#Report 1
Even though I hardly did anything related to the project itself, I've studied a lot of linear algebra and numpy. I've started the course "data-analysis with python" as it greatly complements the skills I'm going to learn in this eigenface project, and I count finding and learning from that resource as a part of this project. It even covers PCA.

I'm still thinking if it's going to be sensible to use ready made libs for the matrix and vector datastructures. Implementing them myself would definitely be interesting, and I'd like to have some guidance related to this part. The problem is that I don't really know if implementing them is really related to the course, and hence would the time be 'wasted' in relation to the evaluation?

The biggest challenge in this project will be meeting the prerequisites of the mathematic background necessary, but learning PCA does not seem impossible within the required timeframe. I don't quite understand it, but I feel like I'm going to figure it out well enough, hopefully within two weeks max.

I also read up on tests, so I will be ready to implement them in due time, within the next week most likely.

I also missed this deadline due to my own absent-mindedness : (

The amount of hours spent is difficult to evaluate as a lot of it was digging theory up that is also related to other courses. This repository was established within a few hours, but learning the required python skills and linear algebra could have taken 10-20 hours.

#Report 2
friday:
On friday I finished the basic vector data structure along with required tests for it, as well as a rudimentary image-greyscale-vector translation. Vectors can also calculate the SSQ and magnitude from their components, and operations for addition, deduction and multiplication are implemented.

Hopefully on saturday I'll be able to try and create either all matrix logic.

saturday:
I managed to not only create the requisite matrix class but also am able to translate vectors to images, and calculate averages for vectors, and I installed a pep-8 checker. Also basic matrix operations such as transposes and row-means are implemented. I have not yet optimized the operations or done time complexity analysis on them, and will have to do it next week.

Next week I'll aim to code the functionality of the PCA algorithm in most parts. I'll also flesh up the required tests for the matrix data structure as I got carried off-track by the coolness of the ability to calculate average faces (even tried it with some friends of mine :DD). However the real cool factor will become apparent when I'll have implemented the ability to extract the eigenfaces from data. Basically I hope I'll be able to deal with the meat of the project during the upcoming two weeks.

I am a bit confused on how precisely I'll analyze the PCA algorithm, but I am hoping that this query clears itself on its own as I start working on the project once more. If the reader has any hints to give on this then that'd be swell.

I'd approximate I spent some 15-20 hours on the tasks, adding to it the time spent on learning linear algebra (course II) on the side as well as miscellaneous topics (watching videos on eigenfaces, PCA, messing around with average faces) I'd approximate 25.

#Report 3
Was sick and failed to deliver an assignment

#Report 4
During the week I managed to complete a few things

-I configured coverage and poetry into the python project. Had to make a new repository for this.
-Testing document has been added as the coverage html file "tests.html"
-I have an approximate test coverage of 90% on my main code with the lacking 10% being tests for error messages that I didn't deem to be prioritizable; will finish them for the final submission
-I have used python PIL and Numpy libraries to test my code; all matrix calculations are directly tested using Numpy
-Code is thoroughly commented
-My program can compute covariance matrices for any sized matrix, which is the biggest progress I've made thus far
-Eigenvalue decomposition is underway

Generally now it all depends on how quickly I'll be able to pull together the eigenvalue decomposition, and then the face recognition itself. Nothing is unclear at this point, just a lot to do.

Hours spent, 20 hours almost precisly

#Report 5
Once again the complexity of this project has slowed me down, but I've made progress!

Covariance matrices had bugs so I spend this week fixing that mostly. Additionally I spent a lot of time just refactoring the main program. Eigenvalue decomposition implemented, but only with python libraries. Self-made implementation of QR decomposition is the last step for the project along with PCA. To reflect, the algorithms and data structures thus far I've manage to implement are as follows:

- Vectors: a list that can contain an N amount of integer or rgb values. Contains all operations for multiplication, division, subtraction and addition with both reals and vectors. Images can be converted to vectors; Vectors can be converted to images.
    - **Dot product | dot():** time complexity: O(2N) or O(N) where N is the amount of components in the vector.

- Matrices: a list of vectors that can be described as an NxN array. All operations that are implemented for vectors are also implemented for matrices, though differently enough that strict inheritance was not justified.
- Imagematrices: matrices that hold matrices.
    - Mean value | **meanvector():** time complexity:
        - **Covariance matrix | covariance_matrix():** time complexity: O**(**NM) where N is the amount of matrices in the imagematrix, and

Everything is clear, just have to implement all of the requisites.

Hours spent hover around 15-20 hours

#Report 6
Face recognition algorithm is nearly implemented! However, the data could be cleaner now, and I've not yet chosen thresholds for judging whether or not something is a face. It does see that the HU logo is not a face, or at least ranks it far away from the cluster of faces in the face space, while judging that a videogame character is a face, which I suppose is up for interpretation.

The biggest problem is that I cropped the Yalefaces dataset carelessly, and that is severely disrupting the performance of the algorithm. I think I'll have to spend an hour or two next week just cleaning the data, sadly.

Otherwise the algorithm has a decent amount of accuracy given perfect circumstances. It's ability to do recognition is hidden behind lackluster data at the moment, but the implementation is set par from threshold quantities for recognition.

So basically the finishing touches will be:
-Clean the data (crop images correctly)
-Implement thresholds for judging if something is a face or not, or is a known face or not
-Create a system that checks the accuracy of the algorithm

Hours spent: 8-10