Proiect - Etapa 1 - Santa Claus is coming to ACS students

- **Data publicarii:** 29.11.2021 00:00
- **Data ultimei modificari:** 28.11.2021 20:30
- ~~**Deadline soft:** 15.12.2021 23:55~~
- **Deadline hard:** ~~17.12.2021 23:55~~ 9.01.2022 23:55
- **Responsabili:** Ionela Nicuță [mailto:ionela.nicuta05@gmail.com], Hermina-Maria Matei [mailto:mateiherminamaria@gmail.com], Maria Leoveanu [mailto:maria.alexandra.leoveanu@gmail.com], Mihail Ungureanu [mailto:mihai2000.ungureanu@gmail.com], Narcis-Florin Căroi [mailto:narciscaroi24@gmail.com], Andreea Oltean [mailto:deea.oltean2@gmail.com], Ioana-Alina Tudorache [mailto:alina.tudorache872@gmail.com], Miruna Banu [mailto:mirunaelena.banu@gmail.com], Radu Pavel [mailto:pavel.radu02@gmail.com], Alexandra-Andreea Imbrișcă [mailto:alexandra.imbrisca6@gmail.com], Ștefan-Alin Pahonțu [mailto:stafy2912@gmail.com], Lucian-Florin Grigore [mailto:luci.grigore99@gmail.com], Andreea Tătulescu [mailto:andreea.tatulescu2610@gmail.com], Yasemin Menan [mailto:yasemin.menan@gmail.com]
- **Actualizări**:
  - 03.12.2021, 11:45 - little fix în checker (modificare refs) [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]
  - 05.12.2021, 20:00 - update la ref-ul 25 [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]
  - 06.12.2021, 18:33 - modificare la testele 18, 19, 24 si la ref-urile corespunzatoare [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]
  - 06.12.2021, 18:50 - adaugarea tuturor oraselor care se regasesc in teste in Cities Enum [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]
  - 08.12.2021, 11:55 - update la ref-ul 25 si enunt - calculare suma averageNiceScore pentru bugetUnit in ordinea id-urilor [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]

Obiective

- dezvoltarea unor abilități de bază de organizare și design orientat-obiect
- scrierea unui cod cât mai generic, ce va permite ulterior adăugarea de noi funcționalități
- folosirea unor design patterns
- respectarea unui stil de codare și de comentare

Scenariu

Moșul știe că ți-ai dat silința întreg semestrul și vrea să te răsplătească pentru toate laboratoarele trimise la timp, toate quiz-urile de curs luate cu 10/10 și, bineînțeles, pentru că în loc să te bucuri cu familia de sărbători, stai și faci Proiectul la POO. Dacă nu te-ai regăsit în descrierea de

mai sus, nu-ți face griji, Moșul are câte ceva pentru fiecare, depinde doar de cât de cuminte ai fost.

De anul acesta, Moșul își dorește să intre în concediu și să se ducă în Hawaii, întrucât a obosit după atâția ani de muncă. Deoarece din ce în ce mai puțini elfi vin la Atelierul de Jucării, Moșul are nevoie de o mână de ajutor. Pentru a păstra intactă magia sărbătorilor, se gândea să îți dea acces la baza de date secretă de la Polul Nord - Lista Copiilor Cuminți și Obraznici.

Salvează Crăciunul și ajută-l pe Moșul să pregătească cadourile pentru copiii din întreaga țară și să ajungă la timp la fiecare dintre ei!

Context

Moș Crăciun dorește să-și ia concediu pentru mai mulți ani, așa că tu va trebui să determini ce cadouri va primi fiecare copil în anii în care Moșul este plecat. Numărul de ani este specificat în formatul de input în câmpul **numberOfYears**.

- Simularea va fi implementată pentru datele inițiale - **initialData** (runda 0) și pentru următorii **numberOfYears** ani.
- În total, se vor simula **numberOfYears + 1** runde, dacă includem și runda 0.
- Pentru runda 0 se oferă datele inițiale - **initialData**, iar pentru următorii **numberOfYears** ani se oferă diversele actualizări.

