Programowanie Obiektowe – Labirynt

Gra Labirynt polega na poruszaniu się po podziemiach, zbieraniu skarbów i rozwiązywaniu zadań.
Aby ją ukończyć trzeba uzbierać określoną dla wybranej wersji labiryntu liczbę punktów.
Podziemia składają się z komnat połączonych jednokierunkowymi, otwartymi przejściami oraz dwukierunkowymi drzwiami zamkniętymi na klucz. Aby przejść przez dane drzwi, gracz musi wcześniej znaleźć odpowiadający im klucz. W momencie rozpoczęcia gry, gracz ma okazję zapoznać się z poleceniami, którymi może się posługiwać.

Klasa Labirynt implementuje strukturę grafu, gdzie wierzchołkami są komnaty, a krawędziami przejścia oraz drzwi. W komnacie mogą się znajdować osoby (złodzieje i gracz) oraz przedmioty (skarby i klucze) i zadania.
W projekcie został wykorzystany wzorzec Polecenie.
Zakłada on utworzenie oddzielnej klasy dla każdego polecenia implementującej wspólny interfejs Polecenie.
Konkretne polecenie implementuje akcję w postaci metody wykonaj().
Konkretne polecenie odnosi się do odbiorcy, który wie jak wykonać dane zadanie (np. w klasie rozejrzyjSie zostaje wywołana metoda rozejrzyjSie() z klasy Gracz).
Mamy również nadawcę, który ustala odbiorcę akcji i wywołuje wykonaj() z odpowiedniej klasy (tutaj nadawcą jest Gra).
Klasa Gra zawiera labirynt oraz tablicę klas implementujących interfejs Polecenie. W zależności od akcji podjętej przez gracza zostaje wywołana metoda wykonaj|() z odpowiedniej klasy.

KLASA Labirynt

Klasa Labirynt zawiera całą strukturę labiryntu (listę komnat, listę przejść między komnatami, listę drzwi, itd.). Zawiera również kilka gotowych wersji labiryntu (level1, level2, level3).

pola:
ArrayList<Komnata> listaKomnat - lista komnat znajdujących się w labiryncie
ArrayList<Przejscie> listaPrzejsc - lista przejść znajdujących się w labiryncie
ArrayList<Drzwi> listaDrzwi - lista drzwi znajdujących się w labiryncie
ArrayList<Zlodziej> listaZlodziei - lista zlodziei znajdujących się w labiryncie
ArrayList<Przedmiot> listaPrzedmiotow - lista przedmiotow znajdujących się w labiryncie
ArrayList<Zadanie> listaZadan - lista zadań znajdujących się w labiryncie
Gracz gracz
boolean koniecGry - gdy ma wartość true, następuje koniec gry
int lapowka - liczba punktów, którą trzeba osiągnąć, by móc ukończyć grę

metody:
Labirynt() – konstruktor
void dodajZlodzieja(String nazwa, Komnata komnata) – dodaje złodzieja o podanej nazwie do podanej komnaty
void dodajKomnate(int nr, String o) – dodaje do labiryntu komnatę o podanym numerze i opisie
void dodajPrzejscie(String n, Komnata k1, Komnata k2) – dodaje do labiryntu przejście o nazwie n, z komnaty k1 do komnaty k2
void dodajDrzwi(String n, Komnata k1, Komnata k2) – dodaje do labiryntu drzwi o nazwie n, pomiędzy komnatami k1 i k2

void dodajSkarb(String n, Komnata k, int war) – dodaje do komnaty k skarb o nazwie n i wartości war
void dodajKlucz(String n, Komnata k, Drzwi d) – dodaje do komnaty k klucz o nazwie n, do podanych drzwi d
void dodajZadanie(String n, Komnata k, String p, String o, int war) – dodaje do komnaty k zadanie o podanej nazwie (n), treści (p) wraz z odpowiedzią (o) oraz wartością punktową zadania (war)
level1() – wersja nr 1
level2() – wersja nr 2
level3() – wersja nr 3
main() – wybór wersji oraz uruchomienie gry

