Parallel Computing seminar

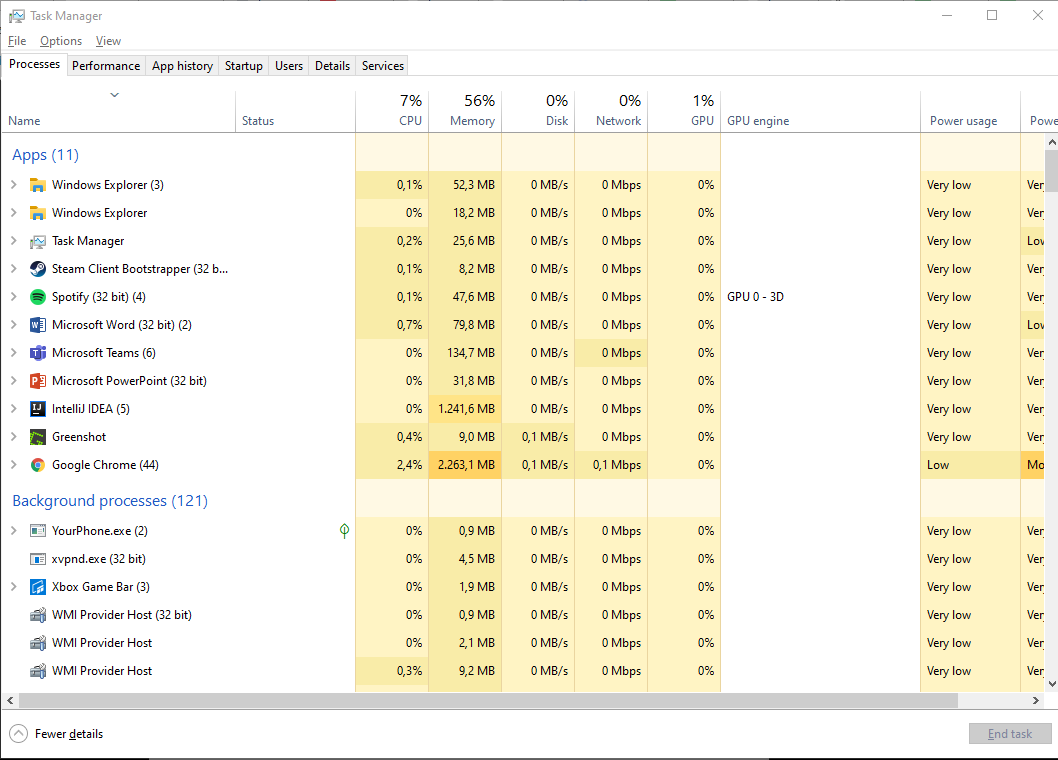
Processes, threads and threadpools

# Exercise 1: Exploring processes

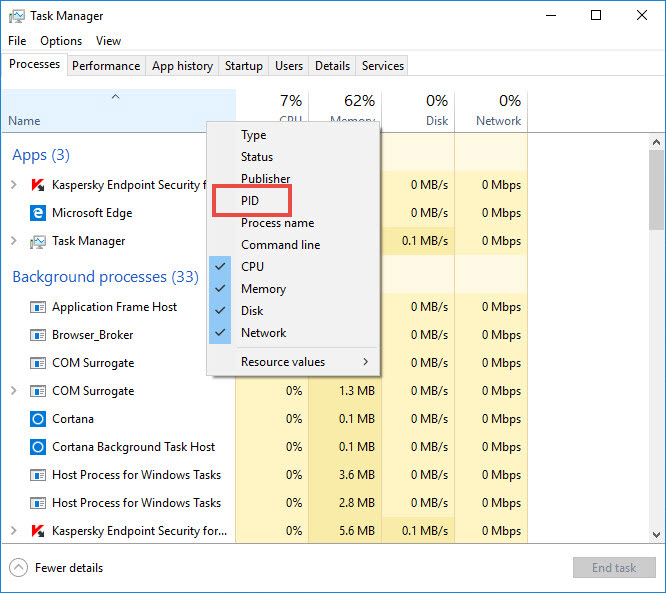
## Exercise 1A

If you are on Windows you can explore the processes running on your system by using the task manager.

Start the task manager and click on “More details” in the lower left of the window. You will see something like this:



Make sure you add the processid (PID) column to the table. You can enable this by right clicking in the table header!



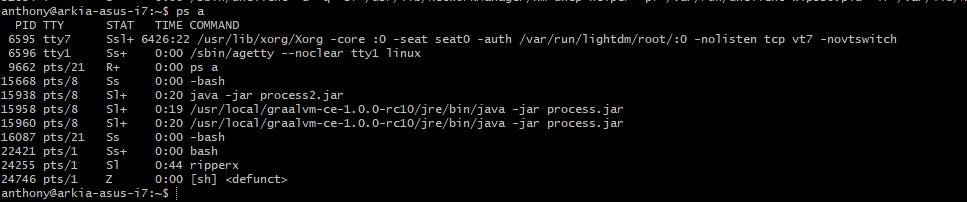
Explore the processes in the window.

How many are there?

Which one uses the most CPU?

Which one uses the most Memory?

On mac or Linux you can use the command ps a to view the processes you started and ps ax to view all processes.



The first column shows you the processid.

## Exercise 1B

You will find three folders inside. Two of those folders contain IntelliJ projects and a third folder called ***jars*** which contains two jar files. These jar files where built with the two projects.

One jar file, called timer.jar simply outputs the current time every second.

The second jar, called multitimer.jar file starts two new timer.jar processes. It will output the processid of each process started.

Have a look at the code of both projects so you have an idea of what to expect.

Open the multitimer folder in IntelliJ. (Be careful to open the right folder). Run the program.

Is the output what you expected?

Have a look in the taskmanager or ps how many processes have been started in the terminal.

Kill the first process started by the multitimer.jar process.

What do you expect will happen?

What did happen?

Can you explain your observations?

## Exercise 1C: create your first thread

Both processes are running at the same time and are outputting data.

But Multitimer.jar is now still a single threaded program, so it will only display data from one process.

Change the program so it becomes multithreaded. You can do this by starting a new thread in the **out** method of the Main class. Make sure you give each Thread a unique name!

What output do you expect to see now?

# Exercise 2: Doing more things at once to enhance performance

We are going to explore some use cases of threads in java.

The project downloads a list of images from a website. It then downloads each image and generates a pdf with those images.

We access the webserver via a very slow connection, so it takes a lot of time to download each image.

## Exercise 2A: Explore

Have a look at the Class WebImagePrinter.

This Class gets a list of images from http://concurrency.anthonyvandenberg.nl/ and saves these images to a pdf (test.pdf).

How many seconds does it take to print the pdf on your system?

You should use the Classes [Instant](https://docs.oracle.com/en%2Fjava%2Fjavase%2F21%2Fdocs%2Fapi%2F%2F/java.base/java/time/Instant.html) and [Duration](https://docs.oracle.com/en%2Fjava%2Fjavase%2F21%2Fdocs%2Fapi%2F%2F/java.base/java/time/Duration.html) to measure this.

# Exercise 2B: Improve the speed

You cannot improve the speed of the connection. But you can do more things at once. We could try to download all the images concurrently.

Change the program so that it uses the Thread Class to download the images concurrently.

Each image should be downloaded in its own thread.

Before you can generate the pdf all images should be downloaded. So we will need to wait for all the threads to finish downloading before we can print the pdf.

How many seconds does it take to print the pdf using threads on your system?