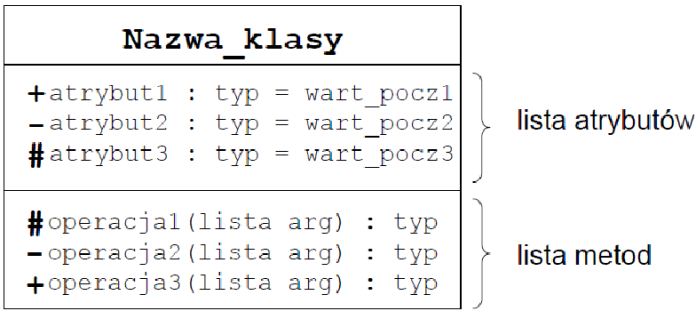
Inżynieria oprogramowania

LAB 6 UML – Diagramy klas

1. Diagram klas

Diagram klas jest ściśle powiązany z projektowaniem obiektowym systemu informatycznego lub wręcz bezpośrednio z jego implementacją w określonym języku programowania. Elementami tego diagramu są klasy, reprezentowane przez prostokąty, które mogą zawierać informację o polach i metodach klasy.

Klasa obrazowana jest za pomocą podzielonego na części prostokąta –każda część reprezentuje inwarianty o zbliżonym przeznaczeniu.



Rys.1. Klasa

Na diagramie klasy dodatkowo określa się widoczność atrybutów i metod przez umieszczanie przed nimi odpowiedniego symbolu:

- + publiczne (public)
- - prywatne (private)
- # chronione (protected)

Dobrze zbudowany diagram klas:

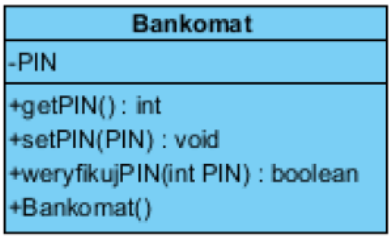
- uwypukla jeden statyczny aspekt perspektywy projektowej systemu
- obejmuje tylko byty niezbędne do zrozumienia tego aspektu
- zawiera szczegóły odpowiednie do przyjętego poziomu abstrakcji, z dodatkami koniecznymi do zrozumienia tego, na czym zależy projektantowi
- nie jest zbyt ogólny

ZAD.1.

Utworzyć diagram klas i zapoznać się z podstawowymi własnościami diagramu klas.

ZAD.2.

W diagramie klas utworzyć klasę przedstawioną poniżej

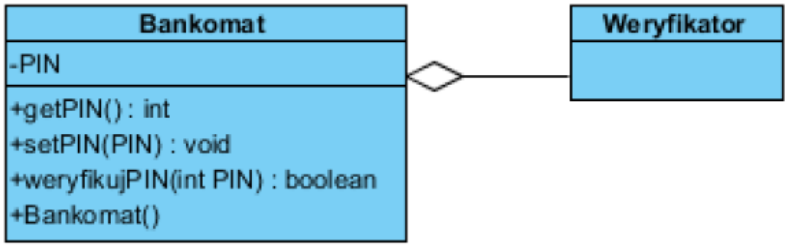


2. Agregacja

Agregacja najprościej mówiąc oznacza zawieranie.

Na przykład:
- lampka zawiera żarówkę,
- komputer zawiera procesor,
- jabłko zawiera robaka, itd.

Agregację na diagramie UML oznacza się pustym rombem.