KLASA Komnata

pola:
int numer – numer komnaty
String opis – opis komnaty
Labirynt labirynt – labirynt, w którym się znajduje komnata

metody:
Komnata(int n, Labirynt l) – konstruktor, tworzy komnatę o numerze n w labiryncie l
Komnata(int n, String o, Labirynt l) - konstruktor, tworzy komnatę o numerze n i opisie o w labiryncie l
void wypiszPrzedmiotyZKomnaty() – wypisuje wszystkie przedmioty znajdujące się w komnacie (skarby i klucze)
void wypiszZadaniaZKomnaty() – wypisuje wszystkie zadania znajdujące się w komnacie
void wypiszPrzejsciaZKomnaty() – wypisuje wszystkie przejścia z komnaty
void wypiszDrzwiZKomnaty() – wypisuje wszystkie drzwi znajdujące się w komnacie wraz z ich stanem (otwarte/zamknięte)

KLASA Przedmiot

pola:
String nazwa – nazwa przedmiotu
String opis – opis przedmiotu
Osoba wlasciciel – właściciel przedmiotu (jeśli nikt tego przedmiotu nie posiada, to wlasciciel przyjmuje wartość null)
Komnata komnata – komnata, w której znajduje się przedmiot (jeśli ktoś ten przedmiot posiada, to komnata przyjmuje wartość null)

metody:
Przedmiot(String n, Komnata k) – konstruktor, tworzy przedmiot o nazwie n i przypisuje go do komnaty k
Przedmiot(String n, String o, Komnata k) – konstruktor, tworzy przedmiot o nazwie n oraz opisie o i przypisuje go do komnaty k
void sprawdz() – wyświetla opis przedmiotu

KLASA Skarb

dziedziczy po klasie Przedmiot

pola:
int wartosc – wartość skarbu

metody:
Skarb(String n, Komnata k, int war) – konstruktor, tworzy skarb o nazwie n i wartości war i przypisuje go do komnaty k

Skarb(String n, String o, Komnata k, int war) – konstruktor, tworzy skarb o nazwie n, opisie o i wartości war i przypisuje go do komnaty k

KLASA Klucz

dziedziczy po klasie Przedmiot

pola:
Drzwi drzwi – drzwi, do których pasuje klucz

metody:
Klucz(String n, Komnata k, Drzwi d) - konstruktor, tworzy klucz o nazwie n, przypisuje go do drzwi d i umieszcza w komnacie k

KLASA Przejscie

pola:
String nazwa – nazwa przejścia
Komnata komnata1 – komnata, z której gracz może wyjść przejściem
Komnata komnata2 – komnata, do której gracz się może dostać przechodząc przez przejście

metody:
Przejscie(String n, Komnata k1, Komnata k2) – konstruktor, tworzy przejście o nazwie n, z komnaty k1 do komnaty k2

KLASA Drzwi

dziedziczy po klasie Przejscie

pola:
boolean czyOtwarte – jeśli drzwi zostały otwarte, przyjmuje wartość true

metody:
Drzwi(String n, Komnata k1, Komnata k2) – konstruktor, tworzy drzwi o nazwie n, pomiędzy komnatami k1 i k2
void otworz() – otwiera drzwi (ustawia wartość czyOtwarte na true)
void zamknij()- zamyka drzwi (ustawia wartość czyOtwarte na false)

KLASA Osoba

pola:
String nazwa – nazwa osoby
Komnata komnata – komnata, w której dana osoba się znajduje

metody:
Osoba(String n, Komnata k) – konstruktor, tworzy osobę o nazwie n i umieszcza ją w komnacie k
void wez(Przedmiot p) – przypisuje dany przedmiot do osoby
void odloz(Przedmiot p) – przypisuje dany przedmiot do komnaty, w której osoba się znajduje (osoba przestaje być jego właścicielem tego przedmiotu)

