Gra "Duck shooter"

Napisz program będący grą w stylu "duck shooter". Podczas rozgrywki od lewej do prawej oraz od prawej do lewej poruszają się kaczki różnych kolorów. Gra polega na "zestrzeliwaniu" kaczek i zapobiegnięciu przedostania się ich na drugą stronę okna. Gracz ma na celu klikać w kaczkę tyle razy, ile potrzeba, aby ją "zestrzelić".

Na przykład: żółte kaczki będą potrzebowały 1 kliknięcia, czerwone 5 kliknięć, purpurowe 10, a różowe 20. Są to przykładowe wartości i kolory.

Celem gracza jest utrzymanie jak najdłużej czystego pola bez kaczek. Gra kończy się w momencie, gdy na drugą stronę okna przedostanie się więcej niż 10 kaczek (gracz ma 10 żyć, a każde przepuszczenie kaczki na drugą stronę odbiera 1 życie).

Należy zaimplementować również przeszkody (co najmniej chmury i drzewa), które będą zakrywały kurczaki i chroniły je przed kliknięciami. Chmury są przeszkodami ruchomymi, które muszą poruszać się w lewo lub prawo.

Dodatkowo należy zaimplementować system ulepszeń, w którym kosztem punktów będziemy mogli "ulepszać" naszą broń w pewnym racjonalnym zakresie. Rozgrywka co 5 sekund ma stawać się coraz cięższa - możemy to zrealizować przyśpieszając kaczki, mnożąc przeszkody lub zwiększając "życie" kaczek.

Należy zapewnić w pełni funkcjonalny interfejs graficzny oparty o pliki graficzne (realizacja we własnym zakresie). Konsola poleceń (*CLI*) może być jedynie pomocą informacyjną, ale nie może zachodzić tam żadna znacząca interakcja użytkownika z programem.

Program po uruchomieniu wyświetla menu główne składające się z opcji:

- *New Game* - nowa gra
- *High Scores* - tabela wyników
- *Exit* – wyjście

Po uruchomieniu nowej gry, gracz zostanie zapytany o stopień trudności rozgrywki (implementacja stopni trudności dowolna, co najmniej trzy). Po uruchomieniu gry w nowym oknie wyświetlana jest plansza gry, a licznik czasu rusza (warto zauważyć, że licznik musi być ***realizowany w osobnym wątku*** wykorzystując klasę ***Thread***). Podczas gry musi być widoczny licznik punktów oraz czasu, aktualizowane na żywo podczas rozgrywki. Gra toczy się według wyżej wymienionych zasad do momentu utraty wszystkich żyć. Należy zapewnić możliwość przerwania gry w dowolnym momencie poprzez ***złożony skrót klawiszowy*** (***Ctrl+Shift+Q***), który spowoduje powrót do menu głównego.

Po zakończeniu gry gracz proszony jest o swoją nazwę pod jaką ma być zapisywany w rankingu. Ranking liczony jest w dowolny sposób - np. na podstawie czasu i stopnia trudności (np. liczba zestrzelonych kurczaków*stopień trudności/czas). Należy zapewnić trwałość rankingu po ponownym uruchomieniu aplikacji, czyli należy go przechowywać w postaci pojedynczego pliku na dysku. Należy wykorzystać interfejs ***Serializable*** przy zapisie listy wyników.

Po wybraniu opcji rankingu z menu głównego, zostaje on wyświetlony użytkownikowi. Ponieważ okno rankingu może być relatywnie duże, dlatego należy zadbać o odpowiednie wyświetlanie rankingu (paski przewijania), w razie gdyby nie mieścił się on w oknie racjonalnych rozmiarów.

Wskazówki:

- Należy zadbać o wyjątki w programie. Jeśli jakiś wystąpi należy wyświetlić jego komunikat użytkownikowi.
- Ranking należy zrealizować przy pomocy komponentu ***JList*** oraz własnej implementacji model w oparciu o ***AbstractListModel***.
- Kaczki można realizować przy pomocy przycisków (jednak tak, aby wyglądało to estetycznie), ale można też zaprojektować własny komponent.
- Nie wszystkie okna muszą być realizowane poprzez klasę ***JFrame***. Przy mniejszych i informacyjnych oknach można wykorzystać okna dialogowe.

Projekt opiera się o materiał z zakresu GUI.

Należy zastosować w projekcie wzorzec projektowy MVC.

Uwaga:

- ***Zabrania się wykorzystywania narzędzi WYSIWYG do generowania okien (tzw. Window Builder'ów).***
- ***W przypadku otrzymania projektu ze znacznymi brakami w implementacji lub niekompilującego się, wynikiem za taki projekt będzie 0 pkt.***
- ***Brak znajomości dowolnej linii kodu lub plagiat skutkować będzie wyzerowaniem punktacji za ten projekt.***
- ***W ocenie projektu poza praktyczną i merytoryczną poprawnością będzie brana również pod uwagę optymalność, jakość i czytelność napisanego przez Państwa kodu.***
- ***Ważną częścią projektu jest wykorzystanie między innymi: dziedziczenia, kolekcji, interfejsów lub klas abstrakcyjnych, lambda-wyrażeń, typów generycznych, dodatkowych funkcjonalności lub struktur oraz innych elementów charakterystycznych (ale tylko w naturalny sposób, nic na siłę)***