Projekt zadania zaliczeniowego z programowania obiektowego 2004/2005

Piotr Truszkowski

29 listopada 2004

Spis treści

1 Wstęp — przedstawienie problemu

Zadanie zaliczeniowe polega na napisaniu gry. Gracz wciela się w grupę rycerzy mających na celu znalezienie legowiska smoka. Aby tam dotrzeć rycerze muszą przejść labirynt, w którym to rozmieszczone są rozmaite pułapki. Ruchy odbywają się na sposób turowy, gdyż na skutek magii czas zatracił tam swoją ciągłość.

2 Ogólny zarys rozwiązania problemu

Rozwiązanie będzie oparte na technice programowania obiektowego. W tym celu posłużę się jezykiem programowania Smalltalk, używając oprogramowania Dolphin.
Każde pole labiryntu jest obiektem, któremu można przypisywać jako atrybut rycerzy. Przez cała gre pola są niezmiennicze z dokładnością do ich atrybutów, tj nie ma takiej sytuacji, że jakiś obiekt zmienia się w inny czy też następuje podmiana. Rycerze również są obiektami posiadającymi pewne cechy (atrybuty), przechodząc nimi plansze mogą oni dokonywać zmian w atrybutach pól (wybuchy min, zjadanie murów...) Nad właściwym przebiegiem gry czuwa obiekt klasy – **Gra**. Właśnie w tym obiekcie występują powiązania z interfejsem. Sam interfejs będzie graficzny.

2.1 Hierarchia klas

Podział na klasy powstał w głównej mierze w oparciu na idei: obiekt – rzeczownik, metoda – czynność, atrybut – przymiotnik. I tak na przykład rycerz ze stalową szczęką należy do klasy **Zjadacz** (nadklas: **Wyposazony** i **Rycerz**). Wykonuje on czynności takie jak zjadanie muru oraz zwykłe chodzenie (i inne...). Określają go atrybuty, przymiotniki, ile ma specjalnych zębów, jak wygląda itp.

- ◇ **Object**
  - ◇ **Pole** — *(klasa abstrakcyjna)*
    - ◇ **Zwykle**
    - ◇ **Legowisko**
    - ◇ **Pulapka** — *(klasa abstrakcyjna)*
      - ◇ **Dziura**
      - ◇ **Mina**
    - ◇ **Mur** — *(klasa abstrakcyjna)*
      - ◇ **Wieczny**
      - ◇ **Zjadalny**
    - ◇ **Magazyn** — *(klasa abstrakcyjna)*
      - ◇ **Dentystyczny**
      - ◇ **Budowlany**
    - ◇ **Poza**

- ◇ **Object**
  - ◇ **Rycerz** — *(klasa abstrakcyjna)*
    - ◇ **Nadzwyczajny**
    - ◇ **Skoczek**
    - ◇ **Wyposazony** — *(klasa abstrakcyjna)*
      - ◇ **Budowniczy**
      - ◇ **Zjadacz**
  - ◇ **Plansza**
  - ◇ **Wspolrzedne**
  - ◇ **Gra**

Część z powyższych klas jest jedynie abstrakcyjna (jak **Pole**, **Wyposazony**...) aby uwydatnić pewien wspólny sposób zachowywania się obiektów różnych klas. Pominąłem opis klas odpowiedzialnych za interfejs graficzny, omówię je w punkcie **4**.

2.2 Powiązania między klasami

Czesto będzie dochodzić do interakcji pomiędzy obiektami.

Gra Obiekt klasy **Gra** odpowiedzialny za właściwy przebieg gry często wywołuje metody na obiektach klasy **Rycerz** (aby wykonali ruch, ściślej to na obiektach podklas tej klasy jak **Skoczek**, **Zjadacz**...) oraz na obiekcie **Plansza** w celu uaktualniania jej po upłynięciu każdej jednostki czasu.

Plansza Obiekt tej klasy zna pola labiryntu, więc to właśnie ten obiekt będzie przekazywał dalej metodę uaktualniającą te pola (obiektów klas: **Zwykle**, **Dziura**...).

Rycerz Mam tu na myśli obiekty podklas tej klasy. One to będa się często komunikować z polami labiryntu a to wchodząc na nie, zjadając mury, teleportując się, łatając dziurę, itp.

