Eremia Evelina-Andreea, 321 CC, tema a avut un grad mediu de dificultate si am rezolvat primele 2 task-uri in 2 saptamani.

Account + Information:

-In clasa Account am creat clasa interna Information, folosind sablonul Builder.
Clasa Account contine constructorul, o metoda toString si o metoda getRandomCharacter.

Cell + CellElement

Clasa Cell contine 2 constructori. Visited se seteaza false, adica este initial nevizitata casuta. Clasa are si metode de tip get pentu CellElement si type si o metoda ce converteste tipul casutei din String in Type. CellElement e o interfata ce contine metoda toCharacter() ce intoarce caracterul de afisat.

Character + CharactersFactory

Clasa abstracta Character extinde clasa abstracta Entity. Character are 2 constructori.
Am folosit formulele strength += level; charisma += level / 2 ;dexterity += level / 3; pentru a calcula atributele.Inventory va avea maxWeigth 100. For-ul de la 1 la 3 alege in mod random tipul de spell ce va avea damage-ul si mana 30 initial.

Metoda de cumparare a potiunii , buyPotion, verifica daca banii si greutatea sunt in regula pentu a fi adaugata. Se intoare un raspuns de tip boolean. In cazul favorabil, se adauga potiunea in lista de potiuni din inventory.

Metoda void interact(Shop shop) trateaza si exceptia pentru care comanda este invalida. Utilizatorul alege indexul potiunii sau "x" in cazul in care vrea sa iasa din shop. Se selecteaza potiunea de la indexul dorit si se cumpara.

In metoda boolean interact(Enemy enemy) se verifica daca e randul utilizatorului. Daca da, apasa "a" daca doreste sa atace sau "p" daca vrea sa foloseasca o potiune.

Daca e selectat "a" si nu exista spells, se calculeaza damage-ul si se foloseste pe enemy.

Daca e selectat "a" si exista spells, se alege "n" pentru atac normal sau indexul spell-ului ce se va folosi. Pentru "n" se procedeaza ca inainte, altfel se foloseste spell sau se arunca exceptia InvalidCommandException() daca indexul nu e corespunzator. Daca e selectat "p", se foloseste potiunea si se sterge din inventory.

Daca e randul inamicului,se calculeaza damage-ul, se primeste si se verifica daca health<0, acesta fiind cazul in care utilizatorul pierde.

Variabila a contorizeaza al cui este randul. Daca currenthealth-ul enemy-ului este<0 jocul intoarce false. Clasa contine si o metoda toString().

CharactersFactory foloseste sablonul Factory pentru a instantia personajele din lista contului, acestea fiind Warrior, Mage, Rogue.

Clasa **Credentials** contine emailul si parola, constructorul si metode de tip set, get.

Spell+Ice+Fire+Earth

Spell e o clasa abstracta ce contine damage, mana si include constructorul.
Clasele Ice, Fire, Earth extind clasa abstracta Spell. Metoda visit a acestora verifica daca entity-ul este imun la respectiv-ul spell sau nu. Daca nu, acesta primeste damage-ul calculat.

Element+Visitor

Aceste interfate respecta sablonul Visitor si sunt implementate in clasele Entity, respectiv Spell.

Entity

Clasa abstracta Entity, pe langa constructori, contine metodele de regenerare pe care le-am implementat folosind functia Math.min pentru a ma asigura ca nu se depaseste maximul de viata sau mana. De asemenea, clasa contine si metoda useSpell() care foloseste abilitatea impotriva inamicului

daca exista destula mana, dar si metodele abstracte receiveDamage(final float damage) si getDamage();

Enemy

Aceasta clasa extinde Entity si implementeaza interfata CellElement.

In constructor se creeaza o noua lista, spells. No reprezinta numarul de abilitati, el fiind ales random. For-ul adauga in spells fiecare abilitate in functie de tipul acesteia. Urmatoarele 3 if-uri stabilesc aleatoriu abilitatile enemy-ului.

