Čo si pozrieme?

V rámci druhého zadania pomôžete skupine priateľov vybrať si, na ktoré premietanie pôjdu do kina. Precvičíte si pritom základy objektovo orientovaného modelovania, vytvárania tried a prácu s objektmi. V riešení použijete nasledovné modely:

- `Movie` – reprezentuje film;
- `Auditorium` – reprezentuje kinosálu;
- `Screening` – reprezentuje premietanie filmu v konkrétnej sále v konkrétnom čase;
- `Cinema` – reprezentuje kino so sálami a ponúkanými premietaniami;
- `Person` – človek, ktorý chce ísť do kina a chce si pozrieť filmy, ktoré patria do jeho okruhu záujmov;
- `FriendGroup` – skupina priateľov, ktorí chcú ísť spoločne do kina.

Okrem týchto tried riešenie obsahuje súbor `constants.py` so zoznamom platných žánrov filmov, ktoré budú existovať v riešení. Ak chcete zadefinovať ďalšie pomocné premenné, ktoré chcete využívať ako konštanty, môžete tak urobiť v tomto súbore.

Triedy v riešení budú navzájom prepojené, takže zadanie odporúčame najprv prečítať do konca, aby bolo jasné, ako majú jednotlivé komponenty spolupracovať. Poradie popisu tried predstavuje odporúčané poradie implementácie.

`Movie` – 2,5 bodov

Trieda `Movie` reprezentuje film, ktorý sa práve premieta v kinách. Každý film je popísaný niekoľkými hodnotami, ktoré sa nastavujú v už predpripravenom konštruktore:
- `title` (`string`) – názov filmu;
- `length` (`int`) – dĺžka filmu (v minútach);
- `genre` (`string`) – žáner filmu, ktorý musí byť zo zoznamu `constants.GENRES`;
- `age_limit` (`int`) – vekový limit, teda minimálny vek, ktorý musia mať návštevníci kina, aby si daný film mohli pozrieť;
- `release_date` (`string`) – dátum premiéry vo formáte YYYY/MM/DD, napríklad 2023/03/20 pre 20. marec 2023.

Do triedy potrebujete doplniť nasledovnú funkcionalitu:
- v konštruktore zvalidujte hodnoty parametrov pred tým, než sa nastavia hodnoty členských premenných nasledovne:
    - `title` musí byť typu `string`, v opačnom prípade vygenerujte `TypeError` s chybovou hláškou *Movie title must be string*;
    - `length` musí byť typu `integer` (v opačnom prípade vygenerujte `TypeError` s chybovou hláškou *Movie length must be integer*) a minimálne 1 (v opačnom prípade vygenerujte `ValueError` s chybovou hláškou *Movie length must be at least 1*);
    - `genre` musí byť zo zoznamu `constants.GENRES`, v opačnom prípade vygenerujte `ValueError` s hláškou *Unknown genre "XX"*, kde namiesto *XX* napíšete hodnotu parametra `genre` (nezabudnite na úvodzovky);
    - `age_limit` musí byť typu `integer` (v opačnom prípade vygenerujte `TypeError` s hláškou *Age limit must be integer*) a musí byť minimálne 1 (v opačnom prípade vygenerujte `ValueError` s chybovou hláškou *Age limit must be at least 1*);

- `release_date` sa zvaliduje v samostatnej metóde `validate_date`, tá však musí byť zavolaná už v samotnom konštruktore.

- implementujte metódu `validate_date(date)`, ktorá slúži na kontrolu hodnoty parametra `release_date`, pričom tá musí spĺňať nasledovné podmienky:
  - musí byť typu `string`, v opačnom prípade vygenerujte `TypeError` s chybovou hláškou *Release date must be string*;
  - musí obsahovať dva znaky `/`, v opačnom prípade vygenerujte `ValueError` so správou *Release date must meet format YYYY/MM/DD*;
  - prvé štyri znaky pred prvou lomkou reprezentujú rok, dva znaky medzi dvomi lomkami mesiac, a posledné dva znaky deň premiéry – ak niektorý z týchto podreťazcov nedokážete prekonvertovať na `integer`, vygenerujte `ValueError` so správou *Could not load date from string: "XX"*, kde namiesto *XX* vypíšete pôvodnú hodnotu parametra (nezabudnite na úvodzovky);
  - ak číslo reprezentujúce mesiac nie je z intervalu <1, 12>, vygenerujte `ValueError` so správou *Invalid month XX*, kde namiesto *XX* napíšete načítané číslo;
  - ak číslo dňa nie je platné pre daný mesiac, vygenerujte `ValueError` so správou *Invalid day for XX: YY*, kde namiesto *XX* napíšete názov mesiaca (teda January, February, March, ...) a namiesto *YY* napíšete číslo dňa, ktoré ste načítali z reťazca – pre jednoduchosť neriešte priestupné roky, pre február budú akceptované čísla od 1 po 29 (vrátane).

