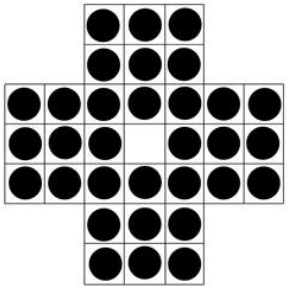
Zadanie 9 5-8 maja 2025 r.

kurs języka C++

samotnik

Instytut Informatyki
Uniwersytetu Wrocławskiego Paweł Rzechonek

Prolog



Samotnik to gra logiczna rozgrywana przez jedną osobę na planszy mającej 33 pola. Rzekomo gra powstała w XVII wieku we Francji. Pierwsza znana rycina dla samotnika wykonana została w 1697 r. przez Claude'a-Auguste'a Bereya. Podobne ale bardziej uproszczone gry były znane wcześniej w starożytnym Rzymie.

Początkowo na 33 kwadratowych polach usytuowanych na planie krzyża rozkładamy 32 pionki (środkowe pole jest puste). Celem gry jest zostawienie na planszy jak najmniejszej liczby pionków. Idealnym rozwiązaniem jest pozostawienie jednego pionka (najlepiej w centrum). Jeżeli tuż za pionkiem jest wolne miejsce, wtedy inny sąsiedni pionek może go przeskoczyć w pionie lub w poziomie i zbić. Nie można przeskakiwać się na ukos oraz nie można bić kilku pionków w jednym ruchu.

Zadanie

Zaprogramuj grę w samotnika. Gra ma być rozgrywana na konsoli. Użytkownik w każdym kroku podaje współrzędne pionka, którym będzie skakać oraz kierunek ruchu. Po każdym ruchu użytkownika jest rysowana na konsoli plansza z aktualnym stanem gry (rysunek wykonany za pomocą semigrafiki). Planszę do samotnika można umieścić na szachownicy o rozmiarach 7×7, więc współrzędne pola określamy podobnie jak w arkuszu kalkulacyjnym podając kolumnę i wiersz (kolumny oznaczamy kolejnymi literami alfabetu A–G a wiersze kolejnymi cyframi 1–7). Kierunek ruchu określamy pojedynczymi literami: L – w lewo (ang. left), R – w prawo (ang. right), U – w górę (ang. up), D – w dół (ang. down). Przykładowy pierwszy ruch na planszy można zapisać następująco: F4L (albo małymi literami `f4l`).

Zaprogramuj grę zgodnie z wzorcem MVC: zdefiniuj osobną klasę do przechowywania stanu gry (model) oraz funkcje do komunikacji z użytkownikiem (view-controller), czyli do odebrania polecenia, jego interpretacji, wykonania ruchu oraz do prezentacji planszy z aktualnym stanem gry.

W trakcie działania programu zgłaszaj wyjątki, gdy użytkownik wprowadzi błędne polecenie albo błędne parametry w poprawnym składniowo poleceniu. Zaprojektuj zatem własną hierarchię klas wyjątków na potrzeby gry, zaczynając od bazowej klasy `wyjątek_samotnika`

dziedziczącej po `std::logic_error`. Wszystkie zgłaszane w programie wyjątki należy wyłapać i adekwatnie zareagować.

Uwaga

Gdy gra się zakończy (użytkownik wygrał pozostawiając na planszy jeden pionek, albo przegrał bo nie może wykonać już żadnego ruchu, albo się poddał) należy wyświetlić informacje o stanie gry (ile pionków pozostało na polu i czy użytkownik wygrał czy przegrał) oraz czas trwania gry.

Ważne elementy programu

- Definicja obiektu reprezentującego stan gry.
- Interpreter poleceń użytkownika.
- Semigrafika do zobrazowania planszy.
- Definicja własnej hierarchii klas wyjątków dziedziczących po `std::logic_error`.
- Zgłaszanie i łapanie wyjątków w trakcie pracy programu.