howest MULTIMEDIA EN COMMUNICATIETECHNOLOGIE	**Voornaam en naam:**	Voorbeeld examen		
	Groep:			
	Opleiding:	**Bachelor MCT**	**Module:**	Basic Programming
	Lector(en):		**Nagelezen door:**	
	Academiejaar:		**Semester:**	1
	Datum en uur:		**Zittijd:**	1

1 LABO EXAMEN (Voorbeeld examen)

1.1 Algemene instructies examen

Dit labo examen maakt 70% uit van de evaluatie voor de module Basic Programming.
Schrijf je naam en voornaam bovenaan deze pagina.

1.1.1 Vooraf

Ga naar Leho om alle bronmateriaal af te halen, dit is een folder die reeds de juiste structuur heeft.
Werk dit project uit in deze folder. In deze folder dienen alle bestanden van je examen te komen.

1.1.2 Opgave

- Lees eerst deze opgave rustig en aandachtig door.
- Maak de toepassing(en) zoals verder in de opgave beschreven.
- Een programma dat afwijkt van deze opgave kan een verlies van punten tot gevolg hebben.
- Volg de gevraagde naamgeving indien de opgave dit vermeldt.
- Let op volgorde van parameters.
- Je maakt gebruik van de programmeertaal Python.
- Je mag gebruik maken van:
  - Alle lesmateriaal op jouw laptop
  - Officiële documentatie op: https://docs.python.org/3/
- Opgelet:
  - Iedere vaststelling van onregelmatigheid (o.a. GSM, spieken) wordt conform het OER gemeld aan de betrokken student en aan de voorzitter van de examencommissie.
- Start: je vertrekt van de github-classroom waarin alle bronmateriaal aanwezig is.
  - Om je repository in onze Github-classroom aan te maken
  - Plaats in elk bestand bovenaan jouw naam, voornaam en groep in commentaar.
- Je werkt op de bronbestanden verder. Voer minimaal twee commits uit nadat je elke oefening voltooid hebt.

1.2 Indienen

- Als je klaar bent of als de voorziene tijdsduur verlopen is:
  Stap 1 (Github):
  - Sla alles op, je commit & pusht alles een laatste keer op jouw repository
  - **We controleren of je voldoende (zinnige) commits geplaatst hebt. De versie op Github wordt niet verbeterd.**

  Stap 2 (Leho):
  - Je zoekt de map op via de verkenner. Controleer of de map alle nodige files bevat.
  - Zip of rar de map.
  - Controleer zeker nog even of alles in het zip/rar-bestand aanwezig is.
  - Dien je project in op Leho bij opdrachten van de module.
  - Controleer of je correct indiende, door je eigen oplossing opnieuw te downloaden.
    Dit is je eigen verantwoordelijkheid!
  - Als dat in orde is geef je deze opgave aan de docent af.
  - **De ingediende versie op Leho wordt verbeterd**.

1.3 Oefening 1

Bestudeer het bestand 'album.json'. Dit bestand bevat de gegevens van (streaming muziek)albums met hun songs.

We maken een kleine console-applicatie in Python die het doorzoeken van al deze informatie vereenvoudigt. Schrijf de data en repository klasses.

Stappenplan: je vetrekt van het bronmateriaal (terug te vinden op Leho). Je wijzigt de folderstructuur niet. De bestanden test_x_x.py mogen niet gewijzigd worden, en moeten vlekkeloos werken (nadat je de nodige imports toegevoegd hebt).

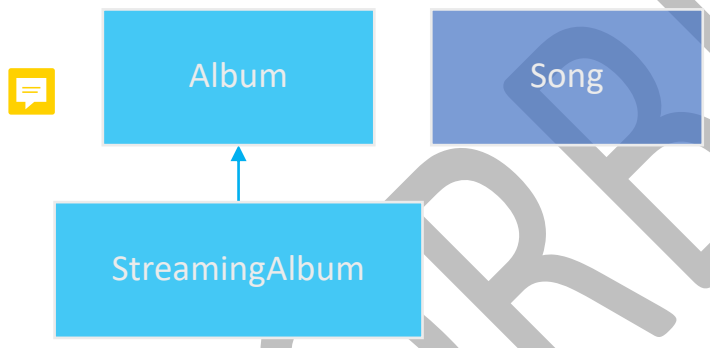
1.3.1 Algemene verkenning van de opdracht, details per klasse volgt later

De volgende klasse is **deels** gegeven:

- De klasse **Album**. Deze klasse houdt de gegevens (*artiest*, *titel* en *jaar*) van een muziekalbum bij.
- Je vervolledigt deze klasse naargelang je hier nood aan hebt.
    - (Straks zal je aan deze klasse een lijst met songs toevoegen)

Maak volgende klassen aan:

- klasse **Song** die het *tracknummer*, *titel* en aantal *seconden* bijhoudt van een song.
- klasse **Streamingalbum,** die **erft** van de klasse **Album.** Deze klasse houdt naast de attributen van Album ook een *id*, aantal maal *afgespeeld* en *uitgelicht* bij.