- implementujte metódu `get_time_passed(date)`, ktorý vráti počet dní uplynutých od premiéry filmu k dátumu `date`, ktorý dostane ako parameter (ten má rovnaký formát ako `release_date` pri volaní konštruktora a v prípade neplatnej hodnoty sa vygenerujú rovnaké hodnoty ako pri kontrole dátumu premiéry). Návratová hodnota metódy je teda jedno celé číslo – počet uplynutých dní, pre deň premiéry to bude 0.

Poznámka: Vstupné parametre pre ďalšie triedy kontrolovať nemusíte, v testoch sa vždy použijú platné hodnoty.

`Auditorium` – 1 bod

Trieda `Auditorium` reprezentuje kinosálu s dvomi vnútornými premennými:
- `capacity` – kapacita (počet miest) sály, ktorá sa nastaví v konštruktore na základe parametra;
- `screenings` – zoznam premietaní v sále, v konštruktore sa nastaví ako prázdny zoznam.

Do triedy potrebujete doplniť nasledovné metódy:
- `is_available(new_screening)` – metóda skontroluje, či sála je voľná v danom čase; vstupný parameter `new_screening` je objekt typu `Screening`. Metóda vracia `True`, ak sála je v danom čase voľná, a `False` naopak. Pri zisťovaní obsadenosti berte do úvahy všetky už pridané premietania v danej sále a skontrolujte si začiatok a koniec premietania – žiadne premietanie nemôže začať v čase, keď už prebieha iné premietanie.
- `add_screening(new_screening)` – metóda pridá do zoznamu nové premietanie, ak sa môže uskutočniť v určenom čase. Čas bude súčasťou objektu `new_screening`. Metóda vracia `True` ak premietanie bolo pridané, `False` v opačnom prípade.

`Cinema` – 1,5 bodov

Pre reprezentáciu kina sa používa trieda `Cinema`, ktorá je definovaná dvomi vnútornými premennými:

- `auditoriums` – zoznam kinosál, ktorý dostane konštruktor ako parameter;
- `screenings` – zoznam premietaní, inicializuje sa ako prázdny zoznam v konštruktore.

Do triedy potrebujete doimplementovať nasledovné metódy:

- `add_movie(movie, screening_times)` – metóda pridá niekoľko premietaní filmu `movie` v časoch, ktoré sú dané zoznamom `screening_times`. Každý čas je daný ako `tuple` s dvomi hodnotami, ktoré reprezentujú hodiny a minúty (napríklad `(20, 30)` pre 20:30). Ak sa pre premietanie filmu v danom čase nájde voľná kinosála, pridajte premietanie (nový objekt typu `Screening`) do zoznamu `Cinema.screenings`, v opačnom prípade vygenerujte `RuntimeError` s chybovou hláškou *Could not add movie XX at HH:MM*, kde namiesto *XX* napíšete názov filmu, a *HH:MM* reprezentuje čas, v ktorom ste nenašli voľnú kinosálu (hodnoty zapíšte s úvodnou nulou ak je to potrebné – teda napríklad 09:30, 09:05, 20:25, 15:30). Metóda nemá návratovú hodnotu.
- `get_movies_shown()` – metóda vráti zoznam filmov, ktoré v kine dávajú. Návratová hodnota metódy je teda zoznam jedinečných filmov (objekty typu `Movie`). Na poradí týchto filmov nezáleží.
- `get_screenings_for_movie(movie)` – metóda vráti zoznam premietaní filmu, ktorý dostane ako parameter `movie` (objekt typu `Movie`). Návratová hodnota je teda zoznam objektov typu `Screening`, na poradí nezáleží.

