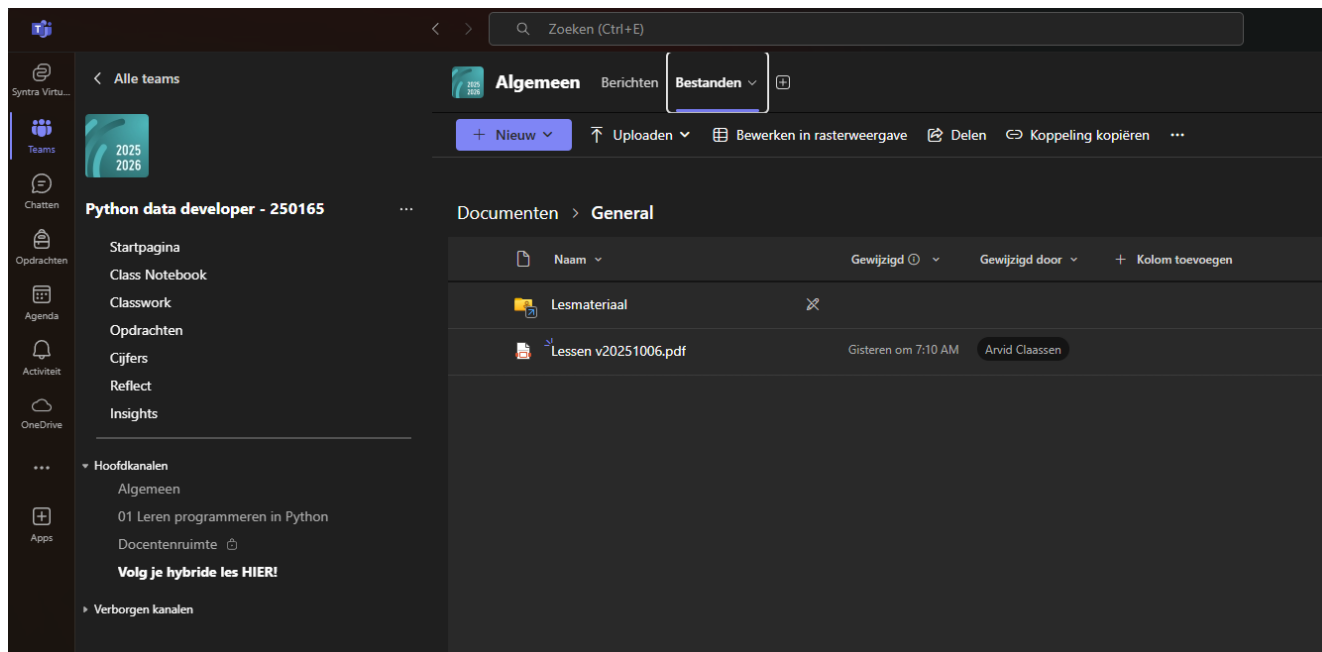


Evaluaties

PYTHON DATA DEVELOPER

De les data met onderwerpen per les zijn terug te vinden op teams onder kanaal 'Algemeen' > Bestanden.



De cursus bestaat uit modules en per module is er een evaluatie.

Module	Totaal score	Evaluatie vorm
Functioneel Programmeren	100	Opdracht + Online Test
Object Oriëntatie	100	Opdracht + Mondelinge Test
Databanken	100	Opdracht + Mondelinge Test
Numpy	100	Opdracht
Pandas	100	Opdracht
Data Exploratie	100	Opdracht
Eindproef	400	Opdracht + Presentatie

Het jaartotaal: 1000 punten waarvan 400 voor de eindproef

Functioneel programmeren

Doelen:

- Een IDE opzetten en gebruiken, inclusief GIT
- Een algoritme schrijven dat een specifiek taak uit voert
- Gebruik maken van
 - o variabelen en constanten
 - o eenvoudige en collectie datatypes
 - o lussen
 - o iteraties
 - o files
 - o built-in functions
 - o modules
- Een gestructureerd programma schrijven
- PEP8 toepassen
- Logisch nadenken over functie-correctheid en complexiteit

Evaluatie vormen

Onderdeel	Evalutie	Punten
Python	Online, voornamelijk multiple choice, met gis correctie	40
Logica	Online, invul velden	30
Programmeer opdracht	In te dienen via GIT	30

Opdracht

Op teams kan je het bestand `bord.csv` vinden.

