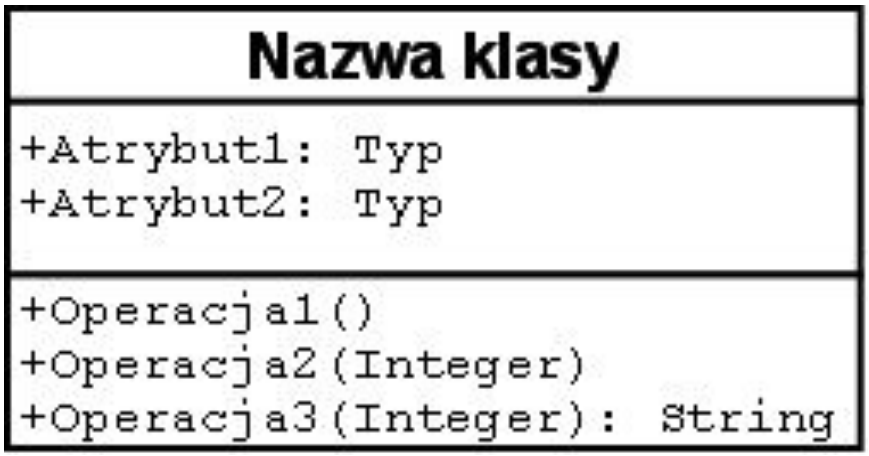
UML

Diagramy Klas

www.twinbottles.com/kop/Skrypt2.doc

Klasa

- Nazwa klasy znajduje się w górnym polu.
- W polu środkowym wymienione są atrybuty klasy.
- Ostatnie pole zawiera operacje klasy.
- Atrybuty oraz operacje mogą być widoczne dla innych klas (publiczne), tylko dla klas dziedziczących (chronione) lub tylko dla klasy w której się znajdują (prywatne).