`Screening` – 1,5 bodov

Ďalšia trieda `Screening` slúži na reprezentáciu konkrétneho filmu v konkrétnom čase v konkrétnej sále. V konštruktore sa nastavia nasledovné hodnoty:

- `movie` – objekt typu `Movie` – film, ktorý sa premieta;
- `auditorium` – objekt typu `Auditorium` – kinosála, v ktorej premietanie bude;
- `time` – `tuple` s dvomi `integermi` – čas premietania, prvá hodnota udáva hodiny a druhá hodnota minúty;
- `tickets_sold` – `integer` – počet predaných lístkov, v konštruktore inicializovaný na 0.

V triede doplňte implementáciu metód:

- `sell_tickets(count)` – reprezentuje predaj niekoľkých lístkov, ich počet je daný parametrom `count`. Ak požadovaný počet lístkov je ešte k dispozícii (nepresiahne sa kapacita kinosály), tak sa lístky predajú, teda aktualizujete hodnotu `tickets_sold`. Metóda vracia `True`, ak lístky boli predané, v opačnom prípade vráti `False`.
- `get_occupancy()` – vráti obsadenosť premietania ako pomer predaných lístkov a kapacity kinosály. Návratová hodnota je teda jedna hodnota typu `float` medzi 0 a 1.
- `get_end_time()` – vráti čas ukončenia premietania na základe začiatku a dĺžky filmu. Návratová hodnota je `tuple` s dvomi `integermi`, ktoré udávajú hodiny a minúty. Dbajte na to, aby minúty boli z intervalu <0, 59>, hodiny nemusíte normalizovať (nikdy nebudete mať premietanie, ktoré končí po polnoci).

`Person` – 0,5 bodov

Predposledná trieda, ktorú potrebujete implementovať je trieda `Person`, ktorá reprezentuje človeka, ktorý chce navštíviť kino. Každý návštevník je popísaný hodnotami:

- `interests` – zoznam reťazcov – zoznam žánrov, ktoré daného človeka zaujímajú (najradšej by si pozrel film patriaci do svojich záujmov);
- `age` – `integer` – vek návštevníka, môže prísť iba na film, ktorý má vekový limit nižší ako jeho vek;
- `bedtime` – `integer` – reprezentuje čas, kedy človek už chce byť doma, resp. premietanie musí končiť skôr, ako tento čas – určený iba ako hodina;
- `tolerance` – `float` – hodnota medzi 0 a 1, vyjadruje to, že pri akej preplnenosti kinosály je človek ešte ochotný kúpiť si lístok. Napríklad ak človek toleruje preplnenosť do 80% percent, ale 90% lístkov je už predaných, tak nerád si vyberie dané premietanie.

V triede potrebujete implementovať metódy:

- `is_interested(movie)` – metóda vracia `True` alebo `False` v závislosti od toho, či žáner filmu, ktorý dostane ako parameter (objekt typu `Movie`) patrí medzi záujmy človeka.
- `is_allowed(movie)` – metóda vracia `True` alebo `False` v závislosti od toho, či človek je už dostatočne starý na návštevu filmu, ktorý dostane ako parameter (objekt typu `Movie`).
- `can_attend(screening)` – metóda vracia `True` alebo `False` v závislosti od toho, či človek je už dostatočne starý na návštevu filmu, ktorý sa premieta v rámci `screening` (objekt typu `Screening`) a zároveň premietanie končí ešte pred jeho `bedtime`-om (skontrolujte iba hodiny). Ak napríklad premietanie končí o 23:00 alebo neskôr a on má `bedtime` 23, metóda vracia `False`.
- `will_attend(screening)` – metóda vracia `True` alebo `False` v závislosti od toho, či preplnenosť premietania `screening` (objekt typu `Screening`) je ešte tolerovateľná človekom (teda nedosiahla hodnotu tolerancie preplnenosti).