Inne... Dochodzi także do interakcji między obiektami tych samych klas (bądź obiektów podobnych – rozumiem przez to np: **Zjadalny** i **Mina**, jedno i drugie jest polem, podobnie z rycerzami). I tak na przykład wybuch miny musi wywierać odpowiednie reakcje u sąsiadów.
Ponadto znajdowanie sąsiadów wymaga współpracy jakiegoś pola z planszą, to pole się „pyta" kto jest obok (alternatywną wersją mogłoby być pamiętanie w atrybucie przez pole kogo ma za sąsiada).

3 Szczegółowy opis klas

3.1 Pole

Atrybuty	Opis
wsp	Współrzedne pola na planszy.
ktojest	Informacja kto jest na tym polu (kolekcja).
czyodwiedzone	Informacja czy pole było już odwiedzone.
czywidoczne	Informacja czy pole jest widoczne.
wygladwid	Wygląd pola widocznego.
wygladniewid	Wygląd pola niewidocznego.
plansza	Plansza.

Metody klasowe	Opis
newX: x y: y	Nowy obiekt tej klasy, ustawiany jest też atrybut wygląd-niewid, jako że każde pole wygląda tak samo gdy jest nie widoczne.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaWejsc	Sprawdzenie czy rycerz może się ruszyć na to pole. Zwraca true lub falsew zależności czy jest jakiś rycerz na tym polu.
czyJestKtos	Sprawdza czy jest jakiś rycerz na polu (true/false).
czyMoznaZjesc	Zwraca false, ta metoda jest przedefiniowana dla muru zjadalnego.
czyMoznaZalatac	Zwraca false, ta metoda jest przedefiniowana dla dziury.
czyMoznaTeleportowac	Zwraca true. Domyślnie można się przenieść, przedefiniowane w odpowiednich podklasach.
czyMoznaZeby	Zwraca false, ta metoda jest przedefiniowana dla magazynu dentystycznego.
czyMożnaCegly	Zwraca false, ta metoda jest przedefiniowana dla magazynu budowlanego.
pokaz	Wyswietla wyglad obiektu w zalezności czy widoczny czy nie. Wywołuje również metode pokaż dla ewentualnie obecnego rycerza.
widoczne	Zmienia wartosc atrybutu czywidoczne na true.
uwidocznijSasiadow	Wywołuje metodę *widoczne* od sąsiadów.

Metody instancyjne	Opis
wejdz: rycerz	Atrybutowi ktojest jest dopisywany rycerz a atrybut czyodwiedzone zmieniany jest na true, jeśli miał wartość false wywoływana jest metoda *uwidocznijSasiadow*. Zwraca false jeśli rycerz nie może wejść na pole, w przypadku powiedzenia się – true.
wybuch	Wysadza pole, tj zabija rycerza obecnego na polu.
wyjdz	Rycerz wychodzi z pola, rycerz jest odpisywany z atrybutu ktojest.
tiktak	Domyślnie nic nie robi, w obiektach innych klasach uaktualnia stan obiektu.
sasiedzi	Wyznacza pola sąsiadujące (zwraca ich kolekcję).
wypisz	Wypisuje odpowiednie informacje o polu (tekst).

Klasa abstrakcyjna. Obiekty tej klasy reprezentują przestrzeń po której się poruszają rycerze.

3.2 Zwykle

Metody klasowe	Opis
newX: x y: y	Nowy obiekt tej klasy. Tak jak w naklasie dodatkowo ustalany jest atrybut wygladwid (dla każdego obiektu tej klasy ten atrybut jest taki sam).

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).

Obiekty tej klasy reprezentują pola zwykłe w labiryncie. Czyli pola, po których rycerze mogą się poruszać „do woli”. Wszystkie niezbędne atrybuty i metody są bezpośrednio dziedziczone z klasy **Pole**.

3.3 Legowisko

Metody klasowe	Opis
newX: x y: y	Nowy obiekt tej klasy, ustawiany jest atrybut wygladwid.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).

Metody instancyjne	Opis
czyMoznaWejsc	Przedefiniowane, zwraca true. Do legowiska może wejść kliku rycerzy.
wybuch	Nic nie robi – jeśli są tu rycerze to się dobrze schowali i wybuch nie wyrządza im krzywdy.
wypisz	Tekstowa informacja o polu.
pokaz	Pokazanie obiektu na planszy. Dodatkowo pokazuje ilu jest rycerzy na polu.

