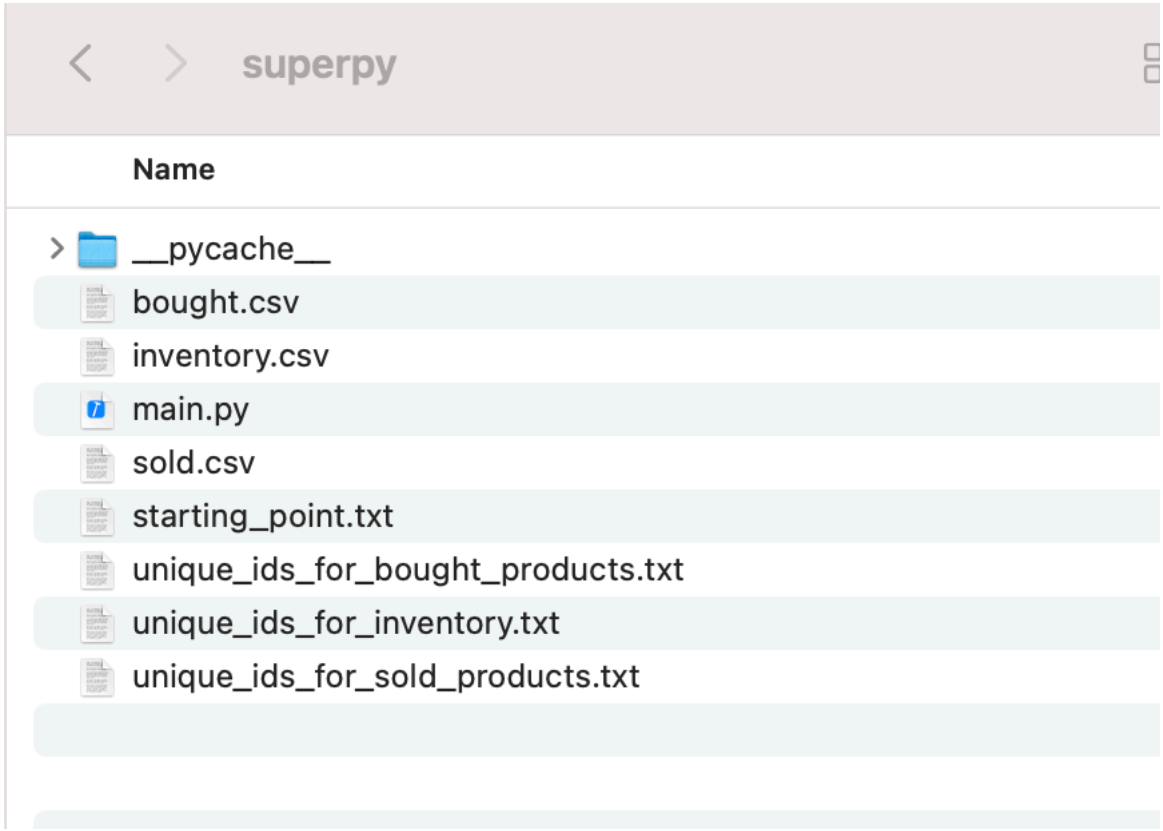
Manual SuperPy Klaas Jan Kostelijk

Okay, let's start! Running 'python3 main.py' in the directory of this assignment will generate all the files you'll need to work with my version of SuperPy.



Let's buy a couple of products. You can either chose short arguments or long arguments, whatever you prefer. You can do this with every argument.