Format JSON input

- **numberOfYears** - numărul de ani pe care se implementează simularea, excluzând runda 0.
- **santaBudget** - bugetul inițial al Moșului (folosit la runda 0).
- **initialData** - datele inițiale, folosite pentru simularea rundei 0 și care vor fi modificate în funcție de actualizările din fiecare an.
- Datele inițiale conțin:
  - o listă inițială de copii
  - o listă cu cadourile pe care Moșul le are la Polul Nord
  - o listă de orașe pe care le vizitează Moșul
- Informații relevante:
  - Copil:
    - id (câmp unic)
    - last name
    - first name
    - age
    - city
    - niceScore - scorul de cumințenie inițial (reprezentat ca Double)

        - giftsPreference - o listă cu preferințe de cadouri (**ATENȚIE**: Preferințele de cadouri reprezintă categoriile din care își dorește un cadou)
    - Cadou:
        - productName (numele cadoului)
        - price - prețul cadoului (reprezentat ca Double)
        - category - categoria din care face parte cadoul (de tip Enum - definită în pachetul enums)

Pentru câmpurile *city* și *category* au fost definite enum-uri pe care le puteți găsi în schelet, în pachetul enums.

- **annualChanges** - lista care conține informațiile despre schimbările care se produc în fiecare an (lungimea listei este **numberOfYears**)

O schimbare anuală conține:

- **newSantaBudget** - noul buget al Moșului pentru anul respectiv
- **newGifts** - o listă cu cadouri noi
- **newChildren** - o listă cu copii noi
- **childrenUpdates** - o listă cu date actualizate pentru o parte din copii

Noul buget al Moșului este tot de tip Double, noile cadouri și noii copii păstrează formatul inițial.

- **ChildUpdate:**
    - **id** - pentru a detecta copil căruiă i se modifică informațiile
    - **niceScore** - un nou scor de cumințenie, care va fi de tip Double
    - **giftsPreferences** - noi preferințe de cadouri, care respectă formatul inițial al unui cadou

Flow-ul simulării

Fiecare copil se încadrează la o anumită categorie în funcție de vârstă:

- Baby: < 5y
- Kid: 5 - 12y
- Teen: 12 - 18y
- Young Adult: > 18y

Aceste categorii de vârsta vor fi folosite pentru determinarea average score-lui de cumințenie, care este definit mai jos.

Runda 0:

- Se determină categoria de vârstă pentru fiecare copil

- Se creează o listă care reține scorul de cumințenie pentru fiecare copil. (În această rundă, lista de scoruri de cumințenie va avea un singur element, scorul de cumințenie inițial; **ATENȚIE**: niceScore-ul inițial **NU** poate fi null)
- Se calculează average score-ul (Double) pentru fiecare copil în funcție de categoria de vârstă:
  - Baby: (< 5y) - averageScore = 10
  - Kid: (5 - 12y) - media aritmetică a scorurilor din lista de scoruri, care pentru runda 0, va fi chiar scorul inițial
  - Teen: (12 - 18y) - media ponderată a scorurilor din lista de scoruri, care pentru runda 0 va fi chiar scorul inițial
  - Young Adult: (> 18y) - nu se va calcula o listă de cumințenie și nu se va mai reține un average score, întrucât copiii care sunt de tip Young Adult nu vor mai primi cadouri și nu se vor include în output
- Se calculează bugetul pe care îl alocă Moșul pentru fiecare copil:

```
budgetUnit = santaBudget / suma scorurilor average de la toți copiii
bugetul alocat unui copil = averageScore * budgetUnit
```

Calculele se efectuează folosind variabile de tip Double!