Cel wyprawy rycerzy. Rycerz po dotarciu na to pole jest bezpieczny i pozostaje mu jedynie czekać na resztę swoich współtowarzyszy. Ponadto zauważmy, ze na tym polu może przebywać kilku rycerzy. Istnieje tylko jedno takie pole jak wynika ze specyfikacji. Wszak zdefiniowanie kilku takich pól w pliku konfiguracyjnym uważam za błędną konfigurację planszy, o której powiadamiać będzie program po wczytaniu pliku.

3.4 Pułapka

Atrybuty	Opis
licznik	Określa za jaką ilość tur ów obiekt będzie aktywną pułapką.
wygladnieakt	To jak wygląda pułapka gdy nie jest aktywna.
czasreaktywacji	Czas po jakim pułapka się uaktywnia.

Metody klasowe	Opis
newX: x y: y	Tak samo jak w nadklasie dodatkowo ustawia licznik na 0 i czasreaktywacji.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
wejdz: rycerz	Jeśli licznik jest większy od zera metoda działa jak w nadklasie w przeciwnym przypadku zabija rycerza.
zabij	Zabija rycerza jeśli ten znajduje się na tym polu.
tiktak	Zmniejsza licznik jeśli licznik wyniesie 0 zabija.
pokaz	Jeśli licznik jest równy 0 metoda zachowuje się jak ta sama metoda z nadklasy, w przeciwnym przypadku wyświetla atrybut wygladnieakt.

Klasa abstrakcyjna zawierająca dwie podklasy **Dziura** i **Mina** (łatwiej teraz dorzucić jeszcze jakiś typ pułapki). Ich podobieństwo spowodowało, że nadałem im tę wspólną nadklasę. Obie powodują śmierć u rycerza gdy ten na to pole wejdzie. Warto zauważyć, że w pewnych okolicznoś-

ciach mur **Zjadalny** też jest pułapką (rycerz stojący na tym polu może zginąć). Jednakże określiłem jego przynależność do klasy **Mur** gdyż intuicyjnie to właśnie z murem **Wiecznym** ma więcej wspólnego niż z **Dziurą**. Można również rozważać czy nie robić osobnych klas dla pułapek aktywnych i nieaktywnych. Jednakże według mnie nie ma takiej potrzeby, gdyż normalną rzeczą jest, że pewnien obiekt (choćby w naturze) zmienia się a mimo tego pozostaje ten sam. Tak samo jest tu, zarówno **Mina** jak i **Dziura** jedynie przez pewien moment pozostają w „uśpieniu".

3.5 Dziura

Atrybuty	**Opis**
szerokosc	Liczba cegieł potrzebna do załatania.

Metody klasowe	**Opis**
newX: x y: y	Tak samo jak w nadklasie dodatkowo ustawia licznik na 0, wygląd, czasreaktywacji...

Metody instancyjne	**Opis**
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaZalatac	Zwraca true jesli dziura nie jest zalepiona, jak jest zwraca false.
zalataj: n	Dziura jest łatana, metoda zwraca ilość nieużytych cegieł. Jeśli niemożna było załatać zwróci n cegieł.
wybuch	Zabija rycerza oraz jeśli Dziura była zalepiona niszczy ją.
wypisz	Tekstowa informacja o polu.

Dziura ma swoją szerokość. A poza tym ten obiekt jak żaden inny nie posiada takiej głębi... I to bezdennej.

3.6 Mina

Metody klasowe	**Opis**
newX: x y: y	Tak samo jak w nadklasie dodatkowo ustawia licznik na 0, wygląd, czasreaktywacji...

Metody instancyjne	**Opis**
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).

Metody instancyjne	Opis
wysadz	Wysadza to pole oraz pola sąsiadujące, wysyła metode wybuch do sąsiadów. Licznikowi zmienia wartosc na czasreaktrywacji.
wybuch	Zabija rycerza oraz wysadza minę jeśli ta jest aktywna.
wypisz	Tekstowa informacja o polu.

Temu polu natomiast można pozazdrościć wystrzałowości...

3.7 Mur

Metody klasowe	Opis
newX: x y: y	Nowy obiekt klasy.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaWejsc	Zwraca false.
czyMoznaTeleportowac	False – nie da się wrzucić rycerza w ścianę (chyba, że sciana jest zjedzona ale o tym w podklasie).
wybuch	Nic nie robi.

Nadklasa dla **Wieczny** i **Zjadalny**. Obiekty z obu tych klas są przecież murami.

3.8 Wieczny

Metody klasowe	Opis
newX: x y: y	Nowy obiekt klasy.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
wypisz	Tekstowa informacja o polu.