`FriendGroup` – 2 body

Posledným komponentom nášho riešenia je trieda `FriendGroup`, ktorá predstavuje skupinu priateľov, ktorí chcú ísť spoločne do kina. Trieda je definovaná zoznamom ľudí (objekty typu `Person`), ktorí do nej patria (členská premenná `members`, nastaví sa priamo v konštruktore). Triedu využívame na výber premietania, ktoré naša skupinka navštívi a to pomocou nasledovných metód:

- `order_movies(cinema)` – metóda dostane ako parameter objekt typu `Cinema`, teda kino, do ktorého skupina pôjde. Následne z tohto kina potrebujete získať zoznam filmov, ktoré sa práve premietajú (zatiaľ neriešite konkrétne premietania). Z tohto zoznamu vyberiete tie filmy, ktoré si môžu pozrieť všetci členovia skupiny – do úvahy sa berie to, či sú dostatočne starí vzhľadom na vekový limit filmu. Potenciálne filmy zoradíte podľa toho, koľkí zo skupiny sa zaujímajú o žáner daného filmu. Návratová hodnota bude zoznam filmov zoradených podľa tohto ukazovateľa, teda bude to zoznam `tuple`-ov, kde každý `tuple` obsahuje najprv film a následne počet ľudí zo skupiny, ktorí sa o žáner zaujímajú. Zoznam bude zoradený zostupne, teda na prvej pozícii bude dvojica hodnôt s filmom, ktorý si chce pozrieť najviac ľudí zo skupiny a nebude obsahovať filmy, ktoré niekto zo skupiny nemôže navštíviť kvôli svojmu veku.

- `choose_screening(cinema)` – metóda dostane ako parameter objekt typu `Cinema`, teda kino, do ktorého skupina pôjde. Následne z tohto kina sa vyberie konkrétne premietanie hociktorého filmu tak, aby s výberom bolo spokojných čo najviac ľudí. Premietania zoraďte a vyberte podľa nasledovných kritérií:

  1. primárne sa vyberie premietanie, ktoré môže navštíviť čo najviac ľudí vzhľadom na vekový limit a čas ukončenia premietania (ak premietanie končí po `bedtime` človeka, ten najradšej by tam neprišiel);
  2. ak viacero premietaní má prvý počet ľudí rovnaký, vyberie sa premietanie, ktoré by navštívilo najviac ľudí vzhľadom na preplnenosť kinosály – tu do úvahy berte iba počet ľudí, ktorí by reálne chceli prísť vzhľadom na prvý bod (ak teda niekto nemôže prísť na premietanie, pretože preňho je to príliš neskoro, jeho nezarátajte aj keď preplnenosť sály by mu nevadila);
  3. ak stále máte rovnaký počet ľudí ochotných prísť na viacero premietaní, zoberte premietanie s filmom, ktorý je novší, teda od jeho premiéry uplynulo menej dní.

  Funkcia vracia dve hodnoty – prvá hodnota je objekt typu `Screening`, teda premietanie, ktoré sa vybralo ako najvýhodnejšie pre skupinu. Druhá hodnota je zoznam všetkých premietaní, ktoré prišli do úvahy, pričom každý prvok bude `tuple` s hodnotami: premietanie (objekt typu `screening`), a hodnota troch ukazovateľov uvedených vyššie (všetky `integer` hodnoty). Zoznam bude zoradený zostupne vzhľadom na tri ukazovatele s uvedenými preferenciami (pre tretí parameter vzostupne).

- `buy_tickets(sreening)` – metóda reprezentuje kúpu lístkov pre ľudí zo skupiny, ktorí na premietanie (parameter `screening`) prídu – ich `bedtime` je po ukončení premietania. Záujmy a tolerancie ľudí sa tu neberú do úvahy, ako poriadny priateľ, každý príde aj keď ho žáner nezaujíma alebo sála je preplnená. V metóde zavolajte príslušnú metódu z triedy `Screening` pre aktualizáciu počtu predaných lístkov. Metóda nemá žiadnu návratovú hodnou.

Do tried môžete pridať ďalšie vnútorné premenné, musíte však tam nechať tie, ktoré sú dané zadaním (inak testy neprejdú a dostanete 0 bodov za danú triedu).

Za implementáciu tried môžete získať maximálne 9 bodov. **Vzhľadom na to, že zadanie bolo zverejnené s oneskorením, ak z 9 bodov získate minimálne 3, jeden bod sa pripočíta automaticky.**