

# Project 3 Unix Shell

## Compiling the program:

1. Go to the directory where wish.c resides
2. Gcc -o wish wish.c -Wall -Werror
3. ./wish or with batch file ./wish [batchFile]

## Different inputs and outputs of the program:

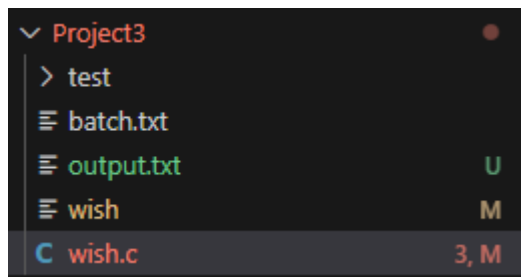
### 1. ./wish

The wish shell opens up a prompt, where you can write the commands that a normal shell would have like ls, cd, mkdir, path, etc. I will go through the basic commands that were requested and a couple of Unix own commands, that are just called from the path variable. No parallel commands are possible in this version of the shell, as I did not get the strtok to work properly.

```
./Project3$ gcc -o wish wish.c -Wall -Werror
./Project3$ ./wish
```

Ls

```
wish> ls
batch.txt  output.txt  test  wish  wish.c
wish> 
```



Project3

- > test
- ≡ batch.txt
- ≡ output.txt U
- ≡ wish M
- C wish.c 3, M

Pwd

```
mattis@mattis-pc:/mnt/c/Users/Mattis/Documents/GitHub/PhaserGame/Kayttojarjestelmat-ja-Systeemi-Ohjelmointi/Project3$ ./wish
wish> hello

wish> pwd
/mnt/c/Users/Mattis/Documents/GitHub/PhaserGame/Kayttojarjestelmat-ja-Systeemi-Ohjelmointi/Project3
wish> 
```

## Path

Setting path to null and trying ls

```
wish> path
wish> ls
could not find the given command in the path variables
wish> █
```

Setting path back to /bin and trying ls

```
wish> path /bin
wish> ls
batch.txt  output.txt  test  wish  wish.c
wish> █
```

## Mkdir

```
wish> dir
batch.txt  output.txt  test  test2  wish  wish.c
wish> mkdir test3
wish> dir
batch.txt  output.txt  test  test2  test3  wish  wish.c
wish> █
```

## Dir

```
wish> dir
batch.txt  output.txt  test  test2  wish  wish.c
wish> mkdir test3
wish> dir
batch.txt  output.txt  test  test2  test3  wish  wish.c
wish> █
```

## Exit

```
wish> exit
mattis@mattis-pc:/mnt/c/Users/Mattis/Documents/GitHub/PhaserGame/Kayttojarjestelmat-ja-Systeemi-Ohjelmointi/Project3$ █
```

## Cd

```
wish> dir
batch.txt  output.txt  test  test2  test3  wish  wish.c
wish> cd ./test
wish> dir
file.txt
wish> mkdir file2.txt
wish> dir
file.txt  file2.txt
wish> 
```

## redirection

```
wish> ls > output.txt
wish> 
```

```
Project3 > ≡ output.txt
1  batch.txt
2  output.txt
3  test
4  test2
5  test3
6  wish
7  wish.c
8
```

## Error cases

### No such file exists

```
i/Project3$ ./wish bat
```

```
marcel@marcel13: ~/Project3$ ./wish bat
Opening the file has failed
*** Error: file not found ***
```

### Too many arguments for exit

```
marcel@marcel13: ~/Project3$ ./wish bat
wish> exit exit
Can't exit due to too many arguments. Write 'exit' to exit the program
wish> 
```

### Wrong input for ls

```
wish> ls -ö
ls: invalid option -- 'ö'
Try 'ls --help' for more information.
wish> █
```

### Wrong input for mkdir

```
wish> mkdir
mkdir: missing operand
Try 'mkdir --help' for more information.
wish> █
```

### Not enough arguments for cd

```
wish> cd
not enough or too many arguments for cd. Write 'cd' '[directory]'
wish> █
```

### Cd directory doesnt exist

```
wish> cd ...
changing directory has failed
wish> █
```

### No such command available

```
wish> mkdör
could not find the given command in the path variables
wish> █
```

### Too many arguments for running the program

```
i/Project3$ ./wish batch.txt batch.txt
```

```
mattis@mattis-pc:~/i/Project3$ ./wish batch.txt batch.txt
too many arguments
```