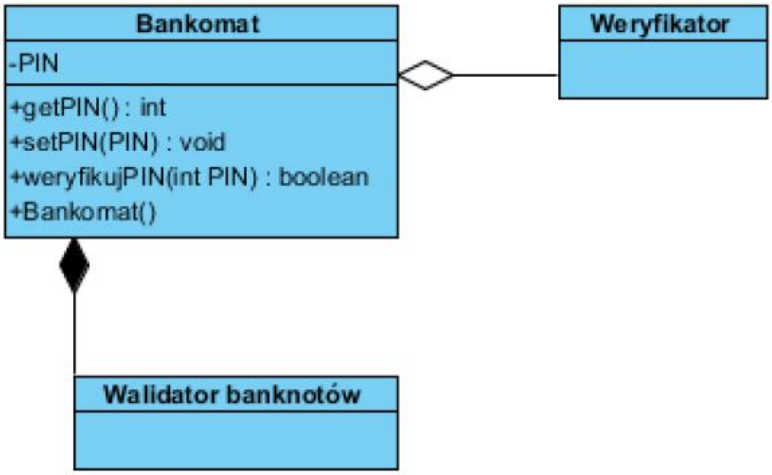


Rys.2. Agregacja

3. Kompozycja

Szczególnym przypadkiem agregacji jest kompozycja. Od agregacji różni się tym, że klasa posiada obiekty (składa się z obiektów), które bez tej klasy istnieć by nie mogły. Przykłady takich związków to:
- blok zawiera mieszkania (mieszkania poza blokiem nie istnieją),
- komputer zawiera procesor,
- łazienka zawiera wannę,

Kompozycja na diagramie UML oznaczona jest wypełnionym rombem.

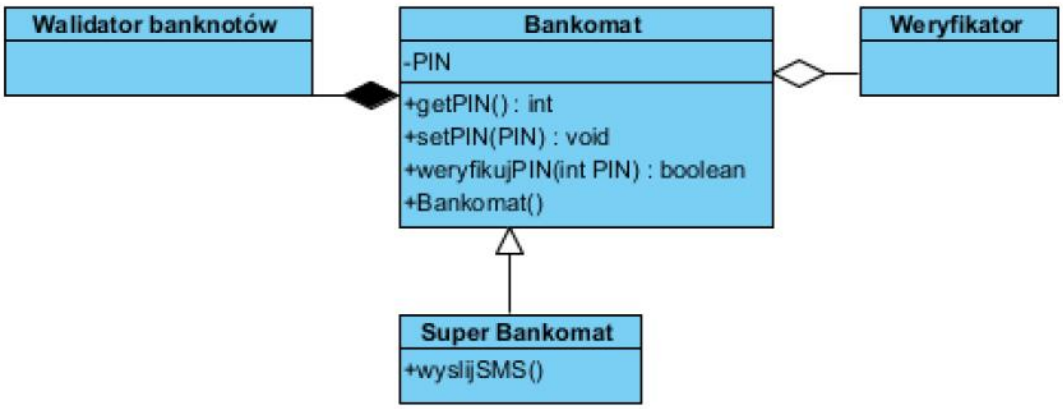


Rys.3. Kompozycja

4. Generalizacja

Generalizacja odpowiada dziedziczeniu znanemu z języków programowania. Inaczej mówiąc, jest to związek pomiędzy bardziej ogólną klasą (rodzicem) a klasą bardziej szczegółową (rodzicem).

Oto przykłady generalizacji:
- manager jest pracownikiem,
- sklep jest firmą,
- latarnia jest lampą, podobnie lampą jest lampka nocna,
- kot jest zwierzęciem,



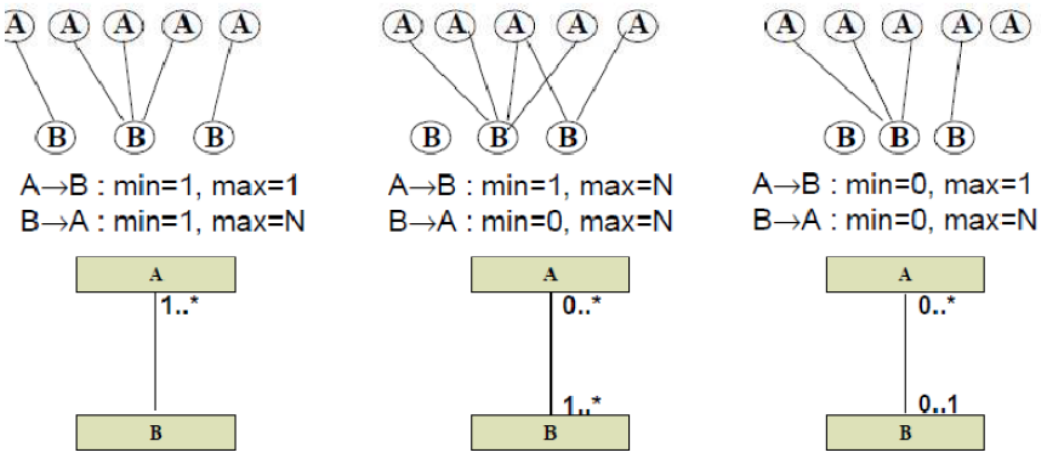
Rys.4. Generalizacja

ZAD.3.