Elke lijn in het bestand heeft 7 velden, gescheiden door een puntkomma ;

Veld	Kolom naam	Betekenis	Data Type en betekenis
1	id	Uniek nummer per inzending	Int
2	startdate	Tijdstip waarop gestart werd met invullen	Datum-Tijd
3	datestamp	Tijdstip waarop de antwoorden werden ingediend.	Datum-Tijd Indien leeg, dan is het antwoord niet ingediend.
4	G01 Q02	Antwoord op 'Willekeurig getal?'	Floating point
5	Q00	Antwoord op 'Plaats'	String
6	G01 Q03	Antwoord op 'Kleur die nog niet is ingegeven'	String
7	G01 Q05	Antwoord op 'Woord met een Q'	String

A/ Vraag de gebruiker waar het CSV bestand staat. Blijf dit vragen tot het bestand succesvol kan worden ingelezen of tot de gebruiker 'Stop' typt om het programma te stoppen.

B/ Lees het volledige bestand in. Negeer de eerste lijn met kolom namen. De datastructuur in het geheugen moet een list of list zijn met de variable naam *contents*.

Voorbeeld: de inhoud van *contents* na het inlezen van het bestand

```
[ ['1', '11/09/2025 20:34', '11/09/2025 20:37', '42', 'Merelbeke', 'lichtgroen', 'Queeste'],
  ['2', '12/09/2025 7:26', '12/09/2025 7:27', '-3', 'Hogwarts', 'Oker', 'quebec'],
  ['3', '12/09/2025 9:15', '12/09/2025 9:17', '8', 'Rimini', 'blauw', 'Quiz'],
  ...
]
```

C/ Daarna worden aan de gebruiker 10 opties aangeboden:

1) **Print het aantal lijnen:** print het aantal elementen in *contents*

2) **Print de inhoud:** Print de inhoud van *contents* lijn per lijn:

Elke lijn is van de vorm:

Lijn 4, gestart om 15/09/2025 17:03, verzonden op 15/09/2025 17:05, 15, Leuven, Zwart, Quiz

Let op: de waarde in kolom 'id' is niet het lijn nummer. Lijn 4 heeft id 5, lijn 7: heeft id 14

3) **Print datums:** Print de verschillende unieke dagen waarop er inzendingen zijn gedaan.

Volgorde is niet belangrijk. Een datum mag maar 1 keer voorkomen in de lijst.

Tip: Er zijn 22 verschillende dagen.

'20/10/2025', '25/09/2025', '12/09/2025', '12/10/2025', '15/09/2025', ...

4) **Print getal stats:** Print het hoogste, laagste, gemiddelde 'willekeurig getal'. Zorg er voor dat onbestaande getallen niet leiden tot een fout.

Ik kom uit op:

Gemiddelde: 9.345794392523365e+98

Laagste: -78.0

Hoogste: 1e+101

- 5) **Print kleur stats:** Print hoeveel keer een kleur (niet hoofdlettergevoelig) voorkomt in de kolom *kleur*, volgorde is niet belangrijk. Print de kleur die het meeste voorkomt.

Voorbeeld:

LICHTGROEN: 2

OKER: 4

BLAUW: 6

ZWART: 2

De kleur die het meeste voorkomt: PAARS

- 6) **Print plaatsnamen:** print een lijst van alle ingegeven plaatsnamen, zonder dubbels, hoofdletter ongevoelig

- 7) **Print deelname stats:** Print het aantal volledige en onvolledige deelnames.

