Pacman

- Specificatie Proiect -

Voicu Diana – Georgiana
Grupa : 30223

1. Ce am implementat mai exact ?

   In urma documentarii, citrii tuturor surselor propuse, dar si a celor mai diverse surse online, am decis sa implementez o versiune "Pacman" neconventionala, destul de originala, numita de mine: ***Pacman – Treasure Hunt Edition*** . Plecand cu acest gand, am alcatuit diverse clase, diverse implementari, insa am ales sa concluzionez cu cele care imi inspirau cea mai multa logica, si sper de asemenea, ca sunt cele mai optime.

2. Cum sunt impartite clasele?

   Clasele sunt impartite dupa cum urmeaza:

- ✓ **Game Controller** - Partea de CONTROL a structurii MVC, clasa de baza pentru creearea "Frame" si pentru realizarea paginii de titlu, contine **metoda main** si ne ofera rularea intregului program. Contine un **constructor** care implementeaza **"frame"-ul pentru titlu**, prezinta un **buton**, a carei apasare determina deschiderea **"frame"-ului pentru joc**, avand metoda **createGameScreen()**.

- ✓ **Player** – Partea de MODEL, clasa ce il implementeaza pe "pacman" , continand atat un **constructor** cu toate **imaginile necesare mutarii lui: right, left, up, down** cat si pozitiile din matrice x si y (tiles), **getters si setters** pentru acestea, **metoda move**, care modifica pozitiile in functie de directia aleasa, si **4 metode : right(), left(), up(), down()**, care seteaza imaginea potrivita pentru pacman.

- ✓ **Ghost** – Partea de MODEL, clasa ce implementeaza **"fantomele"**, care sunt de fapt inamicii lui pacman. Aceasta contine **matricea pentru map/maze** pentru orientarea lor, **un constructor cu imaginiile necesare** fantomei: right, left, up, down, **coordonatele x,y** , precum si **pozitiile din matrice ale acestor coordinate (tiles)**, **directiile de mutare sunt codificate cu 0,1,2,3, getters si setters** pentru toate acestea. Importanta aici este **metoda canMove()** , aceasta verifica daca pozitia pe care dorim sa mutam ghost-ul nu este perete, iar atunci aceasta va returna true, altfel va returna false. Aceasta metoda ne ajuta la impementarea urmatoarei metode**: randomChanging()**, metoda care ia directia generata random si o verifica cu fiecare dintre codificarile noastre. Daca directia generata random corespunde unei directii codificate, se verifica daca fantoma se poate muta, apeland canMove(), iar daca aceasta returneaza true, fantoma se va misca , pozitia din matrice modificandu-se, iar daca nu, se va genera o alta directie random, pana se ajunge la o miscare valida.

- ✓ **Maze** – partea de VIEW, dar si o oarecare parte de CONTROL a structurii MVC, de departea, cea mai complexa clasa, aceasta imbina majoritatea claselor create si controleaza sirul evenimentelor. Acesta contine o **matrice de int-uri** , prezenta si in clasa Ghost, care ne va ajuta la generarea mapei, contine un **constructor** unde se adauga KeyListener si se **apleaza metoda initialize()** –