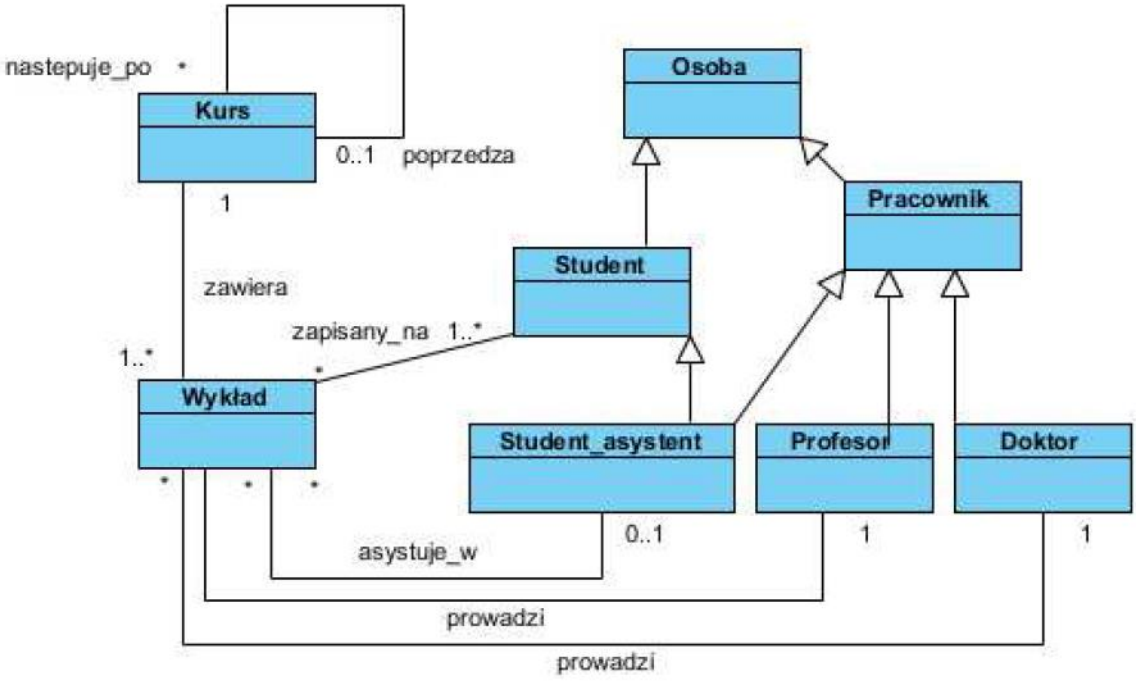
Utworzyć diagram przedstawiony na rys.4.

5. Asocjacje – liczność, nazwy ról

Asocjacje mogą być wyposażone w oznaczenia liczności

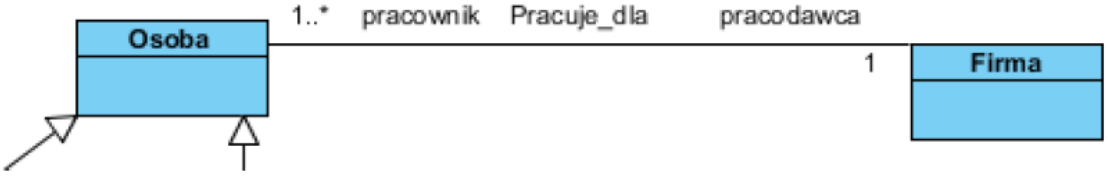


Liczność oznacza, ile obiektów innej klasy może być powiązane z jednym obiektem danej klasy (para liczb oznaczająca ilość minimalną i maksymalną).



Rys.5. Asocjacje – liczność

Asocjacje mogą być także wyposażone w dodatkowe nazwy ról (przy odpowiednich końcach).



Rys.6. Asocjacje - role

ZAD.4.

Narysować diagram klas przedstawiony na rysunku 5.

ZAD.5.

Utworzyć w programie NetBeans projekt programu zawierający przynajmniej 3 współpracujące ze sobą klasy, z których każda zawiera co najmniej 5 metod (w tym prywatne, publiczne i chronione).

Na podstawie utworzonego projektu narysować diagram klas obrazujący jego strukturę i powiązania w nim występujące.