Sprawozdanie  
PZWP

6 – Wzorce behawioralne

Jakub Ferens

Zadanie B1

Przeprojektowana klasa lew ze wzorcem State:

*class* LewContext:  
 *def \_\_init\_\_*(*self*):  
 *self*.jedzacy = LewJedzacy(*self*)  
 *self*.spacerujacy = LewSpacerujacy(*self*)  
 *self*.spiacy = LewSpiacy(*self*)  
 *self*.stan = *self*.spacerujacy  
  
 *def* nakarm(*self*):  
 *print*("Bierze się do jedzenia")  
 *self*.stan = *self*.jedzacy  
  
 *def* zaczep(*self*):  
 *self*.stan.zaczep()  
  
 *def* zaspiewaj(*self*):  
 *self*.stan.zaspiewaj()  
  
 *def* odbierz(*self*):  
 *self*.stan.odbierz()  
  
*class* LewInterface:  
 *def* zaczep(*self*):  
 *raise NotImplementedError  
  
 def* nakarm(*self*):  
 *raise NotImplementedError  
  
 def* zaspiewaj(*self*):  
 *raise NotImplementedError  
  
 def* odbierz(*self*):  
 *print*("Nie ma mu czego odebrać")  
  
*class* LewSpiacy(LewInterface):  
 *def \_\_init\_\_*(*self*, lewkontekst: LewContext):  
 *self*.lewkon = lewkontekst  
  
 *def* zaczep(*self*):  
 *print*("Nie reaguje")  
  
 *def* zaspiewaj(*self*):  
 *print*("Budzi się")  
 *self*.lewkon.stan = *self*.lewkon.spacerujacy  
  
  
*class* LewJedzacy(LewInterface):  
 *def \_\_init\_\_*(*self*, lewkontekst: LewContext):  
 *self*.lewkon = lewkontekst  
 *def* zaczep(*self*):  
 *print*("Warczy")  
  
 *def* zaspiewaj(*self*):  
 *print*("Warczy")  
  
 *def* jedzacy(*self*):  
 *print*("Odebrano mu jedzenie")  
 *self*.lewkon.stan = *self*.lewkon.spacerujacy  
  
  
*class* LewSpacerujacy(LewInterface):  
 *def \_\_init\_\_*(*self*, lewkontekst: LewContext):  
 *self*.lewkon = lewkontekst  
 *def* zaczep(*self*):  
 *print*("Rzuca się na kratę")  
  
 *def* zaspiewaj(*self*):  
 *print*("Zasypia")  
 *self*.lewkon.stan = *self*.lewkon.spiacy

Zadanie B2

Spróbuj rozwiązać ten sam problem unikając serii instrukcji if – za pomocą wzorca Visitor:

*from* \_\_future\_\_ *import* annotations  
*from* abc *import* ABC, abstractmethod  
  
*class* JednostkaPolicji(ABC):  
 @abstractmethod  
 *def* accept(*self*, V: Visitor) -> *None*:  
 *pass  
  
class* KomendaGlowna(JednostkaPolicji):  
 *def \_\_init\_\_*(*self*):  
 *self*.komendant = "Papała"  
 *self*.obszar = (200, 100)  
  
 *def* accept(*self*, V: Visitor) -> *None*:  
 V.visit\_komenda\_glowna(*self*)  
  
  
*class* Posterunek(JednostkaPolicji):  
 *def \_\_init\_\_*(*self*):  
 *self*.nazwiskoKomendanta = "Adamski"  
 *self*.dlugosc = 10  
 *self*.szerokosc = 20  
  
 *def* accept(*self*, V: Visitor) -> *None*:  
 V.visit\_posterunek(*self*)  
  
*class* ABW(JednostkaPolicji):  
 *def \_\_init\_\_*(*self*):  
 *self*.obszar = *None # Placeholder as in the original Java example  
  
 def* accept(*self*, V: Visitor) -> *None*:  
 V.visit\_abw(*self*)  
  
*class* Dzielnicowy(JednostkaPolicji):  
 *def \_\_init\_\_*(*self*):  
 *self*.nazwiskoDzielnicowego = "Kowalski"  
 *self*.powierzchnia = 100.0  
  
 *def* accept(*self*, V: Visitor) -> *None*:  
 V.visit\_dzielnicowy(*self*)  
  
  
*class* Test:  
 *def \_\_init\_\_*(*self*):  
 *self*.jednostki = []  
 *self*.jednostki.append(Dzielnicowy())  
 *self*.jednostki.append(Posterunek())  
 *self*.jednostki.append(KomendaGlowna())  
 *self*.jednostki.append(ABW())  
 *self*.odwiedzajacySzefow = VisitorSzefow()  
 *self*.odwiedzajacyPowierzchni = VisitorPowierzchni()  
  
 *def* wypiszSzefow(*self*):  
 *print*("Szefowie jednostek:")  
 *for* jednostka *in self*.jednostki:  
 jednostka.accept(*self*.odwiedzajacySzefow)  
  
 *def* wypiszPowierzchnie(*self*):  
 *print*("Powierzchnie jednostek:")  
 *for* jednostka *in self*.jednostki:  
 jednostka.accept(*self*.odwiedzajacyPowierzchni)  
  
*class* Visitor(ABC):  
 @abstractmethod  
 *def* visit\_dzielnicowy(*self*, jednostka: Dzielnicowy):  
 *pass* @abstractmethod  
 *def* visit\_posterunek(*self*, jednostka: Posterunek):  
 *pass* @abstractmethod  
 *def* visit\_komenda\_glowna(*self*, jednostka: KomendaGlowna):  
 *pass* @abstractmethod  
 *def* visit\_abw(*self*, jednostka: ABW):  
 *pass  
  
  
class* VisitorSzefow(Visitor):  
 *def* visit\_dzielnicowy(*self*, jednostka: Dzielnicowy):  
 *print*(f"Dzielnicowy: {jednostka.nazwiskoDzielnicowego}")  
  
 *def* visit\_posterunek(*self*, jednostka: Posterunek):  
 *print*(f"Posterunek: {jednostka.nazwiskoKomendanta}")  
  
 *def* visit\_komenda\_glowna(*self*, jednostka: KomendaGlowna):  
 *print*(f"KG: {jednostka.komendant}")  
  
 *def* visit\_abw(*self*, jednostka: ABW):  
 *print*("ABW: Nazwisko jest tajne!")  
  
  
*class* VisitorPowierzchni(Visitor):  
 *def* visit\_dzielnicowy(*self*, jednostka: Dzielnicowy):  
 *print*(f"Dzielnica: {jednostka.powierzchnia}")  
  
 *def* visit\_posterunek(*self*, jednostka: Posterunek):  
 *print*(f"Posterunek: {jednostka.dlugosc \* jednostka.szerokosc}")  
  
 *def* visit\_komenda\_glowna(*self*, jednostka: KomendaGlowna):  
 *print*(f"KG: {jednostka.obszar}")  
  
 *def* visit\_abw(*self*, jednostka: ABW):  
 *print*("ABW: Nazwisko jest tajne!")  
  
  
*# Example usage  
if* \_\_name\_\_ == "\_\_main\_\_":  
 test = Test()  
 test.wypiszSzefow()  
 test.wypiszPowierzchnie()