Bij een onvolledige deelname is *datum verzonden* leeg

Voorbeeld:

Er zijn 137 inzendingen: 112 volledige en 25 onvolledige.

- 8) **Print id stats.** Controleer of de kolom *id* consistent is. Het moet een oplopende rij getallen zijn, zonder ontbrekende waarden. Print welke waardes ontbreken.

Voorbeeld:

Deze waardes ontbreken: [4, 6, 7, 9, 11, 12, 13, 15, 16, 17, 18, 19, 20, 22, 24, 25, 27, 28, 29, 30, 31]

- 9) **Print Q stats:** Print hoeveel q-woorden beginnen met een q, hoeveel worden niet beginnen met een q, hoeveel woorden geen q bevatten

voorbeeld:

106 woorden beginnen met q

6 woorden bevatten een q maar niet als eerste letter

0 bevatten geen Q

- 10) **Save:** Vraag de gebruiker om een bestandsnaam. Indien het bestand gemaakt kan worden, schrijf dan alle inzendingen met een geldig getal en een woord dat met q beginweg naar dat bestand, met velden: id, getal, kleur, q-woord.

1;42;lichtgroen;Queeste

2;-3;Oker;quebec

3;8;blauw;Quiz

Als de gebruiker STOP typt (niet hoofdletter-gevoelig) dan stopt het programma.

D/ Vereisten:

- Code moet voldoen aan PEP8
- Code moet maximaal getypeerd worden
- Voeg relevante commentaar toe
- Alles in 1 bestand in een map *module1*. Bestandsnaam *eval_01.py*
- De werking van elk menu item moet in een afzonderlijke function staan.
- De code moet zo robuust mogelijk zijn: onverwachte gebruikersinput moet behandeld worden.
- In de kolom *willekeurig getal* kunnen komma getallen voorkomen: zowel 3,5 als 3.5 zijn geldig als formaat.
- Je mag het input bestand niet wijzigen. Jouw code zal getest worden met het inputbestand 'bord.csv' dat op teams gepost is.
- Push de oplossing naar github.

E/ Tips:

- Lees en herlees wat elk menu item moet doen
- Is er iets niet duidelijk? Durf te vragen, ik help je met plezier.
- Zit je vast? Vraag hulp. Ik ga je de oplossing niet geven, maar ik kan je wel op weg zetten.
- Zoek eerst zelf voor je AI inschakelt.
- Push op tijd jouw code naar GIT

F/ Punten verdeling: 40punten

- 3 punten per menu item dat correct werkt. (30pt)
- 10 punten voor code kwaliteit: PEP8, commentaar, error handling

Object Oriented programmeren

Doelen:

- De basis concepten van OO
 - o klasse
 - o attributen
 - o class functions
- De hoofdconcepten van OO
 - o Encapsulatie
 - o Overerving
 - o Polymorfisme
- Class hiërarchie kunnen uitwerken
- Factory opzetten
- PEP8 toepassen

Evaluatie vormen

Onderdeel	Evalutie	Punten
Opdracht	Thuis Programmeer opdracht Indienen tegen ten laatste zondag 14/01/2026, 18u Vroeger mag altijd.	40
Extra Opdracht	Tijdens examen op de Syntra campus: Opdracht om 1 extra item bij te programmeren in de thuis opdracht (30 minuten)	20
Mondelinge Bevraging	Tijdens examen op de Syntra campus: Vragen over de ingediende programmeer opdracht Vragen over OO.	40

Opdracht

Rotten Tomatoes (www.rottentomatoes.com) is een commerciële website met film recensies. De naam is afgeleid van een cliché uit het tijdperk waarin het publiek tomaten en andere groenten naar het podium gooide om duidelijk kenbaar te maken dat de voorstelling niet beviel.

De beheerders willen graag een tool om de film-data te analyseren. Ze stellen een gedeeltelijke export van hun website ter beschikking.

