```
tarjeisv@itstud:~/OS2022/labs/lab6$ tarjeisv@itstud:~/OS2022/labs/lab6$ gcc -pthread -o Exercise1 Exercise1.c
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hello, the world is round and this is message 1.
Hei, redbull gives you wings and this is message 2.
this is main
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hello, the world is round and this is message 1.
Hei, redbull gives you wings and this is message 2.
this is main
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hei, redbull gives you wings and this is message 2.
Hello, the world is round and this is message 1.
this is main
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hello, the world is round and this is message 1.
this is main
Hei, redbull gives you wings and this is message 2.
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hello, the world is round and this is message 1.
this is main
Hei, redbull gives you wings and this is message 2.
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise1
Hello, the world is round and this is message 1.
Hei, redbull gives you wings and this is message 2.
this is main
```

You can see that what message is beeing printed first is random, this is because of multi-threading. The code has no logic that makes another message wait for the other so its random witch message is printed first. But the first threaded message will always come first since the thread_join is before the main message but after the second thread_join.

```
tarjeisv@itstud:~/OS2022/labs/lab6$ gcc -pthread -o Exercise2 Exercise2.c
tarjeisv@itstud:~/OS2022/labs/lab6$ ./Exercise2
inital value: Balance= 1000
how much do you want to deposit?
100
depositing 100 kr
deposit completed
new balance: 1100
how much do you want to withdraw?
600
withdrawing 600 kr
withdraw completed
new balance: 500
```

You can see that its not random witch code is ran first, even tho its threads, this is because of the user input for withdraw beeing after pthread join for deposit. And the balance show the right amount.