Suma scorurilor average de la toți copiii se va calcula în ordinea id-urilor copiilor pentru a nu aparea diferențe de zecimale. (La început se va adăuga la suma average score-ul copilului cu id-ul 1, apoi average score-ul pt copilul cu id-ul 2, etc.)

A NU se folosi **DoubleStream.sum()** deoarece nu garantează ordinea corectă a calculării sumei!

- Se distribuie cadourile pentru fiecare copil:
  - Se consideră că lista de preferințe a unui copil este ordonată de la categoria cea mai dorită, la categoria cea mai puțin dorită.
  - Moșul încearcă să ofere câte un cadou din fiecare categorie din listă copilului, în ordinea în care apar categoriile în listă.
  - Moșul poate cumpăra cadouri unui copil, atâta timp cât încă are bani pentru copilul respectiv (nu se depășește bugetul alocat pentru copil).
  - Moșul parcurge lista de preferințe a copilului și verifică dacă în lista sa de cadouri există un cadou din categoria respectivă. În cazul în care cadoul există, Moșul va asigna acel cadou copilului dacă prețul cadoului se încadrează în buget. Dacă se găsesc mai multe cadouri din aceeași categorie, Moșul decide să ofere copilului cadoul cu prețul cel mai mic.

Se garantează ca 2 cadouri cu același preț nu pot face parte din aceeași categorie. (Astfel, se poate considera că fiecare cadou este unic).

Rundele care se desfășoară pe cei numberOfYears ani:

- Pentru fiecare copil care a rămas în lista Moșului (nu a devenit Young Adult), se incrementeaza vârsta cu 1. Copiii care devin Young Adult, vor fi șterși din lista Moșului, întrucât Moșul nu le va asigna cadouri.
- Moșul citește lista de update-uri pentru fiecare an. (pentru anul i, se va folosi al i-lea element din lista **annualChanges**)
  - Inițial, Moșul adaugă noii copii în lista de copii, decât dacă noii copii nu sunt de tip Young Adult (age > 18).
  - Se actualizează datele despre copiii existenți, folosind lista **childrenUpdates** în felul următor:
    - se verifică dacă id-ul copilului pentru care avem un update se mai găsește în lista Moșului (dacă acest copil a devenit Young Adult între timp, el nu se va mai găsi în listă).

- se adaugă noul niceScore la lista de nice scoruri doar dacă niceScore-ul primit la updates **NU** este null (dacă este null, se consideră că nu a fost adăugată nicio informație despre niceScore).
- se adaugă noile preferințe ale copilului la lista de preferințe existente : **ATENȚIE**: noile preferințe se vor adăuga la începutul listei de preferințe existente
- Exemple:
  - **Exemplul 1**:
    - Dacă Moșul are in lista un copil cu preferințele:
      [“BoardGames”, “Toys”] și în cadrul unui update apare pentru același copil noile preferințe [“Clothes”, “Technology”], atunci lista de preferințe pentru runda curentă va fi [“Clothes”, “Technology”, “BoardGames”, “Toys”]
  - **Exemplul 2**:
    - Dacă în cadrul unui update se adaugă la lista de preferințe, categorii deja existente, se va șterge vechea apariție a acelei categorii:
      Lista existentă: [“BoardGames”, “Toys”]
      Update: [“Clothes”, “Toys”]
      Lista pentru runda curentă: [“Clothes”, “Toys”, “BoardGames”]

Dacă lista de categorii noi este goală, înseamnă că se păstrează lista deja existentă.