KLASA Gracz

dziedziczy po klasie Osoba

pola:
int majatek – wartość majątku jaki gracz posiada
ArrayList<Skarb> listaSkarbow – lista skarbów, które gracz posiada
ArrayList<Klucz> listaKluczy – lista klucz, które gracz posiada

metody:
Gracz(String n, Komnata k) – konstruktor, tworzy gracza o nazwie n i umieszcza go w komnacie k
void wez(Przedmiot p) – rozszerza metodę wez(Przedmiot p) z klasy Osoba o dopisywanie przedmiotu do odpowiedniej listy (listy skarbów lub listy kluczy) oraz zwiększenie majątku gracza w przypadku wzięcia skarbu
void odloz(Przedmiot p) – rozszerza metodę odloz(Przedmiot p) z klasy Osoba o usuwanie przedmiotu z odpowiedniej listy (listy skarbów lub listy kluczy) oraz zmniejszenie majątku gracza w przypadku odłożenia skarbu
void rozejrzyjSie() – wyświetla opis komnaty i wypisuje wszystko co się w niej znajduje (przedmioty, drzwi, przejścia, zadania)
void przejdz(Przejscie p) – jeśli przejście przez dane przejście lub drzwi jest możliwe, zmienia położenie gracza
void wypiszCoMam() – wypisuje wszystkie przedmioty, które gracz posiada

KLASA Zlodziej

dziedziczy po klasie Osoba

pola:
ArrayList<Komnata> listaKomnat – lista komnat, do których złodziej może przejść z komnaty, w której się znajduje
ArrayList<Skarb> listaKradzionych – lista skarbów, które złodziej ukradł

metody:
Zlodziej(String n, Komnata k) – konstruktor, tworzy złodzieja o nazwie n i umieszcza go w komnacie k
void aktualizujListeKomnat() – aktualizuje listę komnat, do których złodziej może przejść z komnaty, w której się znajduje
void przejdz() – przemieszcza złodzieja do losowej z komnat, do których może on przejść z tej, w której się aktualnie znajduje
void kradnij() – jeżeli gracz posiada jakiś skarb, to złodziej kradnie mu losowy ze skarbów z listy
void oddaj() – jeżeli złodziej posiada jakiś skarb, to oddaje go graczowi

KLASA Zadanie

pola:
String nazwa – nazwa zadania
Komnata komnata – komnata, w której się znajduje zadanie
String pytanie – treść zadania
String odpowiedz – prawidłowa odpowiedź
int wartosc – wartość zadania (ile punktów otrzyma gracz za poprawne rozwiązanie)
boolean czyRozwiazane – jeśli zadanie już zostało rozwiązane, czyRozwiazane przyjmuje wartość true

metody:
Zadanie(String n, Komnata k, String p, String o, int war) – konstruktor, tworzy zadanie o nazwie n, treści p, odpowiedzi o i wartości war i umieszcza je w komnacie k
void rozwiazanie() – wyświetla graczowi pytanie, pobiera od niego odpowiedź i porównuje z poprawną odpowiedzią (jeśli jest poprawna to zwiększa majątek gracza), wypisuje czy zadanie zostało poprawnie rozwiązane

INTERFEJS Polecenie

metody:
String nazwa() – zwraca nazwę polecenia
void wykonaj(Labirynt labirynt, String rozkazy[]) – wykonuje polecenie (rozkazy[] to tablica, w której znajduje się polecenie wpisane przez gracza)

KLASA Koniec

implementuje interfejs Polecenie

metody:
String nazwa() – zwraca napis ”koniec”
void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to ustawia koniecGry na true

KLASA Majatek

implementuje interfejs Polecenie

metody:
String nazwa() – zwraca napis ”majątek”
void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to wypisuje wartość majątku, jaki gracz posiada

KLASA Odloz

implementuje interfejs Polecenie

metody:
String nazwa() – zwraca napis ”odłóż”
void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli gracz posiada przedmiot, który chce odłożyć, to dany przedmiot jest przypisywany do komnaty, w której gracz się aktualnie znajduje, a pole wlasciciel tego przedmiotu jest ustawiane na null. Gracz jest informowany o tym jaki przedmiot odłożył i (w przypadku skarbu) ile był on wart. Wartość majątku gracza zostaje zmniejszona, jeśli odłożył skarb. Zostaje wywołana metoda odloz() z klasy Gracz.