- Klasse **StreamingAlbumRepository** die instaat voor het inlezen van het json-bestand, en kan zoeken/filteren in een list van objecten van de klasse **StreamingAlbum**.

Voorzie de klasses **StreamingAlbum** en **Song** van de correcte attributen, methodes, getter en setter properties. (zie volgende stap)

Waar vermeld houd je rekening met overerving!

Je past op de juiste plaatsen **exception handling** toe:

- Is de property een geheel of komma getal? Raise dan een exception indien de value een foutief gegevenstype is.

1.3.2 Klasse per klasse

Specifiek voor de klasse Song:

- Voorzie de klasse van de nodige private attributen:
    - *tracknr*, *titel* en *seconden*
- De constructor
    - heeft 3 parameters: *tracknr*, *titel* en *seconden*
- Maak voor elk attribuut een publieke property aan.
    - *tracknr*, *titel* en *seconden* hebben zowel een **getter** als een **setter**.

        - Voeg in de setter controle toe voor het correcte datatype (int of string).
- Er is een extra **get**-property zonder bijhorend attribuut duur_string.
    - Deze geeft een string terug die de duur in seconden omzet naar de string “xx min : xx seconden”.
    - Tip: er zitten meestal 60 seconden in een minuut.
- Objecten van de klasse Song zijn gelijk indien de titel en de duur gelijk zijn.
    - Voeg hiervoor de juiste methode toe.
- Objecten van de klasse Song kunnen geprint worden en getoond worden in een list
    - Output kan bijvoorbeeld: “(1) - Old Town Road - 5 min : 8 sec” zijn

Specifiek voor de klasse Album:

- Vul de klasse Album aan:
    - Aan de parameters van de constructor verander je niets.
    - In de constructor wordt een extra private attribute *songs* aangemaakt dat een lege list is.
    - Voor dit nieuwe attribuut maak je een nieuwe **getter-**property *songs* aan.
- Schrijf een extra methode ‘**voeg_song_toe’**:
    - Deze methode heeft een object van de klasse Song als parameter.
    - Indien de gebruiker een **correcte** parameter meegeeft, voeg je de song toe aan de list *songs*.
    - Indien de gebruiker een parameter meegeeft die **niet** van de **klasse Song** is, creëer je een error.
    - Indien de gebruiker een song wil toevoegen die **reeds op het album staat**, creëer je een error.

Specifiek voor de klasse StreamingAlbum:

- De klasse **StreamingAlbum erft** van de klasse **Album**
- Voorzie de klasse van de nodige **extra** private attributen:
    - *id*, *afgespeeld* en *uitgelicht*
- De constructor
    - heeft 5 **parameters**: *artiest, titel, jaar, id, afgespeeld*
        - de **parameter** *afgespeeld* krijgt een standaardwaarde van 0
    - de **property** uitgelicht wordt op False ingesteld.
- Maak voor elk attribuut een publieke property aan.
    - *Afgespeeld* heeft zowel een **getter** als een **setter**.
        - Voeg in de setter controle toe. Het moet een geheel getal zijn én groter of gelijk aan 0.
    - *Uitgelicht* heeft zowel een **getter** als een **setter**.
        - Voeg in de setter controle toe. Het moet een bool(ean) zijn.
    - *Id* heeft enkel een **getter**
- Voorzie de correcte methode zodat een object van het type StreamingAlbum kan **geprint** worden.
    - De output zal verschillen naargelang het **attribute** *uitgelicht* op True of False staat ingesteld.
        - Bij True: “(!!!) Album *titel* songs: *aantal songs*”
        - Bij False: “(---) Album *titel* songs: *aantal songs*”
- Voorzie dezelfde output indien er een **list van een object** van het type StreamingAlbum wordt geprint.
- Objecten van de klasse StreamingAlbum kunnen gesorteerd worden.
    - De sortering gebeurt op basis van het aantal songs op het album.
- Heeft een extra methode ‘**speel_af’**
    - Deze methode heeft geen parameters.
    - Deze methode verhoogt het aantal keer dat een bepaald album is afgespeeld.