Hun bestand ziet er als volgt uit:

```
40 m/10005104-intruder,The Intruder (L'Intrus),6,DRAMA,Claire Denis,2005-12-23,2006-04-25,130,Wellspring,73,9701
41 m/10005178-saint_ralph,Saint Ralph,PG-13,COMEDY,Michael McGowan,2005-08-05,2017-05-11,98,IDP,85,6203
42 m/10005403-the_baxter,The Baxter,PG-13,COMEDY,Michael Showalter,2005-08-26,2018-07-02,91,IFC Films,75,7436
43 m/10005499-oliver_twist,Oliver Twist,PG-13,DRAMA,Roman Polanski,2005-09-30,2012-04-16,130,Sony Pictures,63,23294
44 m/1000560-alamo,The Alamo,NR,ACTION & ADVENTURE,John Wayne,1960-10-24,2016-11-30,192,Batjac Productions,63,7337
```

- Lijn 1 zijn de kolomnamen.
- Elke lijn bevat 11 kolommen

Beschrijving van de kolommen:

Veld	Kolom naam	Betekenis	Data Type en betekenis
1*	rotten_tomatoes_link	Uniek deel van het URL dat naar de film verwijst.	String
2*	movie_title	Volledige titel	String
3*	content_rating	Code voor de filmkeuring	String met beperkt aantal waarden
4*	genres	Filmgenre	String met beperkt aantal waarden
5	directors	1 of meerdere regisseurs, gescheiden door een komma	String
6	original_release_date	Datum waarop de film verscheen in de bioscoop	Datum: YYYY-MM-DD
7	streaming_release_date	Datum waarop de film verscheen op een streaming platfotm.	Datum: YYYY-MM-DD
8	runtime	Lengte in minuten	Integer
9	production_company	Filmproducent	String
10	audience_rating	Gemiddelde score	Integer
11	audience_count	Aantal stemmen	Integer

* velden met een sterretje zijn verplicht, andere velden kunnen leeg zijn

Extra uitleg bij veld 3 – Content rating

De *content rating* wordt door de filmkeuring toegekend aan een film om aan te geven of er leeftijdsbeperkingen zijn.

De mogelijke waardes zijn in oplopende volgorde:

NR: Niet gekeurd

G: Alle leeftijden

PG: Ouderlijk toezicht aangeraden

PG-13: Ouderlijk toezicht ten zeerste aangeraden

R: Onder 17 is ouderlijk toezicht aangeraden

NC17: Enkel voor volwassenen

Als we films moeten ordenen volgens hun rating, dan is NR de laagste rating en NC17 de hoogste rating.

Extra uitleg bij veld 4 – genres

Elke film behoort tot exact 1 genre. Dit zijn de mogelijke waarden:

ACTION & ADVENTURE

COMEDY

DRAMA

HORROR

ROMANCE

SCIENCE FICTION & FANTASY

WESTERN

Bestanden:

Op Teams zijn deze 3 bestanden ter beschikking gesteld:

- 1) **reviews.csv**: de export aangereikt door rottentomatoes.com
- 2) **eval_02.py**: het main bestand. Voor de evaluatie zal de docent dit bestand starten en controleren of de opdracht naar behoren is geïmplementeerd.

Er worden twee constanten gedefinieerd. Je mag deze aanpassen om de code te laten werken op jouw computer:

```
# These constants must point to the input file reviews.csv,
# and the location of the output file (see item 9)
MOVIE_FILE = "D:/dev/learning/reviews.csv"
EXPORT_FILE = "D:/dev/learning/export.csv"
```

Het bevat ook al function `load_movies()`. Daar mag niets aangewijzigd worden.

- 3) **test_movies.py**: unittest bestand dat helpt controleren of jouw code voldoet aan de vereisten. Voer dit bestand zelf uit ter controle. De opdracht classes worden getest op structuur en functionaliteit. In principe zouden alle testen moet slagen.