Short arguments:

```
klaas@Klaass-iMac superpy % python3 main.py buy -pn orange -bp 2 -ed 2021-08-06
klaas@Klaass-iMac superpy % python3 main.py buy -pn orange -bp 2 -ed 2021-08-06
klaas@Klaass-iMac superpy % python3 main.py buy -pn apple -bp 2 -ed 2021-08-06
klaas@Klaass-iMac superpy % python3 main.py buy -pn apple -bp 2 -ed 2021-08-06
klaas@Klaass-iMac superpy % python3 main.py buy -pn apple -bp 2 -ed 2021-08-06
```

Long arguments:

```
klaas@Klaass-iMac superpy % python3 main.py buy --product_name light_bulb --buying_price 3 --expiration_date 2024-07-06
klaas@Klaass-iMac superpy % python3 main.py buy --product_name light_bulb --buying_price 3 --expiration_date 2024-07-06
```

Running this command will make sure the purchases are tracked automatically by adding the purchased products to the file 'bought.csv':

bought

id	product_name	buying_date	buying_price	expiration_date
1	orange	2021-07-08	2.0	2021-08-06
2	orange	2021-07-08	2.0	2021-08-06
3	apple	2021-07-08	2.0	2021-08-06
4	apple	2021-07-08	2.0	2021-08-06
5	apple	2021-07-08	2.0	2021-08-06
6	light_bulb	2021-07-08	3.0	2024-07-06
7	light_bulb	2021-07-08	3.0	2024-07-06

Because a bought product is automatically part of the inventory, these products are automatically added to the file ‘inventory.csv’:

inventory

id	product_name	buying_price	expiration_date
1	orange	2.0	2021-08-06
2	orange	2.0	2021-08-06
3	apple	2.0	2021-08-06
4	apple	2.0	2021-08-06
5	apple	2.0	2021-08-06
6	light_bulb	3.0	2024-07-06
7	light_bulb	3.0	2024-07-06

Getting the different products currently present in the inventory:

```
klaas@Klaass-iMac superpy % python3
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from main import get_products_currently_in_inventory
>>> get_products_currently_in_inventory()
Overview of products currently in our inventory:
['orange' 'apple' 'light_bulb']
```

Getting the amount of items per product:

```
>>> from main import get_count_per_product
>>> get_count_per_product()
Number of products currently in our inventory:
product_name
apple         3
light_bulb    2
orange        2
dtype: int64
```

Selling a product requires to check whether or not it's part of the inventory and whether or not this product isn't expired. First, I'm going to try to sell chips (something we do not have):

```
klaas@Klaass-iMac superpy % python3 main.py sell -pn chips -sp 3
Life's tough sometimes: product not part of the inventory or expired
```

Let's sell an orange at a ridiculous price:

```
klaas@Klaass-iMac superpy % python3 main.py sell --product_name orange --selling_price 10
Product sold
```

Now, this orange is no longer part of the inventory:

inventory

id	product_name	buying_price	expiration_date
2	orange	2.0	2021-08-06
3	apple	2.0	2021-08-06
4	apple	2.0	2021-08-06
5	apple	2.0	2021-08-06
6	light_bulb	3.0	2024-07-06
7	light_bulb	3.0	2024-07-06

This sale is now tracked in the file called 'sold.csv':

sold

id	product_name	selling_date	selling_price
1	orange	2021-07-08	10.0

Now, let's try to sell an apple of which we know it's expired after advancing time:

```
klaas@Klaass-iMac superpy % python3 main.py sell --product_name apple --selling_price 10.0
Life's tough sometimes: product not part of the inventory or expired
```

You might have noticed the colours (red and green so far). This is the result of using the external module 'Rich'. I've also added the external module called 'Halo' inside the function called 'send_email'. The ability to send emails with my version of SuperPy and the ability to send the .csv files by email. I use Gmail for that. The function is protected by the password I shared via the Winc Academy board.