Metoda toCharacter returneaza 'E' in acest caz.

Metoda recieveDamage calculeaza damage-ul. Se stabileste randomly(cu o sansa 50%) daca inamicul a fost sau nu evitat si se retine in variabila type. Daca nu, se scade din currHealth damage-ul, currHealth ramandand 0 daca scaderea da un rezultat <0.

Pentru metoda getDamage, type-ul este ales la intamplare cu sansa de 50%. Am ales damage-ul 12,5 pentru type==0, si dublu in rest.
Clasa contine si o metoda toString si metoda accept(Visitor visitor).

Grid

Metoda generateMap creaza o lista de liste. In primul for, pentru fiecare element din lista initiala, se creeaza o noua lista numita line, in care se adauga Cell-uri cu cel de-al 2-lea for. Am setat 3 Shop-uri, Enemy si Finish pe harta. Metoda returneaza map-ul.

Metodele goNorth(), goSouth(), goWest(), goEast() adauga/scad 1 pentru x-ul sau y-ul curent, in functie de directia si sensul deplasarii, si afiseaza "Can't go there!" in cazul in care deplasare nu poate fi efectuata. Clasa contine si o metoda toString. Cu 2 for-uri se parcurge harta si afiseaza simbolul corespunzator.

HealthPotion + ManaPotion + Potion

Aceste clase implementeaza interfata Potion. Contin contructorii necesari, metodele getPrice, getWeigth, metoda use(Entity entity), care regenereaza mana/health-ul si o metoda toString.

Inventory

Constructorul primeste maxWeight-ul. Se creaza o lista de potiuni. Coins se seteaza 100. addPotion adauaga potiunea in potions si scade pretul din coins. removePotions sterge potiunea. getRemainWeight foloseste un for si aduna greutatea tuturor potiunilor in currWeight pe care o scade din maxWeight.

Warrior-Mage-Rogue

Aceste clase extind Character. In constructorul fiecareia se seteaza artributul principal cu 4 si cu true la ce este imun fiecare (ice/earth/fire). Fiecare are implementata alta metoda getDamage, iar recieveDamage calculeaza tinand cont de atributele secundare ale fiecaruia.

Shop

Contine o metoda pentru selectarea potinunii si eleiminarea acesteia.

Game

Pentru a crea clasa Game am folosit sablonul Singletone

In metoda run(), accounts este lista cu conturi. Se printeaza lista de conturi, iar utilizatorul alege indexul unuia. Asemenator se alege si un character. Se genereaza harta care este 5x5. Am adaugat miscarile in moves. In while se verifica daca s-a ajuns pe casuta finish.
Daca nu, jucatorul va trebui sa apese p pentru a face urmatoarea miscare. Se verifica aceasta si se apeleaza metoda respectiva.Metoda showCellOptions(Grid<Cell> grid) verifica daca player-ul se afla pe o casuta de tip shop sau enemy si apeleaza metodele interact(shop) sau interact(enemy) in functie de tip.

Test

Aceasta clasa contine metodele parseAccounts() si parseStories().
ParseAccounts() returneaza o lista cu conturi. In primul for s-au citit initial credentials, characters si favorite_games.De asemenea, pentru fiecare character s-au citit in al 2-lea for numele, profesia, nivelul si experienta si s-a adaugat in lista gameCharacters character-ul respectiv. In favoriteGames se adauga jocurile favorite si se sorteaza alfabetic. Se citesc in continuare email-ul, parola, numele, tara si numarul de jocuri, ca apoi cu acestea sa se construiasca Information cu ajutorul sablonului Builder. In final, se adauga un nou Account in lista gameAccounts.
In ParseStories() se ia fiecare obiect din json-ul cu stories, fiecare element contine 2 stringuri si se iau tipul si valoarea.

In aceasta metoda se arunca exceptiile IOException, InformationIncompleteException.

Clasa contine si metoda main in care se instantiaza un game si se apeleaza run.