## 2. ./wish [batchFile]

```
i/Project3$ ./wish batch.txt
```

```
Project3 > ≡ batch.txt
```

```
1  ls
2  pwd
3  cd /tmp
4  ls -l
5  exit
```

```
batch.txt output.txt test test2 test3 wish wish.c
/mnt/c/Users/Mattis/Documents/GitHub/PhaserGame/Kayttojarjestelmat-ja-Systeemi-Ohjelmointi/Project3
total 0
drwxrwxrwx 1 mattis mattis 4096 Mar  3  2022 loose
drwxr-xr-x 1 mattis mattis 4096 Mar  7  2022 mattis
srwxr-xr-x 1 mattis mattis  0 Mar  7  2022 vscode-git-026d73704e.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-git-065cb183b2.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-git-555caca6f9.sock
srwxr-xr-x 1 mattis mattis  0 May  7  2024 vscode-git-7c09138f0f.sock
srwxr-xr-x 1 mattis mattis  0 May  8  2024 vscode-git-94b6455d81.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-git-c18daba62f.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-git-e041f64f07.sock
srwxr-xr-x 1 mattis mattis  0 Mar 24  2022 vscode-git-e580d2b4d7.sock
srwxr-xr-x 1 mattis mattis  0 Mar  6  2022 vscode-ipc-13107867-a71b-4bf4-80d4-d3ceef1cdd46.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-14d37efb-37ee-4400-be09-4601120a21a0.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-1f24236f-76e2-4e6d-bee4-843cdc7ac037.sock
srwxr-xr-x 1 mattis mattis  0 Mar 17  2022 vscode-ipc-1f951e18-6721-472f-a88b-09df6e01bc78.sock
srwxr-xr-x 1 mattis mattis  0 Mar  4  2022 vscode-ipc-47ce17b8-0dc1-462b-a585-c7c93940d9e9.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-557ed010-7659-4cf7-8a03-52707510f1ae.sock
srwxr-xr-x 1 mattis mattis  0 Mar 18  2022 vscode-ipc-605190c0-103d-4816-9410-314bdac49da4.sock
srwxr-xr-x 1 mattis mattis  0 May  7  2024 vscode-ipc-61460753-7e18-4180-83e2-dc940575e331.sock
srwxr-xr-x 1 mattis mattis  0 Mar 24  2022 vscode-ipc-618f14bc-c407-4414-b385-e2ce67d51242.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-74c0af61-c51f-4f40-ad62-803d56d192d8.sock
srwxr-xr-x 1 mattis mattis  0 Mar  6  2022 vscode-ipc-88ff5c46-7672-4fba-9ca4-c6530932e2ce.sock
srwxr-xr-x 1 mattis mattis  0 May  8  2024 vscode-ipc-a931ebef-f044-47a1-a16b-89603b16fc8a.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-bbf8d4b3-4c28-4145-9bcc-94ea537ed3a0.sock
srwxr-xr-x 1 mattis mattis  0 May  7  2024 vscode-ipc-cdc7b9c0-0be6-41a3-8f1b-b3fbf30de1b7.sock
srwxr-xr-x 1 mattis mattis  0 Mar  5  2022 vscode-ipc-ec26267d-44b5-4c2a-8ca9-f15897bed0ff.sock
srwxr-xr-x 1 mattis mattis  0 Mar  3  2022 vscode-ipc-f20b5f38-0027-4017-96f8-a0a567269cc7.sock
srwxr-xr-x 1 mattis mattis  0 Mar 22  2022 vscode-ipc-f5f177a9-bc32-4546-aa37-4cfb70f12114.sock
srwxr-xr-x 1 mattis mattis  0 Mar  7  2022 vscode-ipc-f834a7a1-23dc-4cee-b95f-e318a107a9ce.sock
drwxr-xr-x 1 mattis mattis 4096 Mar  7  2022 vscode-typescript1000
```

## 3. Working principle of the program

This was clearly the hardest task, and I had to look up a lot of help for this project and couldn't make it work with all the given requirements like the parallel commands.

How this program works is it has three built in commands exit, cd and path. These work in any directory or with any path that you have chosen.

How the shell works is it is in a while(1) loop until the exit command is given, or a more fatal error occurs. The while loop checks if there is a file(batch) or if it's just basic stdin. Then it (tries) to parse the input based on "&" marker but it doesn't work as it should. Nevertheless, it takes the line or

command into the `parse_and_execute` function, where it checks if the command is built in or if its the own from Unix. The built ins just check if the argument matches the built in commands name and executes it. on `cd` it uses `chdir` to change the directory. On exit it just checks the amount of arguments and if there is only one argument, it exits the program. For path, we have set up the basic path to `/bin` and you can change the amount of directories in this variable by calling it.

Then it checks if there is a `">"` character and if there is it checks if an output file has been given and then runs the `execute_command` function, which searches for the command in the paths in `paths` variable and executes the command with `execv`. If there is an error, it will print it out. If there is no output file or redirection, the program just uses `execv` to print the result in the `stdin`.

For error management I have made a `print_error` function, that just prints the passed in line to the `stdin`, but the program keeps running.