Let's send an email with attachment! But first, I'm going to try to send a .csv file that doesn't exist: inventor.csv:

```
klaas@Klaass-iMac superpy % python3 main.py send_email -email_address klaasjankostelijk@gmail.com -sub subject -bd body -att inventor.csv
You can only send bought.csv, sold.csv or inventory.csv
```

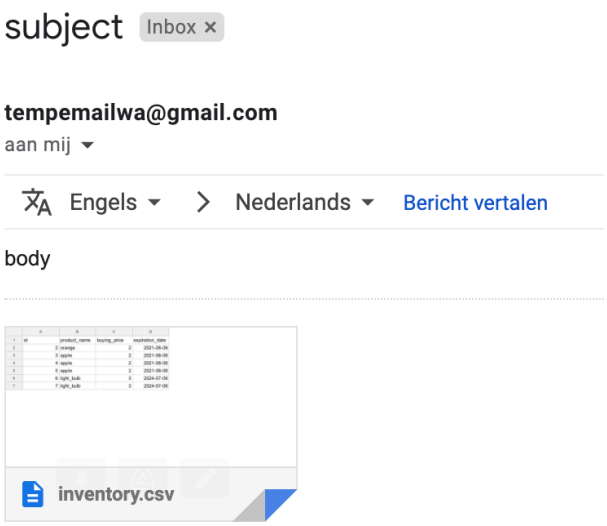
Now, I'm going to try to send a .csv file that does exist: inventory.csv. It will ask for the password:

```
klaas@Klaass-iMac superpy % python3 main.py send_email -email_address klaasjankostelijk@gmail.com -sub subject -bd body -att inventory.csv
Password:
```

After entering the correct password, the email is sent:

```
klaas@Klaass-iMac superpy % python3 main.py send_email -email_address klaasjankostelijk@gmail.com -sub subject -bd body -att inventory.csv
Password:
⠸ Loading
Email sent to klaasjankostelijk@gmail.com
```

The result:



The application can also keep track of revenue, expenses and profit. Because the idea behind these is the same, I'm only showing profit. This is the difference between revenue minus expenses, either on a specific date or over a specific period.

An example of our revenue a specific date:

```
klaas@Klaass-iMac superpy % python3 main.py report_revenue_of_given_day --day 2021-07-08
Today's revenue: EUR  10.0
```

Our profit (well, loss…) of a given time period:

```
klaas@Klaass-iMac superpy % python3 main.py report_profit_of_time_period -fd 2021-07-01 -td 2021-08-01
Revenue over specified time period: EUR 10.0
Expenses over specified time period: EUR 16.0
Profit over specified time period: EUR -6.0
```

Let's advance time by 60 days:

```
klaas@Klaass-iMac superpy % python3 main.py advance_time -d 60
The current date in YYYY-MM-DD is 2021-07-08
Adding 60 days to today's date, resulting in 2021-09-06
```

All of a sudden, it's the 6th of September 2021:

```
klaas@Klaass-iMac superpy % python3
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from main import get_current_date
>>> get_current_date()
datetime.date(2021, 9, 6)
>>>
```

We now know that our apples and oranges are expired. This will prevent you from selling them. Light bulbs, however, can still be sold (because they're not expired):

```
klaas@Klaass-iMac superpy % python3 main.py sell --product_name orange --selling_price 10
Life's tough sometimes: product not part of the inventory or expired
klaas@Klaass-iMac superpy % python3 main.py sell --product_name light_bulb --selling_price 10
Product sold
None
```

One of the requirements of this assignments is to be able to export a selection of data to .csv files. I've created the ability to write expired products to a separate .csv file. Running the following piece of code will make sure that expired products (products of which the expiration date is smaller than the current date) are written to a separate .csv file. This .csv file will also contain the ids these products have in inventory.csv. In the real world, an employee with a barcode scanner will then be able to find the expired products based on

those ids.

```
klaas@Klaass-iMac superpy % python3
Python 3.9.2 (v3.9.2:1a79785e3e, Feb 19 2021, 09:06:10)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from main import export_expired_products
>>> export_expired_products()
```

Result:

expired_products

2	**2021-08-06 00:00:00**
3	2021-08-06 00:00:00
4	2021-08-06 00:00:00
5	2021-08-06 00:00:00