- Se adaugă noile cadouri la cadourile deja existente în lista Moșului
- Se actualizează bugetul Moșului (se înlocuiește bugetul vechi cu bugetul nou)
- Se determină categoria de vârstă pentru fiecare copil (copiii care sunt Young Adults vor fi șterși din lista de copii, întrucât Moșul nu le va asigna cadouri)
- Se calculează average score-ul folosind următoarele formule:
- Baby: < 5y - averageScore = 10
- Kid: 5-12y - media aritmetică a scorurilor din lista de scoruri
  Exemplu :
  lista = [7, 8, 9]
  averageScore = (7 + 8 + 9) / 3
- Teen: 12-18y - media ponderată a scorurilor din lista de scoruri
  Fiecare element din lista va avea asociată o pondere:
  pentru elementul de pe poziția i, ponderea va fi (i + 1), întrucât ultimele niceScore-uri sunt mai relevante
  formula: suma(a_i * (i + 1)) / (1 + 2 + … + n)
  Exemplu :
  lista = [7, 8, 9]
  averageScore = (7 * 1 + 8 * 2 + 9 * 3) / (1 + 2 + 3)

- Young Adult: >18y - nu se va calcula o listă de cumințenie și nu se va mai reține un average score, întrucât copiii care sunt de tip Young Adult nu vor primi cadouri și nu se vor include în output
- Se calculează unitatea de buget pentru fiecare copil și se asignează un buget fiecărui copil exact ca la runda 0:
  budgetUnit = santaBudget / suma scorurilor average de la toți copiii
  bugetul alocat unui copil = averageScore * budgetUnit
- Se distribuie cadou pentru fiecare copil ca la runda 0:
- Se consideră că lista de preferințe a unui copil este ordonată de la categoria cea mai dorită, la categoria cea mai puțin dorită.
- Moșul încearcă să ofere câte un cadou din fiecare categorie din lista copilului, în ordinea în care apar categoriile în listă.
- Moșul poate cumpăra cadouri unui copil, atâta timp cât încă are bani pentru copilul respectiv (nu se depășește bugetul alocat pentru copil).
- Moșul parcurge lista de preferințe a copilului și verifică dacă în lista sa de cadouri există un cadou din categoria respectivă. În cazul în care codul există, Moșul va asigna acel cadou copilului dacă prețul cadoului se încadrează în buget. Dacă se găsesc mai multe cadouri din aceeași categorie, Moșul decide să ofere copilului cadoul cu prețul cel mai mic.

Se garantează că 2 cadouri cu același preț nu pot face parte din aceeași categorie. (Astfel, se poate considera ca fiecare cadou este unic).

Format Output

Se va afișa o listă de lungime numberOfYears + 1, care pe poziția i va conține o listă cu copiii care au primit cadouri la runda i. Fiecare copil va avea urmatoarele informații:

- id (câmp unic)
- last name
- first name
- city
- age
- giftsPreference - o listă cu preferințe de cadouri (**ATENȚIE**: Preferințele de cadouri reprezintă categoriile din care își dorește un cadou, acestea vor fi afișate sub formă de **enum**)
- averageScore - scorul average calculat la runda i
- niceScoreHistory - lista care conține scorurile de cumințenie
- assignedBudget - bugetul alocat la runda i
- receivedGifts - lista cu cadourile primite la runda i (cadourile trebuie să apară în ordinea în care Moșul le-a dat acestui copil, de la categoria cea mai dorită, la categoria cea mai puțin dorită)

Lista de copii de la fiecare rundă este sortată crescător în funcție de id-ul copilului.

Copiii care au devenit Young Adults nu vor fi incluși în outputul de la runda i.

Important

- Integers: id, age
- Double: everything else (budget, price)
- **TREBUIE** utilizate design patterns (dintre cele studiate la laborator) pentru versionarea codului.

Indicații

- Separați conceptele de sine stătătoare în clase separate, nu le îmbinați - clasele ar trebui să aibă un sigur rol.
- Adaptați agregarea și moștenirea la situație, grupați pe cât posibil informația și acțiunile comune în clase generale.
- Nu vă apucați să scrieți direct; alocați timp modelării și abstractizării, pentru că altfel vă puteți trezi cu o temă muncitorească, cu mult cod din care să nu înțelegeți prea multe și pe care să îl extindeți greu.
- Acesta este prima etapă a proiectului, ceea ce presupune că va există și o a doua etapă; vă recomandăm să încercați să scrieți un cod cât mai generic, care să permită adăugarea ulterioară de noi funcționalități; cu toate acestea, etapa a doua se poate trimite fără să fi trimis prima etapă a proiectului, însă va fi nevoie ca în cadrul celei de-a doua etape să se implementeze și funcționalitățile primei etape pentru a putea primi punctajul total pe testele celei de-a doua părți.