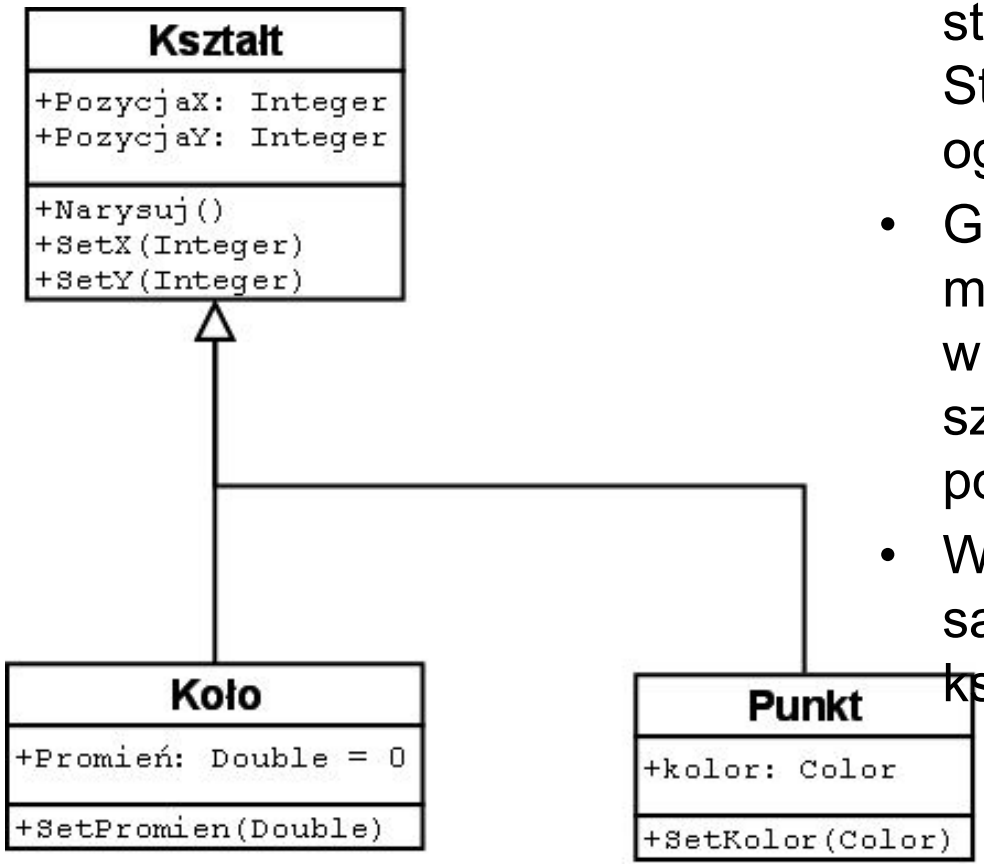


- Widoczność operacji lub atrybutów zaznaczmy znakiem ‘+’ (publiczne), ‘-’ (prywatne), lub ‘#’ (chronione) przed ich nazwami.

Klasa - Przykład

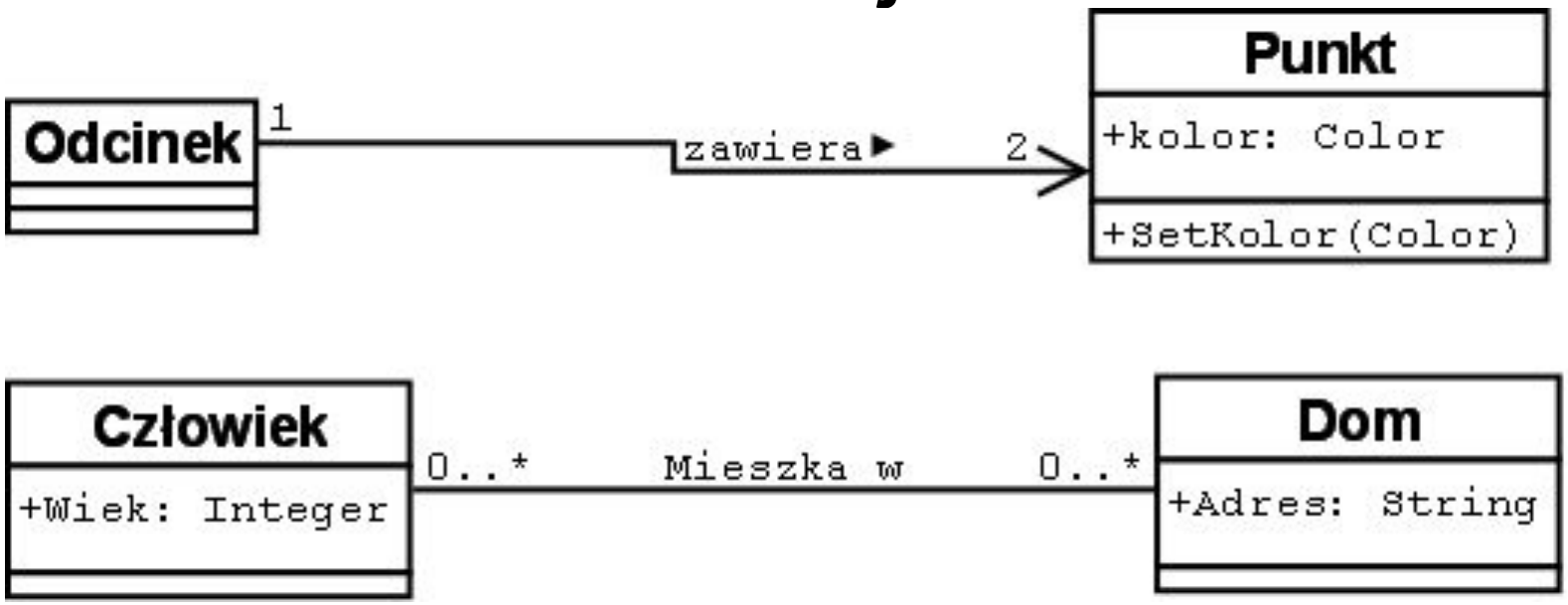
- Klasa kształt posiada atrybuty pozycja X oraz Y i kolor. Koordynaty będą liczbami całkowitymi natomiast kolor będzie obiektem klasy Color.
- Klasa ta posiada także trzy operacje. Operacja Narysuj nie przyjmuje żadnych argumentów, natomiast operacje SetX oraz SetY przyjmują po jednej liczbie całkowitej. Żadna z tych operacji nie ma zadeklarowanego typu.