Opdracht

MovieRating

Maak een file module02/movie/rating.py, met daarin:

1/ Een klasse MovieRating aan de hand van het pattern PredefinedFlyweight. (Zie Grades voorbeeld)

- met minstens attributen: code, description
- code en description mogen niet leeg zijn, anders exception ValueError
- representatie: "Rating(code)" Rating(NR)
- ratings kunnen gesorteerd worden: NR < G < PG < PG-13 < R < NC17
- twee ratings zijn gelijk als de code overeenkomt
- er mogen geen rating objecten gemaakt met eenzelfde code

Voor elk van de 6 mogelijke waardes moet reeds een object bestaan (PredefinedFactory)

2/ Maak een function get_rating() die op basis van de rating code het bijbehorende rating object teruggeeft.

```
def get_rating(code: str) -> MovieRating: 3 usages  docent-12test *
    """
    Return a rating object for a given code
    :param code: rating code
    :return: MovieRating object
    """
    pass
```

Person

Maak en file module02/person/person.py, met daarin:

1/ een klasse *Person* aan de hand van het pattern Flyweight Factory.

- met minstens 1 attribuut: fullname
- er mogen geen twee personen met dezelfde fullname zijn
- het mag niet mogelijk zijn om de fullname te wijzigen
- personen zijn gelijk aan elkaar als de fullname gelijk is zonder rekening te houden met hoofdletters en kleine letters
- representatie: Persoon(Arvid Claassen)

2/ function get_person() die op basis van de naam (niet hoofdletter gevoelig) bijbehorende person object teruggeeft.

```
def get_person(full_name: str) -> Person: 7 usages
    """
    Get an existing person or create a new one
    :param full_name: Full name of the person
    """
    pass
```

Movie

Maak een bestand module02/movie/movie.py met daarin:

1/ een Abstract Base Class (ABC) **Movie**

- met minstens deze attributen van het correcte type:

Attribuut naam	Veld in CSV	Datatype
rt_link	1	Str – niet leeg
title	2	Str – niet leeg
Rating	3	MovieRating – niet leeg
Directors	5	List[Person] of lege lijst
release_date	6	datetime of None
streaming_date	7	datetime of None
length	8	datetime of None
company	9	str of None
score	10	int of None
count	11	int of None

- met minstens de functions

relevant_score(): De score van een film is relevant wanneer er een score gegeven is en bepaald werd door minstens 100 stemmen

is_classic(): Een film is een klassieker als de film minstens 20 jaar oud is en de relevante score hoger is dan 80.

is_short(): Een fim is een kortfilm als hij korter is dan 30 minuten.

url(): Geeft de het internet adres van de film <https://www.rottentomatoes.com/> + rt_link

2/ Maak voor elke genre een subklasse met namen: **ActionAdventure, Comedy, Drama, Horror, Romance, ScienceFictionFantasy, Western**

Sommige subklassen hebben een extra function:

Comedy:

- is_slapstick() : True als relevante score < 40

Romance:

- is_cosy(): True als lengte tussen 70 en 100 minuten is

Horror:

- is_scary(): True als rating > PG is

3) een factory function create_movie(movie_info:dict) -> Movie die op basis van het genre een object van de juiste klasse retourneert. Als het genre niet bestaat: raise ValueError. De input parameter movie_info is de dict van de csv.DictReader (zie load_movies()) in

Eval02

Bij het opstarten van eval02.py moet het bestand reviews.csv ingelezen worden, met behulp van de function load_movies()

Je krijgt deze functie op voorhand. Je mag hier niets aan wijzigen.

```
def load_movies(file_location: str) -> List[Movie] | None: 1 usage  docent-12test *
    """
    Load all movies from a csv file into a list of Movie objects.
    :param file_location: Location of the csv file
    :raise FileNotFoundError: When the file cannot be found
    :return: List of Movie objects or None in case of error
    """
    """
    Als het bestand niet bestaat, raise FileNotFoundError.
    Lees het opgegeven bestand in via een csv DictReader uit de module csv.
    Voor elke rij in het bestand:
        maak het juiste Movie-object aan: Drama, Comedy, Horror, ...
        voeg het object toe aan de lijst movies
    Retourneer de lijst movies
    """
    pass
```

B/ Daarna worden aan de gebruiker 10 opties aangeboden:

- 1) **Print het aantal films:** print het aantal films in het input bestand
- 2) **Print de films per genre:** Print een lijst van genres en hoeveel keer dat genre voorkomt, gesorteerd op aantal. Het meest voorkomende genre eerste.