Testarea soluției

Pentru testarea soluției, rulați funcția **main** a clasei **Test**. Aceasta va rula atât testele, cât și checkstyle-ul. Pentru rularea checkerului, aveți nevoie ca proiectul vostru să aibă încărcate bibliotecile pentru citirea fișierelor JSON. Mai multe detalii aici.

Evaluare

Punctajul constă din:

- 60p implementare - trecerea testelor
- 10p coding style (vezi checkstyle)
- 15p design și organizare (folosire design patterns)
- 10p README clar, concis, explicații axate pe design (flow, interacțiuni)
- 5p utilizare Git (minimum 3 commit-uri personale, relevante pentru flow-ul proiectului)

- Folosirea git pentru versionare va fi verificată din folderul .git pe care trebuie să îl includeți în arhiva temei.
- Punctajul se va acorda dacă ați făcut minim 3 commit-uri relevante și cu mesaj sugestiv.
- **NU** este permis să aveți repository-urile de git publice până la deadline-ul hard.

Pe pagina Indicații pentru teme găsiți indicații despre scrierea README-ului și depunctările generale pentru teme

Depunctările pentru **designul și organizarea codului** se vor scădea din punctajul testelor. Dacă vor apărea depunctări specifice temei în momentul evaluării, nemenționate pe pagina cu depunctări generale, ele se vor încadra în limitele de maxim 15 pentru design, 10p pentru README. Dacă tema nu respectă cerințele, sau nu are design OOP, atunci pot apărea depunctări suplimentare.

Bonusuri: La evaluare, putem oferi bonusuri pentru design foarte bun, cod bine documentat, dar și pentru diverse elemente suplimentare alese de voi.

Temele vor fi testate împotriva plagiatului. Orice tentativă de copiere va duce la **anularea punctajului** de pe parcursul semestrului şi **repetarea materiei** atât pentru sursă(e) cât şi pentru destinație(ii), fără excepție.

Checkstyle

Unul din obiectivele temei este învățarea respectării code-style-ului limbajului pe care îl folosiți. Aceste convenții (de exemplu cum numiți fișierele, clasele, variabilele, cum indentați) sunt verificate pentru temă de către tool-ul checkstyle [https://checkstyle.sourceforge.io/].

Pe pagina de Recomandări cod găsiți câteva exemple de coding style.

Dacă numărul de erori depistate de checkstyle depășește 30, atunci punctele pentru coding-style nu vor fi acordate. Dacă punctajul este negativ, *acesta se trunchiază la 0*.

Exemple:

- `punctaj_total = 125` și `nr_erori = 200` ⇒ `nota_finala = 115`
- `punctaj_total = 125` și `nr_erori = 29` ⇒ `nota_finala = 125`
- `punctaj_total = 80` și `nr_erori = 30` ⇒ `nota_finala = 80`
- `punctaj_total = 80` și `nr_erori = 31` ⇒ `nota_finala = 70`

Upload temă

Arhiva pe care o veţi urca pe VMChecker [https://vmchecker.cs.pub.ro/ui/#POO] va trebui să conţină în directorul rădăcină:

- fișierul `README`
- folder-ul src cu pachetele și cu fișierele .java

Resurse și linkuri utile

- Schelet de cod [https://github.com/oop-pub/oop-asignments/tree/master/proiect1]
- Indicații pentru teme

- Recomandări coding style & javadoc

poo-ca-cd/teme/proiect/etapa1.txt · Last modified: 2022/01/06 12:16 by ioana.tudorache2507