Nikt nie może wejść, nic go nie zmienia. Jedynie czym może się różnić od innego obiektu tej samej klasy to to, że jeden jest widoczny a drugi nie.

3.9 Zjadalny

Atrybuty	Opis
wygladruiny	Opisuje wygląd muru zburzonego wybuchem miny bądź zjedzonego.
czasreaktywacji	Czas po jakim mur się odbudowywuje.

Metody klasowe	Opis
newX: x y: y	Działa tak samo jak w nadklasie tyle tylko, że ustawia czasreaktywacji, wygladwid, itd...

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
wybuch	Niszczy mur jeśli ten ma licznik równy 0 i ustawia go na czasreatywacji.
czyMoznaTeleportowac	Można jeśli mur jest zburzony bądź zjedzony.
czyMoznaZjesc	Jeśli licznik jest równy 0 to daje true w przeciwnym przypadku false.
zjedz: rycerz	Mur jest zjadany do wartości atrubutu ktojest dopisywany jest rycerz. Zmieniana jest również wartośc licznika na czasreaktywacji.
tiktak	Zmienia wartość licznika o 1, jesli zmieni na 0 to mur odrasta i zabija rycerza jeśli ten znajduje się na polu.
wypisz	Tekstowa informacja o polu.

Mur słabszej konstrukcji niż wieczny. Może go zniszczyć wybuch miny lub rycerz-zjadacz. Wtedy to pole jest przez pewnien czas dostępne dla każdego rycerza, potem mur odrasta zabijając wszytko w swoim zasięgu. Każdej tury wywoływana jest metoda tiktak uaktualniająca stan obiektu. Tutaj podobnie jak w **Pulapka** można rozpatrywać utworzenie klas odpowiadającym obiektom muru do zjedzenia, wysadzonego czy zjedzonego. I tak jak wcześniej uważam to za zbędne, gdyż mur pozostaje ten sam, co najwyżej nadgryziony gdzie po pewnym czasie się „regeneruje”.

3.10 Magazyn

Atrybuty	Opis
pojemnosc	Pojemność magazynu.
ilejest	Ile jest materiału w magazynie.
licznik	Za ile tur będzie nowy materiał.
czasreaktywacji	Szybkość produkcji.

Metody klasowe	Opis
newX: x y: y poj: pojemnosc	Działa jak w nadklasie oraz dodatkowo ustala pojemność, ustawia atrybuty licznik i ilejest.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
pobierz: n	Pobiera n składników, dodając je stojącemu na polu rycerzowi oraz zmiejszając wartość atrybutu ilejest o n, oraz zmieniając licznik na czasreaktywacji (jeśli nie ma tylu cegieł ile chce rycerz dodawane są całe zasoby). Zwraca liczbę pobranych cegieł.
tiktak	Zmniejsza licznik o 1. Gdy zmieni na 0 zwiększa się o 1 wartość atrybutu ilejest. Wartość licznika w zależności od pojemności magazynu jest zmianiana na czasreaktywacji lub pozostaje na 0 gdy magazyn jest pełny.

Nadklasa dla **Dentystyczny** i **Budowniczy**. Ze zrozumiałych powodów ta klasa łączy pewne cechy dwóch pozostałych.

3.11 Dentystyczny

Metody klasowe	Opis
newX: x y: y poj: pojemnosc	Działa jak w nadklasie oraz dodatkowo ustala pojemność, ustawia atrybuty licznik, ilejest, itd...

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaZeby	Zwraca true.

Metody instancyjne	Opis
PobierzZeby: n	Pobiera n żelaznych zebów (opiera się na metodzie pobierz z nadklasy).
wypisz	Tekstowa informacja o polu.

Magazyn ze stalowymi zębami, zaopatrzenie dla rycerzy-zjadaczy.

3.12 Budowlany

Metody klasowe	Opis
newX: x y: y poj: pojemnosc	Działa jak w nadklasie oraz dodatkowo ustala pojemność, ustawia atrybuty licznik, ilejest, itd...

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaCegly	Zwraca true.
PobierzCegly: n	Pobiera n cegieł (opiera się na metodzie pobierz: z nadklasy).
wypisz	Tekstowa informacja o polu.

Zaopatrzenie dla rycerzy-budowniczych w cegły.

3.13 Teleport

Atrybuty	Opis
dokąd	Istniejący obiekt tej klasy, bądź nil.