Dziedziczenie - Generalizacja



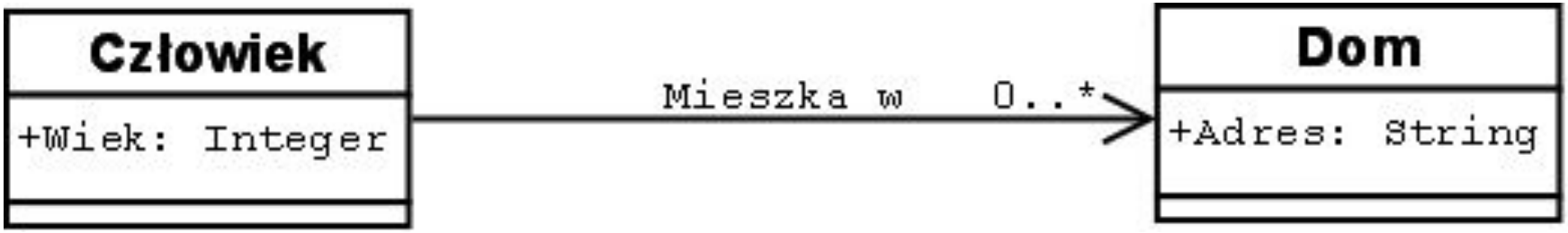
- Dziedziczenie zaznaczamy strzałką o pustym grocie. Strzałka wskazuje na klasę ogólniejszą.
- Grot jest jeden, jednak strzałka może się rozdzielać i łączyć większą ilość bardziej szczegółowych klas z klasą podstawową.
- W tym przypadku Koło i Punkt są wyspecjalizowanymi kształtami

Relacje



- Relacja zaznaczana jest linią.
- Relacje mają swoje nazwy, wypisane nad linią.
  - Zazwyczaj jest to czasownik określający rodzaj relacji.
- Przy końcach linii oznaczającej relację zapisane są ilości elementów znajdujących się w relacji. W tym przypadku jeden odcinek jest w relacji z dokładnie dwoma punktami (jego końcami). Natomiast Człowiek może mieszkać w dowolnej ilości domów (lub w żadnym). W jednym domu może mieszkać dowolna ilość ludzi lub może on stać pusty.