Fictief voorbeeld:

ActionAdventure : 200

Comedy : 150

ScienceFictionFantasy : 15

Romance : 5

Western : 5

Horror : 2

- 3) **Aantal personen:** Print hoeveel objecten van de class Person er zijn aangemaakt.
- 4) **Hoogste score:** Print de film(s) met de hoogste relevante score.
- 5) **Actiefste regisseur:** Print de regisseur(s) die de meeste films regisseerde(n)
- 6) **Kortste en langste film:** Print de kortste en langste film(s)

- 7) **Enge horror:** Print alle horror films die eng/scary zijn.
- 8) **Scorelijst:** print alle scores van 0 tot 100 en hoeveel films deze score hebben.

Fictief ingekort voorbeeld:

1%: 2
 2%: 4
 ...
 78%: 12
 79%: 7
 ...
 99%: 0
 100%: 0

- 9) **Exporteer:** Maak een CSV bestand met daarin de film informatie van de films zonder relevante score, in alfabetische volgorde.
- 10) **Stop :** Einde van het programma

Vereisten:

- Alle bestanden in een map *module02*, als volgt
 - module02/eval_02.py*
 - module02/movie/movie.py*
 - module02/movie/rating.py*
 - module02/movie/test_movies.py*
 - module02/person/person.py*
- Als je nog andere bestanden wil maken, moeten deze ook in de map *module02* staan
- Code moet voldoen aan PEP8
- Code moet maximaal getypeerd worden
- Voeg relevante commentaar toe
- De werking van elk menu item moet in een afzonderlijke function staan in *eval_02.py*
- Elk van deze functies heeft een lijst van films als input. Mogelijke uitzondering is menu item 3.
- De code moet zo robuust mogelijk zijn: onverwachte gebruikersinput moet behandeld worden.
- Je mag het input bestand niet wijzigen. Jouw code zal getest worden met het inputbestand 'reviews.csv' dat op teams gepost is.
- De oplossing moet beschikbaar gemaakt worden github. **Deze keer geen uitzonderingen!** Probeer dus op tijd of de code naar github gepusht kan worden en beschikbaar voor de docent.

Tips:

- Lees en herlees wat elk menu item moet doen
- Is er iets niet duidelijk? Durf te vragen, ik help je met plezier.
- Zit je vast? Vraag hulp. Ik ga je de oplossing niet geven, maar ik kan je wel op weg zetten.
- Zoek eerst zelf voor je AI inschakelt.
- Geen enkel menu item doet iets met de twee datum velden. Kans is dus groot dat je daarvoor op het examen iets zal moeten doen. Oefen dus zeker
- Push op tijd jouw code naar GIT
- Belangrijk: Dit is de eerste keer dat deze opdracht wordt opgelegd. Er kunnen een dus nog zaken onvoldoende beschreven zijn

Verloop van het on-site examen:

- Breng je laptop mee. Zorg dat de meest recente versie van jouw programmeeropdracht er op staat.
- Het lot bepaalt een bijkomend menu item. Je krijgt 30 minuten om dat menu item te programmeren in jouw opdracht. Er zullen geen aanpassingen nodig zijn aan de klasse structuur: exact één method bij programmeren.
- Het lot bepaalt een paar vragen over OO die je mondeling beantwoordt.
- Je beantwoordt vragen over jouw opdracht. Waarom heb je dit zo gedaan? Wat doet dit stukje uit jouw code?
- Taal: Engels / NL