdään, kun ollaan tehty unzip.

```

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc < 2) { // Check that at least one file is given
        printf("my-zip: file1 [file2 ...]\n");
        return 1;
    }

    for (int i = 1; i < argc; i++) { // Go through all given files
        FILE *file = fopen(argv[i], "r");
        if (file == NULL) {
            printf("my-unzip: cannot open file\n");
            return 1;
        }

        // Going through the file one character at a time
        int count = 0;
        char ch;
        while (fread(&count, sizeof(int), 1, file) == 1) { // Read the number
            // Read the char and make sure that the file is in correct format
            if (fread(&ch, sizeof(char), 1, file) != 1) {
                printf("my-unzip: invalid file format\n");
                return 1;
            } else {
                for (int i = 0; i < count; i++) { // print as many of the chars as indicated
                    printf("%c", ch);
                }
            }
        }

        fclose(file);
    }

    return 0;
}

```

Tämän jälkeen kuitenkin zipattu tiedosto näyttäisi olevan väärässä formaatissa.

```

markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-unzip zipped.txt
aaaaaaaaaamy-unzip: invalid file format
markus@markus-X550CL:~/KYTEEMI/Project2$ gcc -o my-zip my-zip.c

```

Luultavasti kuitenkin merkki täytyy kirjoittaa zipatessa samalla tavalla kuin numerokin. Tehdään muutos koodiin:

```

while ((current = fgetc(file)) != EOF) {
    if (current == previous) {
        count++;
    } else {
        fwrite(&count, 4, 1, stdout); // Writes a 4-byte integer in binary
        fwrite(&previous, 1, 1, stdout); // Writes the ASCII character
        count = 1; // Resetting counter (next will be the first occurrence)
    }
    previous = current;
}

```

Nyt kaikki näyttää toimivan halutulla tavalla.

```

markus@markus-X550CL:~/KYTEEMI/Project2$ gcc -o my-zip my-zip.c -Wall -Werror
markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-zip tobezipped.txt > zipped.txt
markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-zip tobezipped.txt
a
markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-unzip zipped.txt
aaaaaaaaaabb
markus@markus-X550CL:~/KYTEEMI/Project2$

```

Lisätään vielä koodin loppuun rivinvaihdon tulostus niin ohjelman loppuminen näyttää mukavammalta.

```

markus@markus-X550CL:~/KYTEEMI/Project2$ ./my-unzip zipped.txt
aaaaaaaaaabb
markus@markus-X550CL:~/KYTEEMI/Project2$

```

Project 3: Unix Shell

Aloitetaan luomalla pohja shellille niin, että se vain lukee käyttäjän syötteitä ja jos syöte on "exit", se lopettaa.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

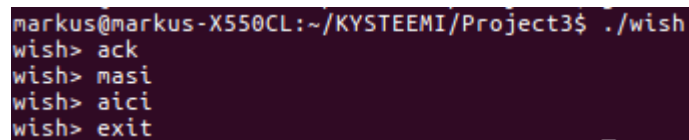
int main(int argc, char *argv[]) {
    char *line = NULL;
    size_t len = 0;
    ssize_t nread;

    while(1) {
        printf("wish> ");
        nread = getline(&line, &len, stdin); // length of the read line
        if (nread == -1) { // If reading the line didn't work
            break;
        }

        // Remove the newline character
        if (line[nread - 1] == '\n') {
            line[nread - 1] = '\0'; // Replace with empty
        }

        // Check for exit command
        if (strcmp(line, "exit") == 0) {
            break;
        }
    }

    free(line);
    return 0;
}
```



```
markus@markus-X550CL:~/KYTEEMI/Project3$ ./wish
wish> ack
wish> masi
wish> aici
wish> exit
```

Seuraavaksi komennot on pilkottava paloihin, tokeneiksi (esim aptitude -v moo on kolme tokenia, "aptitude", "-v" ja "moo") ja on luotava fork, jossa suoritetaan lapsiprosessi.

```

// Now the command and the arguments need to be tokenized
char *args[100]; // Here tokens will be saved
int i = 0; // Counter for the tokens
char *token = strtok(line, " "); // First token
while (token != NULL) {
    args[i] = token; // Saves token
    i++;
    token = strtok(NULL, " "); // Next token
}
args[i] = NULL; // Last one is NULL

// This creates a child process that will execute the command
pid_t pid = fork();
if (pid == 0) { // Child process
    execvp(args[0], args); // Execute the command
    // If it fails
    perror("execvp");
    exit(EXIT_FAILURE);
} else if (pid > 0) { // Parent process
    wait(NULL); // Waits for the child process to be done
} else {
    // If fork failed
    perror("fork");
    exit(EXIT_FAILURE);
}

free(line);
return 0;
}

```

Se toimii:

```

markus@markus-X550CL:~/KYTEEMI/Project3$ ./wish
wish> hello
execvp: No such file or directory
wish> echo "Hello"
"Hello"
wish> ls
wish wish.c
wish> apt-get moo

      (__)
      (oo)
    /-----\
   /  |      | \
  * /  |      | \
   \  |      | /
    \-----/
     ~~~~ ~~~~

... "Have you mooed today?" ...
wish> exit
markus@markus-X550CL:~/KYTEEMI/Project3$

```

Seuraavaksi lisätään mahdollisuus lisätä uusia patheja sisäänrakennetulla path-komennolla ja vaihtaa kansiota cd:llä. Lisätään myös virheitä varten tulostus, jota käytetään kaikkien virheiden tulostukseen.

```

void handle_error(void) { // An error handler for basically everything
    char error_message[30] = "An error has occurred\n";
    write(STDERR_FILENO, error_message, strlen(error_message));
}

```

Path:

```

void initialize_path() {
    path[0] = strdup("/bin");
}

void set_path(char *new_paths) {
    // Free previously allocated path memory
    for (int i = 0; i < path_count; i++) {
        free(path[i]);
    }
    path_count = 0;

    // If there are no new paths, set it to default
    if (new_paths == NULL || strlen(new_paths) == 0) {
        path[0] = strdup("/bin");
        path_count = 1;
        return;
    }

    // Add new paths
    char path_str[MAX_PATHS * 100]; // Make sure the buffer is large enough
    path_str[0] = '\0'; // Initialize the string

    char *token = strtok(new_paths, " ");
    while (token != NULL) { // All the defined paths are added
        if (path_count < MAX_PATHS) {
            path[path_count] = strdup(token);
            strcat(path_str, path[path_count]);
            strcat(path_str, ":"); // Delimiter if many paths
            path_count++;
        } else {
            handle_error();
            break;
        }
        token = strtok(NULL, " ");
    }
    // Take off the colon from the end
    if (strlen(path_str) > 0) {
        path_str[strlen(path_str) - 1] = '\0';
    }
    // Set the PATH environment variable
    setenv("PATH", path_str, 1);
}

```

Path testailua:

```

markus@markus-X550CL:~/KYSTEEI/Project3$ ./wish
wish> ls
wish wish.c
wish> path /usr/local/bin
wish> ls
An error has occurred
wish> cd ..
wish> ls
An error has occurred
wish> path
wish> ls
An error has occurred
wish> path bin
wish> path /bin
wish> ls
Project1 Project3 raportti.abw.bak~ raportti.v2.p1.pdf
Project2 raportti.abw raportti.v1.pdf raportti.v2.pdf
wish> path /bin /usr/bin
wish> ls
Project1 Project3 raportti.abw.bak~ raportti.v2.p1.pdf
Project2 raportti.abw raportti.v1.pdf raportti.v2.pdf
wish>

```

Cd:

```

// Check for cd command
if (strncmp(line, "cd", 2) == 0) {
    char *dir = line + 3; // Skip "cd " part
    if (chdir(dir) != 0) { // chdir changes the directory|
        handle_error(); // The directory is not correct
    }
    continue;
}

```

Cd testailua:

```

wish> pwd
/home/markus/KYSTEEMI
wish> cd Project3
wish> pwd
/home/markus/KYSTEEMI/Project3
wish>

```

Seuraavaksi yritetään saada redirect toimimaan. Tehdään sen käsittelyyn oma funktio. Tässä kohtaa kohtasin monia ongelmia. Lopulta selvisi, että rivi, jossa teen tokenit parametreista, muuttaa muuttujani line vain komennon ensimmäiseksi sanaksi ja kun yritin sen jälkeen liittää linen funktioon, joka etsii redirection merkin ja tiedoston nimen, se ei sitä löydä vaan luulee, että redirectionia ei tehdä ja ottaa esimerkiksi cat komentoon > merkin ja tiedoston johon ohjataan vain tiedostoina, joille cat tulisi tehdä. Näin ollen se tulostaa ensimmäisen tiedoston stdiniin ja antaa errorin ensimmäisestä ja toisesta argumentista. Mutta tämän sai korjattua luomalla kopion linesta ja käyttämällä sitä tokenien teossa.

Tämän lisäksi kirjoittaminen tiedostoon ei ollut itsestäänselvyys, lopulta <https://www.geeksforgeeks.org/input-output-system-calls-c-create-open-close-read-write/> löytäneistä esimerkeistä löytyi tapa, jolla kirjoittaa tiedostoon samalla tavalla kuin redirect > sen tavallisesti tekee eli voi tehdä uuden tiedoston jos sitä ei ole yms.

Lopulta koodi melkoisen tuskan ja työn jälkeen näyttää tältä:

```

void handle_redirection(char *line, char **args, int *redirect_out, char
**tothisfile) {

    char *redirect_pos = strrchr(line, '>'); // Finds > placement
    if (redirect_pos != NULL) { // If '>' is found

        if (strchr(redirect_pos + 1, '>') != NULL) { // Check that only one
            handle_error();
            return;
        }

        *redirect_pos = '\0'; // replace with empty
        *redirect_out = 1; // boolean that yes, we are redirecting

        char *filename_start = redirect_pos + 1; // finding the filename and
taking whitespace off around it
        while (isspace((unsigned char)*filename_start)) {
            filename_start++;
        }

        *tothisfile = filename_start;

        // Find the end of the filename and put empty to it
        char *filename_end = filename_start;
        while (*filename_end && !isspace((unsigned char)*filename_end)) {
            filename_end++;
        }

        if (filename_end != filename_start) {
            *filename_end = '\0';
        }

        // Remove redirection args from args[]
        char *arg;
        int i = 0;
        while ((arg = args[i]) != NULL) {
            if (strcmp(arg, ">") == 0) {
                args[i] = NULL; // End the arguments here
                break;
            }
            i++;
        }
    } else {
        *redirect_out = 0;
        *tothisfile = NULL;
    }
}

```

```

// Create a copy of the original line for redirection handling
char *line_copy = strdup(line);
if (line_copy == NULL) {
    handle_error();
    exit(1);
}

char *args[100]; // Here tokens will be saved
int i = 0; // Counter for the tokens
char *token = strtok(line_copy, " "); // First token

while (token != NULL) {
    args[i] = token; // Saves token
    i++;
    token = strtok(NULL, " "); // Next token
}
args[i] = NULL; // Last one is NULL

// Check for redirection

int redirect_out;
char *tothisfile;
handle_redirection(line, args, &redirect_out, &tothisfile);

pid_t pid = fork();
if (pid == 0) { // Child process

    if (redirect_out && tothisfile) {
        int fd = open(tothisfile, O_WRONLY | O_CREAT | O_TRUNC, 0644); //
        Open file in write mode, source https://www.geeksforgeeks.org/input-output-system-calls-c-create-open-close-read-write/
        if (fd < 0) {
            perror("Error opening file");
            handle_error();
            exit(1);
        }
        dup2(fd, STDOUT_FILENO);
        dup2(fd, STDERR_FILENO);
        close(fd);
    }

    execvp(args[0], args);
    perror("Error executing command");
    handle_error();
    exit(1);
} else if (pid > 0) { // Parent process
    wait(NULL);
} else {

```

Lisätään vielä mahdollisuus lisätä batch-tiedosto:


```

FILE *input_source = stdin; // Default to stdin for interactive mode

// Check if there is a batch file
if (argc > 1) {
    // Open the batch file
    input_source = fopen(argv[1], "r");
    if (input_source == NULL) {
        handle_error();
        exit(1);
    }
}

initialize_path(); // Initialize the path array with default values

while(1) {
    if (input_source == stdin) {
        printf("wish> ");
    }

    // Read a line from the input source
    nread = getline(&line, &len, input_source);
    if (nread == -1) { // End of file or error
        if (input_source != stdin) {
            fclose(input_source);
        }
        break;
    }
}

```

Loin tämänköisen batch-tiedoston:

```

markus@markus-X550CL:~/KYTEEMI/Project3$ cat batch.txt
ls -la
cat hello.txt > moimoi.txt
cat moimoi.txt
ls
cd ..
ls
markus@markus-X550CL:~/KYTEEMI/Project3$

```

```

markus@markus-X550CL:~/KYTEEMI/Project3$ ./wish batch.txt
yhteensä 52
drwxrwxr-x 2 markus markus 4096 heinä 25 23:24 .
drwxrwxr-x 5 markus markus 4096 heinä 25 23:08 ..
-rw-rw-r-- 1 markus markus 61 heinä 25 23:22 batch.txt
-rw-rw-r-- 1 markus markus 19 heinä 25 22:10 hello.txt
-rw-rw-r-- 1 markus markus 58 heinä 25 23:22 moi.txt
-rw-r--r-- 1 markus markus 19 heinä 25 23:01 new.txt
-rwxrwxr-x 1 markus markus 14176 heinä 25 23:24 wish
-rw-rw-r-- 1 markus markus 6494 heinä 25 23:24 wish.c
-rw-r--r-- 1 markus markus 8 heinä 25 23:15 yeah.txt
yks
kaks
kol
testi
batch.txt hello.txt moimoi.txt moi.txt new.txt wish wish.c yeah.txt
Project1 Project3 raportti.v1.pdf raportti.v2.pdf
Project2 raportti.abw raportti.v2.p1.pdf

```

Siispä tämä näyttää toimivan.

En implementoinut mahdollisuutta laittaa useampia komentoja.