Relacje jednostronne

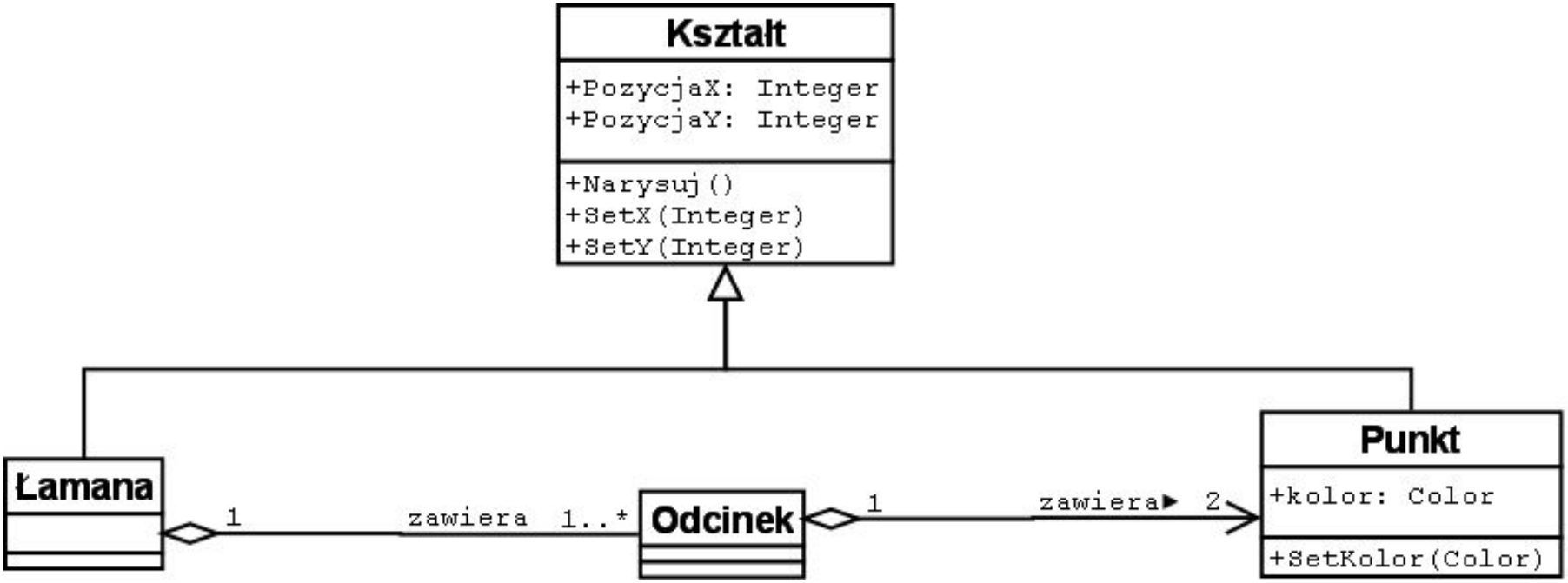


- Relacje mogą być zakończone strzałkami. Strzałka oznacza że człowiek mieszka w dowolnej ilości domów a domy nie wiedzą o istnieniu ludzi.
- W poprzednim przykładzie z ludźmi i domami domy trzymały informacje ile mieszka w nich osób. W tym przykładzie nie jest to potrzebne więc zaznaczamy że dom nie musi wiedzieć o istnieniu ludzi.
- Jest to relacja jednostronna między człowiekiem a domem.

Zapis liczności w relacjach

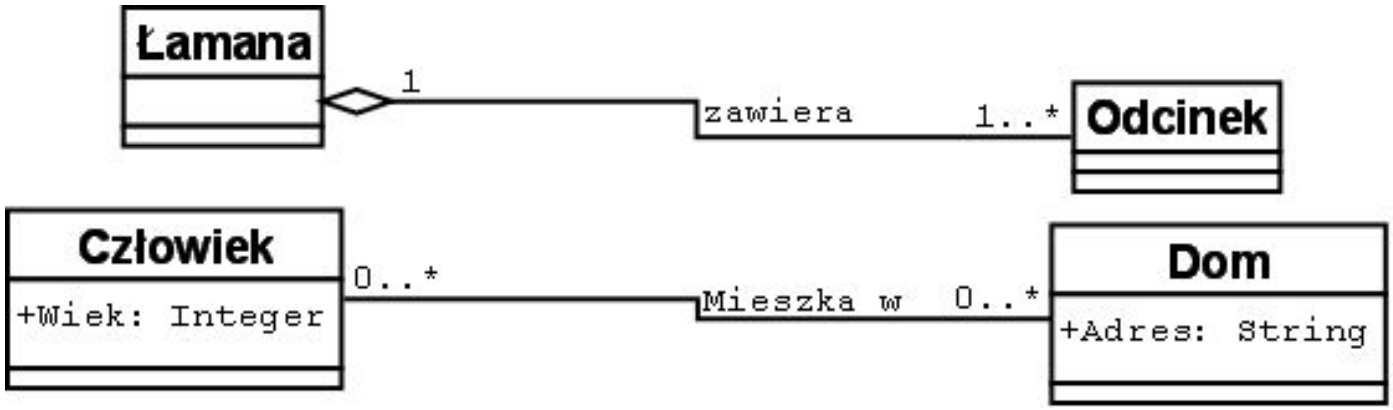
Zapis	Znaczenie
0..1	Zero lub jeden
1	Dokładnie jeden
0..*	Zero lub więcej
1..*	Jeden lub więcej
n	Dokładnie n (dla n>0)
0..n	Między zero a n (dla n>0)
1..n	Między jeden a n (dla n>0)