KLASA Wez

implementuje interfejs Polecenie

metody:
String nazwa() – zwraca napis ”weź”
void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli w komnacie, w której obecnie znajduje się gracz jest przedmiot o podanej nazwie, to zostaje on przypisany do gracza, a pole komnata tego przedmiotu jest ustawiane na null. Gracz jest informowany o tym jaki przedmiot wziął i (w przypadku skarbu) ile jest on wart. Wartość majątku gracza zostaje zwiększona, jeśli wziął skarb. Zostaje wywołana metoda wez() z klasy Gracz.

KLASA Otworz

implementuje interfejs Polecenie

metody:
String nazwa() – zwraca napis ”otwórz”
void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli gracz posiada klucz do podanych drzwi i takie drzwi znajdują się w komnacie, w której gracz aktualnie się znajduje, to te drzwi zostają otwarte (wywołanie metody otworz() dla danych drzwi)

KLASA Zamknij

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”zamknij”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli gracz posiada klucz do podanych drzwi i takie drzwi znajdują się w komnacie, w której gracz aktualnie się znajduje, to te drzwi zostają zamknięte (wywołanie metody zamknij() dla danych drzwi).

KLASA Przejdz

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”przejdź”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli przejście dalej przez podane przejście bądź drzwi jest możliwe, to gracz zostaje przemieszczony (wywołanie metody przejdz() z klasy Gracz). Jeśli gracz chce wyjść z labiryntu (przez specjalne przejście o nazwie „wyjście”), to sprawdzane jest czy wartość jego majątku jest wystarczająca do zakończenia gry.

KLASA RozejrzyjSie

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”rozejrzyj”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to zostaje wyświetlony opis komnaty, w której gracz się znajduje oraz zostaje wypisane wszystko, co jest w tej komnacie (przedmioty, drzwi, przejścia, zadania) – wywołanie metody rozejrzyjSie() z klasy Gracz.

KLASA Rozwiaz

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”rozwiąż”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli podane przez gracza zadanie znajduje się w tej samej komnacie co gracz, to zostaje wyświetlona treść zadania oraz zostaje pobrana odpowiedź od gracza i porównana z prawidłową odpowiedzią. Zostaje wyświetlona informacja, czy zadanie zostało poprawnie rozwiązane. (wywołanie metody rozwiaz() z klasy Zadanie)

KLASA Sprawdz

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”sprawdź”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to jeśli dany przedmiot znajduje się w tej komnacie co gracz, to zostaje wyświetlony opis tego przedmiotu (wywołanie metody sprawdz() z klasy Przedmiot).

KLASA WypiszCoMam

implementuje interfejs Polecenie

metody:

String nazwa() – zwraca napis ”wypisz”

void wykonaj(Labirynt labirynt, String rozkazy[]) – jeśli polecenie zostało poprawnie wprowadzone, to zostają wypisane wszystkie przedmioty, które gracz posiada (wywołanie metody wypiszCoMam() z klasy Gracz).

KLASA Gra

pola:

Labirynt labirynt – labirynt, w którym odbywa się gra

Polecenie rozkazy[] – tablica poleceń, których gracz może używać

metody:

Gra(Labirynt l, Polecenie r[]) – konstruktor, tworzy grę w labiryncie l, z poleceniami z tablicy r[]

void graj() – rozpoczyna grę, wyświetla wszystkie informacje potrzebne graczowi do zrozumienia na czym polega gra, pobiera od gracza polecenia i je rozpoznaje (wywołuje metodę wykonaj() z odpowiedniej klasy implementującej interfejs Polecenie) oraz kończy grę w momencie, gdy pole koniecGry z klasy Labirynt ma wartość true.