OD - LAB 8: Multitasking

Inlcluded files.

- code1.c
- code2.c
- code3.c
- code4.c
- code5.c
- makefile

Exercise 1:

Your tasks:

- Make a Dockerfile that:
  - Includes the provided C files
  - Installs the required packages
  - Compiles the provided C files
- Access the container and run the compiled scripts.

Creating the dockerfile:

Dockerfile

```
FROM ubuntu

RUN apt-get update && apt-get install build-essential -y

COPY . .

RUN make all
```

Building the dockerfile

```
$ docker build -t exercise1 .
```

Accessing the container

```
docker run -it exercise1 bash
```

Running the compiled scripts

```
$ ./code1
$ ./code2
$ ./code3
..
```

Exercise 2-5

Your tasks:

- Run code1 and examine the output
- What is happening?
- What does the pthread_create do? What are the arguments?

Your tasks:

- Run code2 and examine the output
- What is different from before?
- What is a mutex and how does it work?

Your tasks:

- Run code3 and examine the output
- What is happening?
- Why does the method count to 1000 twice?

Your tasks:

- Run code4 and examine the output
- What is happening?
- Look at the code, why is it happening? Can you fix it?

Exercise 6

Your tasks:

- Install htop in your container with the C code
- Run code5 in the container, and examine the cores with htop
- Edit the code, to make the threads run on the same core
- Examine what is happening with htop
- Try killing the process

Installing htop in our container

Dockerfile

```
FROM ubuntu

RUN apt-get update && apt-get install build-essential -y

RUN apt-get install htop -y

COPY . .

RUN make all
```

Building the dockerfile

```
$ docker build -t exercise1 .
```

Accessing the container

```
$ docker run -it exercise1 bash
```

Running code 5 in our container

```
$ ./code5 &
```

Examing the cores with htop

```
$ htop
```

In order to let the threads run on the same core, we have to change line 25 inside the code1.c

```
25 CPU_SET(1, &cpusetB);
```

Should be changed to

```
CPU_SET(0, &cpusetB);
```

Building the dockerfile

```
$ docker build -t exercise1 .
```

Accessing the container

```
$ docker run -it exercise1 bash
```

Running code 5 in our container

```
$ ./code5 &
```

Examing the cores with htop

```
$ htop
```

Killing the process:

1. Hit Fn + F9 (kill)
2. Choose "Sigkill"