Metody Klasowe	Opis
newX: x y: y dokad: dokad	Tak jak w nadklasie ale dodatkowo ustawia atrybut dokąd.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
CzyMoznaTeleportowac	Zwraca false, nie można teleportować się do drugiego teleportu.
Teleportuj	Przenosi rycerza na pole dokad. W zależności od powiedzenia się zwraca true lub false.

Metody instancyjne	Opis
wypisz	Tekstowa informacja o polu.

Obiekt tej klasy teleportuje wchodzącego rycerza do innego miejsca labiryntu.

3.14 Poza

Atrybuty	Opis
IstniejącyObiekt	Atrybut klasowy! Istniejący obiekt tej klasy, bądź nil.

Metody Klasowe	Opis
new	Tworzy obiekt tylko raz, przy powtórnym wywołaniu zwraca obiekt tej klasy już istniejący.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
czyMoznaWejsc	Zwraca false – niemożna wyjść poza planszę.
czyMoznaTeleportowac	False – nie da się wyteleportować z labiryntu.
wybuch	Nic nie robi
widoczne	Nic nie robi

Klasa reprezentująca te pola które są poza labiryntem. Ponieważ dla istoty gry nie są ona ważne dlatego interpretuje je jako tylko jedno pole. Mianowicie wszystkie pola innych klas składają się na pola labiryntu a to pole na wszystko co jest poza labiryntem (czyli wystarczy tylko jeden obiekt tej klasy!). Warto zauważyć, że definiuję tak obiekt tej klasy, iż istnieje tylko jeden taki obiekt klasy **Poza**.

3.15 Rycerz

Atrybuty	Opis
wyglad	Wygląd rycerza.
czyzaznaczony	Informuje czy rycerz ma własnie ruch.
stan	Informuje czy rycerz szuka(1), znalazl(2), matrwy(0).

Metody klasowe	Opis
new	Stworzenie rycerza.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
rusz: kierunek	Wykonuje ruch rycerza we wskazanym kierunku. Zwraca true lub false w zależności od poprawności ruchu.
smierc	Rycerz ginie,
pokaz	Wyświetlenie rycerza na polu (metoda wywoływana przez metode *pokaz* w **Polu**).
wypisz	Tekstowa informacja o rycerzu.
czyZyje	True lub false. Być może rycerz odpowiada już ostatni raz...

Klasa abstrakcyjna, zestaw podstawowych metod i atrybutów dziedziczonych przez podklasy. Nadklasa dla **Nadzwyczajny**, **Skoczek** i **Wyposazony**. W metodzie *rusz*: pojawia się argument kierunek. Definiuję go jako jedną z liczb $2, 4, 6, 8$ (2 – dół, 8 – góra, 4 – lewo, 6 – prawo, tak jak na klawiaturze „numerycznej"). Tak samo będzie zdefiniowany poźniej w metodach *skocz:*, *zalataj:*, *zjedz:*.

3.16 Nadzwyczajny

Atrybuty	Opis
cecha	Niewiarygodny upór.

Metody klasowe	Opis
new	Stworzenie rycerza.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
wypisz	Tekstowa informacja o rycerzu.

Najdzielniejszy ze wszystkich, zaprawiony w bojach rycerz. W odróżnienia od reszty cechuje go wyjątkowy upór.

3.17 Skoczek

Metody klasowe	Opis
new	Stworzenie rycerza.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
skocz: kierunek	Rycerz skacze we wskazanym kierunku (true/false).
wypisz	Tekstowa informacja o rycerzu.

Rycerz niedoświadczony, cechuje go zbytni entuzjazm i nierozwaga. Potrafi skakać o 2 pola.

3.18 Wyposazony

Atrybuty	Opis
iloscekw	Liczba posiadanych egzemplarzy ekwipunku rycerza.
maxekw	Pojemność worka na ekwipunek rycerza.

Metody klasowe	Opis
newIl: iloscekw	Nadawanie stosownych wartosci atrybutom nowemu obiektowi tej klasy.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
wez	Rycerz bierze z magazynu ekwipunek.

Nadklasa dla **Budowniczy** i **Zjadacz**. Abstrakcyjna klasa, grupuje tych rycerzy, którzy posiadają dodatkowo jakiś ekwipunek.

3.19 Budowniczy

Metody klasowe	Opis
newIl: iloscekw	Stworzenie rycerza i nadanie mu właściwych atrybutów.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
rusz: kierunek	Tak jak w nadklasie z tym, że dodatkowo po wejściu do magazynu następuje „doładowanie”.
wezCegly	Rycerz napelnia swój wór cegłami z magazynu. Zwraca tru lub false.