Specifiek voor de klasse StreamingAlbumRepository:

- Een static methode ‘**inlezen_muziek’** die in staat is om het bronbestand in te lezen en een list van StreamingAlbums terug te geven. Verwerk hierin exceptionhandling. Doe waar nodig conversie van string naar float/int.
- een static methode ‘**geef_albumtitels_by_artiest’**
    - met 2 parameters

        - een verzameling van objecten van de klasse StreamingAlbum
        - een artiestennaam.
    - Deze methode geeft een **list** van albumtitels(string) terug van deze artiest zijn.
- een static methode '**geef_alle_songs_die_langer_duren'**
    - met 2 parameters
        - een verzameling van objecten van de klasse StreamingAlbum
        - een minimum duur van een song (in seconden).
    - Deze methode geeft **een list** van objecten van Songs die aan de voorwaarde voldoen.

Maak gebruik van het test-bestand test_zonder_json.py en test_met_json.py om de verschillende functionaliteiten uitvoerig te testen.

Veel Succes!

Gewenste output van de testklasse

test_zonder_json.py

```
******* TEST 1 - SONG *******
Het liedje duurt:       308 sec
Info over liedje:       (1) - Old Town Road - 5 min : 8 sec
Info over hit:          (5) - Old Town Road - 7 min : 5 sec
-------
Test gelijkheid 1
(1) - Old Town Road - 5 min : 8 sec  is NIET aan (5) - Old Town Road - 7 min : 5 sec
Test gelijkheid 2
(1) - Old Town Road - 5 min : 8 sec  is GELIJK aan (12) - Old Town Road - 5 min : 8 sec

******* TEST 2 - Album *******
TIM van Avicii - bevat volgende songs []
TIM van Avicii - bevat volgende songs [(1) - Peace of Mind - 3 min : 0 sec , (2) - Heaven - 4 min : 37 sec , (3) -
- 2 min : 37 sec ]

******* TEST 3 - StreamingAlbum *******
-> maak list
[(---) Album Casco songs: 2, (---) Album When We All Fall Asleep, Where Do We Go songs: 1]
-> sort list
(---) Album When We All Fall Asleep, Where Do We Go songs: 1 - afgespeeld 0
(---) Album Casco songs: 2 - afgespeeld 100
-> licht Billie Eilish uit en speel Brihang af
(!!!) Album When We All Fall Asleep, Where Do We Go songs: 1 - afgespeeld 0
(---) Album Casco songs: 2 - afgespeeld 101

******* TEST 4 - Worden fouten opgevangen? *******
-> song verkeerde input
FOUT: Seconden moet een geheel getal zijn
-> album tweemaal dezelfde song proberen toevoegen
FOUT: Kan song niet toevoegen
Volgende songs staan op Casco - [(3) - Steentje - 3 min : 26 sec ]
-> verkeerde waarde om uit te lichten
FOUT: Uitgelicht kan enkel TRUE / FALSE bevatten
```

test_met_json.py

```
**** Alle ingelezen albums****
(---) Album Brol songs: 12
(---) Album Divide songs: 12
(---) Album Doom Days songs: 11
(---) Album Wild World songs: 14
(---) Album Wildness songs: 10

**** MCT *****
-> Geef de albumTITELS van Bastille
        Doom Days
        Wild World
-> Geef alle songs die langer duren dan 250 sec
        (11) - Flemme - 4 min : 16 sec
        (2) - Castle on the Hill - 4 min : 21 sec
        (5) - Perfect - 4 min : 23 sec
        (11) - How Would You Feel (Paean) - 4 min : 40 sec
        (5) - Million Pieces - 4 min : 11 sec
        (10) - Those Nights - 4 min : 30 sec
```

```
        (5) - Glory - 4 min : 41 sec
        (10) - Four Walls (The Ballad of Perry Smith) - 4 min : 12 sec
        (1) - Life on Earth - 5 min : 22 sec
        (4) - Empress - 4 min : 29 sec
        (5) - A Dark Switch - 4 min : 18 sec
        (8) - Soon - 4 min : 20 sec
        (9) - Wild Horses - 4 min : 38 sec
        (10) - Life and Death - 5 min : 35 sec
```