Dokumentacja

Projekt nr 3 - Logika gry w szachy

Uściślenie projektu i ważne decyzje:

* Zdecydowałem się na stworzenie, zestawu klas, który po połączniu, może prowadzić rozgrywkę w szachy, jednak stwierdziłem, że zajmę się jedynie stworzeniem atrybutów do gry, a nie przeprowadzaniem gry.
* Opierałem się o podstawowe, możliwości szachowe, takie jak wykonanie ruchu, zbijanie, czy szach. Pominąłem m. in. Sprawdzanie czy możliwa jest roszada, sprawdzanie czy gra kończy się remisem, opcję gry na czas.
* Postanowiłem stworzyć obiekty jak najbardziej zbliżone do rzeczywistych używanych podczas gry w szachy, pomimo tego, że zwiększa to znacznie liczbę klas.
* Gdyby figury znały swoje współrzędne, można byłoby zrezygnować z tworzenia szachownicy, co zmniejszyłoby ilość klas. Jednak, przy chęci wykonania ruchu należałoby sprawdzić, wszystkie pionki, w celu stwierdzenia czy żaden ich nie zajmuje wybranej przez nas pozycji do ruchu, a tak to wystarczy wybrać współrzędne i poprzez wskaźnik dowiedzieć się czy pole jest puste czy zajęte.
* W związku z nie przeprowadzaniem rozgrywki, za ruch figur po planszy odpowiada szachownica, gdyż zna ona położenie wszystkich figur.
* Świadomie nie sprawdzam czy jest szach i mat, gdyż uważam, że dobrym pomysłem byłoby w trakcie rozgrywki, gdy obiekt klasy gameStatus stwierdza ,że jest szach symulować ruch na szachownicy obiektu rulesChecker . Jeżeli nie zwróciłby on dla żadnego pola, na którym jest pionek możliwości ruchu, takiego że po nim nie będzie szachu, to oznaczałoby szach i mat.
* Bardzo zależało mi, aby była metoda, która zwraca listę pól z możliwościami ruchu z danego pola, ponieważ fantastycznie, można byłoby zrealizować to graficznie, gdyby po kliknięciu na figurę podświetlały się pola, na które może się ruszyć.

Klasy :

* chessPiece – klasa abstrakcyjna utożsamiona z figurą szachową. Dziedziczą po niej klasy wszystkich poszczególnych figur szachowych.
* King
* Queen
* Rook
* Bishop
* Knight
* Pawn

Wszystkie powyższe klasy reprezentują odpowiednie figury szachowe. Jedyne dane jakimi dysponują to swoja nazwa oraz kolor.

* Field – klasa reprezentująca pole szachownicy. Jej atrybutem jest wskaźnik na figurę która się na niej znajduje, a jeśli jest puste wskaźnik ten ustawiony jest na NULL.
* chessBoard – klasa reprezentująca szachownicę. Składa się z dwuwymiarowej tablicy pól(obiektów klasy Field). Potrafi ustawić bierki przed rozpoczęciem gry, przesuwać figury, sprawdzać czy pole o danych współrzędnych leży na planszy, określać czy dany ruch jest zbiciem.
* rulesChecker – klasa odgrywające rolę sędziego, który zna zasady gry w szachy. Dysponuje szachownicą na podstawie, której określa poprawność wykonywanych ruchów. Sprawdza poprawność każdego wykonanego ruchu.
* gameStatus – klasa określająca jaki jest obecny status gry (np. czy jest szach). Dysponuje szachownicą na podstawie której określa ten status oraz obiektem klasy rulesChecker.
* Test – klasa służąca do testowania automatycznego poszczególnych części programu. Testowane są w niej takie funkcjonalności jak:
  + - Stworzenie szachownicy z pustymi polami
    - Ustawienie poszczególnych figur na płaszczyźnie
    - Poruszanie się poszczególnymi figurami
    - Zwracanie możliwości ruchu dla wybranych figur
    - Sprawdzanie czy jest szach

Zależności między klasami:

* Pojedyncze pole szachowe, może dać informacje jaka figura się na nim znajduję dzięki odpowiednim metodom, bo ma wskaźnik na figurę .
* Analogicznie szachownica może dostarczyć informacje jaka figura znajduję się na każdym polu(jaką ma nazwę i kolor), wystarczy określić pole poprzez współrzędne i użyć odpowiednich metod.
* Obiekt klasy RulesChecker, dzięki przypisaniu szachownicy, na której rozgrywa się rozgrywka zna położenie wszystkich figur. Po wybraniu odpowiedniego pola, dzięki metodzie można otrzymać listę(wektor współrzędnych) wszystkich pól, na które może przemieścić się figura z danego pola.
* Obiekt klasy GameStatus podobnie jak obiekt klasy RulesChecker, może załadować planszę, poprzez metode load. Przy określaniu statusu gry, wykorzystuję wskaźnik na obiekt rulesChecker np. sprawdza czy dla jakaś figura w grze w tym momencie może atakować król, żeby określić czy jest szach.
* Poszczególne figury nie mają pojęcia o swojej, a tym bardziej innych figur pozycji, jedynie szachownica zna położenie figur.
* Klasa Testowa korzysta ze wszystkich innych klas, ponieważ testuje ich

funkcjonalności

Możliwości rozbudowywania projektu:

* W celu przeprowadzenia rozgrywki trzeba byłoby dostarczyć klasę, która kontrolowałaby przebieg gry, przy użyciu klasy RulesChecker oraz GameStatus. Ponadto, niezbędne byłoby sprawdzanie wszystkich ruchów, jak roszada, czy szach mat. Dobrym posunięciem mogłoby być zabranie szachownicy możliwości ruszania figur i przekazanie tej umiejętności graczom.
* Napisanie interfejsu graficznego wymagałoby przypisaniu poszczególnym polom i figurom kolorów oraz narysowanie szachownicy. Uważam, że dzięki jednoznacznych i niezmieniających się współrzędnych pól, wykorzystaniu tego, że figura zna swój kolor i nazwę oraz istniejącej metodzie zwracającej możliwości ruchu, mogłoby to przebiegać łatwiej niż w innym przypadku.
* Aby uczynić grę, bardziej profesjonalną, należałoby wprowadzić np. klasę zegar odmierzającą czas każdego ruchu danego gracza
* Można byłoby również odejść od tradycyjnych szachów i grać w inne odmiany tej gry(np. wybijanka). Wymagałoby to jedynie innych ruchów od graczy, a implementacja logiki byłaby nie zmieniona.
* W celu umożliwienia gry z komputerem, należałoby jedynie zaimplementować odpowiedni algorytm i odpowiednio wpleść go do klasy kontrolującą przebieg gry oraz do interfejsu.