aceasta metoda intializeaza timer-ul, adauga toata imaginile necesare noua, il creeaza pe pacman, impreuna cu fantomele si creeaza o copie a matricei de care vom avea nevoie mai tarziu. Urmeaza **metoda paint()** – care parcurge matricea si deseneaza intregul maze, deseneaza fantomele, il deseneaza pe pacman, scorul si numarul de keys ramase. Determina si apelarea metodei de endGame() in caz ca pacman moare, si metoda grabTreasure() in caz ca toate cheiile au fost colectionate. **Metoda endGame()** deseneaza un ecran negru in cazul in care jocul se termina, cu un mesaj: game Over si posibilitatea de a apasa tasta Space pentru a reincepe jocul. Matricea initiala va fi umpluta cu valorile din matricea fixed, pentru a se putea reseta jocul initial.
Metoda eatFood() verifica daca coordonatele matricei pe care se afla pacman, coincide cu coordonatele pe care se afla mancarea, iar daca da, scorul se va modifica in functie de tipul mancarii si bucata din matrice va fi inlocuita cu un drum normal. **Metoda grabKey()** procedeaza la fel precum cea anterioara, insa aici verificarea se va face cu coordonatele pe care se afla keys, determinant modificarea numarului de keys daca pacman obtine cheia. **Metoda grabTreasure()** procedeaza la fel precum cele anterioare, insa intai este verificata conditia ca toate cheile sa fie obtinute de pacman, numai atunci acesta va putea obtine si comoara si va determina castigarea jocului. Astfel se determina un ecran de WIN, ce va contine o imagine, scorul curent, highest score, precum si un buton pentru reinceperea jocului**. Metoda highest()** determina cel mai mare scor provenit din toate jocurile, si il tine minte pe cel mai mare, **metoda updateAll()**, actualizeaza scorul, highest score si numarul de keys dupa fiecare mutare, **metoda randomMovement()**, apeleaza metoda randomChanging() din clasa Ghost, pentru fiecare fantoma in parte permanent, iar **metoda pacmanLife()** verifica daca pozitiile x,y ale matricei la care se afla pacman, coincid cu pozitiile x,y ale matricei la care se afla fantomele, iar daca acest criteriu este indeplinit, inseamna ca pacman va fi omorat, altfel acesta traieste. Daca pacman moare, tot jocul se va opri, astfel dand comanda t.stop(). **Metoda actionPerformed()**, este extinsa din clasa ActionListener, avand grija ca dupa fiecare actiune sa apeleze toate metodele necesare si sa redeseneze tot. **Clasa MyAL()** declarata inauntru reprezinta o clasa speciala ce extinde **KeyAdapter**, determinand metodele de override : **keyPressed(), keyRealeased(), keyTyped()**, aici am dezvoltat doar **metoda keyPressed()**, determinant care dintre taste au fost apasate si dand functii fiecarora. Astfel, daca se va apasa tasta right, se va deplasa spre dreapta, daca se apasa stanga spre stanga, si asa mai departe, iar daca se apasa Space, se va reseta jocul. Aici am aplicat si conditia ca daca pacman iese cumva din frame, acesta sa revina in joc pe partea opusa, prin tunel. Aceasta clasa contine de asemenea setters si getters pentru variabile.

3. Regulile jocului

Pacman este intitalizat in mijlocul maze-ului, zona protejata unde fantomele nu pot patrunde. Acestea sunt intializate in cele 4 regiuni ale maze-ului, iar fiecare incepe sa se miste random pe toata suprafata matricei in afara de zona restransa.

Regulile jocului implementat sunt dupa cum urmeaza:

- Pacman trebuie sa colectioneze mancarea pentru a primi puncte. Pentru fiecare mar, acesta primeste 5 puncte, iar pentru fiecare portocala, acesta primeste 10 puncte (the Jackpot).
- Pacman se poate misca cu ajutorul sagetilor : right, left, up, down. Acesta trebuie mutat cu cate o tasta pe rand! Tasta nu trebuie tinuta apasata in permanenta, ci pe rand, mutare cu mutare.

- Exista 5 keys raspandite in tot maze-ul, pe care acesta trebuie sa le adune fara a fi prins de ghosts.
- Daca pacman este prins de ghost => moare => game over
- Daca toate cheile au fost colectionate, acesta trebuie sa caute “the treasure chest” pe care il va putea colectiona si astfel va termina jocul => game won
- Daca cheile nu sunt colectionate in totalitate, jocul nu poate fi terminat, in schimb jocul se poate termina chiar daca mancarea nu e toata mancata, insa punctajul va fi mult mai mic, iar asta va fi dezavantajos pentru un nivel ulterior.

4. Implementari ulerioare si dezvoltare

- O implementare ulterioara ar putea fi reprezentata de crearea mai multor nivele ale acestei editii, cu mai multe tipuri de mancare, mai multe reguli legate de ghosts, mai multe tipuri de maze.
- O alta dezvoltare poate fi reperezentata de crearea conceptului de multiplayer, sau crearea conceptului prin care pacman are un inamic in loc de ghosts, iar acest inamic este reprezentat drept cel de al doilea jucator. Astfel, “the evil pacman” trebuie sa il urmareasca si sa il prinda pe pacman, iar pacman trebuie sa fuga si sa se ascunda, acumuland toate punctele posibile, iar cel care reuseste sa isi indeplineasca misiunea va castiga.

5. UML