Metody instancyjne	Opis
zalataj: kierunek	Rycerz łata dziurę która znajduje się we wskazanym kierunku. W wyniku zwraca true lub false w zależności od wykonania pracy.
wypisz	Tekstowa informacja o rycerzu.

Ryczerz z doświadczeniem inżynierem. Dzięki swojej wiedzy potrafi szybko zabudowywać dziury. Posiada worek przystosowany do przenoszenia cegieł.

3.20 Zjadacz

Metody klasowe	Opis
newIl: iloscekw	Stworzenie rycerza i nadanie mu właściwych atrybutów.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
rusz: kierunek	Tak jak w nadklasie z tym, że dodatkowo po wejściu do magazynu następuje „doładowanie”.
wezZeby	Rycerz uzupełnia swój garnitur zębów. (true/false).
zjedz: kierunek	Rycerz zjada mur (true/false).
wypisz	Tekstowa informacja o rycerzu.

Weteran wojenny, po pewnej bitwie doznał obrażeń szczęki. Wtedy to zbudowano dla niego stalową szczękę. Dzięki czemu może na niej montować specjalne zęby kruszące mury.

3.21 Plansza

Atrybuty	Opis
pola	Kolekcja pól.

Metody klasowe	Opis
new	Tworzy obiekt z pustą kolekcja pól.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
dodaj: pole	Dodaje pole do kolekcji pól.
sąsiedzi: pole	Określenie sąsiadów zadanego pola.

Metody instancyjne	Opis
pole: pole kierunek: kierunek	Zwraca pole we wskazanym kierunku od zadanego pola.
wspX: x Y: y	Zwraca obiekt o podanych współrzędnych.
tiktak	Uaktualnia planszę czyli zwiększa liczniki (jak są) u pól z kolekcji.

Obiekt reprezentujący świat. Zatem i labirynt i wszystko co poza nim (obiekt klasy **Poza**) kierunek zdefiniowany jak wczesniej.

3.22 Wspolrzedne

Atrybuty	Opis
x	Współrzedna x.
y	Współrzędna y.

Metody klasowe	Opis
newX: x Y: y	nadaje wartości współrzędnym.

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
x	Zwraca atrybut x.
y	Zwraca atrybut y.

Każde pole ma współrzędne, która jest obiektem tej klasy.

3.23 Gra

Atrybuty	Opis
plansza	Plansza.
rycerzeZywi	Rycerze żyjący.
rycerzeOk	Rycerze którzy dotarli do legowiska.
rycerzeMartwi	Rycerze martwi.

Metody klasowe	Opis
newPl: plansza Ryc: rycerze	Inicjowany jest obiekt reprezentujący grę

Metody instancyjne	Opis
initialize: ...	Inicjuje odpowiednie wartości dla atrybutów (metoda wywoływana przez newX: y:).
zarzadzajGra.	Pętla, która przydziela kolejno ruchy rycerzom, oraz każdej tury aktualizuje plansze.
ruch: rycerz	Czeka aż rycerz wykona prawidłowy ruch.
tiktak	Uaktualniacz planszy.
wczytajGre: plik	Wcztanie gry z pliku, w razie nie powodzenia zwraca false.

Obiekt tej klasy reprezentuje grę jaką toczy gracz, w tej klasie znajdują się metody czuwające nad turowościa gry, aby pola na planszy były aktualizowane, aby każdy z rycerzy miał jeden ruch w turze, aby gra mogła się zakończyć gdy odpowiednia ilość rycerzy dojdzie do legowiska lub gdy wiadomo, że to nie może mieć miejsca. Być może w tej klasie pojawią się jeszcze jakieś pomocnicze metody.

4 Interfejs

Interfejs gry będzie w pełni graficzny. Intuicyjność będzie się opierała w głównej mierze na okienku statusu gdzie program będzie powiadamiał gracza o stanie gry, opisie zadanego pola i który rycerz ma ruch. Gracz będzie sterować rycerzami za pomocą myszki klikając w odpowiednie przyciski. Obok wyświetlonej planszy z polami będa się znajdować przyciski odpowiedzialne za ruch oraz w przypadku budowniczego – przycisk załatania dziury, zjadacza – przycisk zjedzenia muru, skoczka – przycisk skoku.
Interfejs będzie się opierał na MVP — Model View Presenter. Zaimplementowane zostaną również metody odpowiedzialne za obsługę zdarzeń.