Agregacja



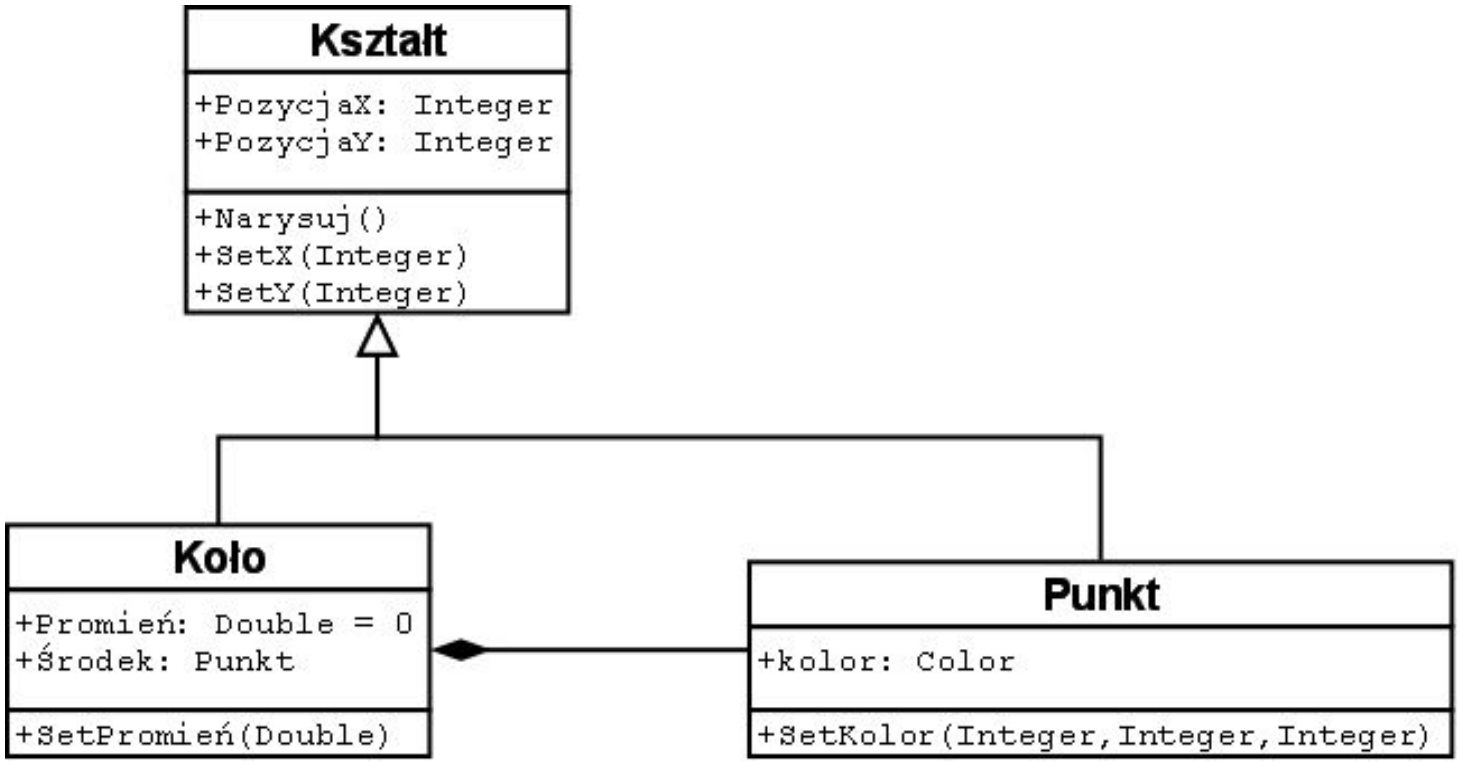
- Agregacja jest zapisywana za pomocą linii zakończonej pustym rombem. Jest to relacja typu Całość/Część. Oznacza ze Łamana jest pewną całością a odcinek stanowi część tej całości.
- Różnica pomiędzy agregacją a zwykłą relacją jest ulotna. Agregacja oznacza to samo co relacja z zaznaczeniem zależności typu Całość/Część.
- W tym przykładzie zakładamy że łamana składa się z co najmniej jednego odcinka a każdy odcinek składa się z dwóch punktów.

Agregacja a relacja



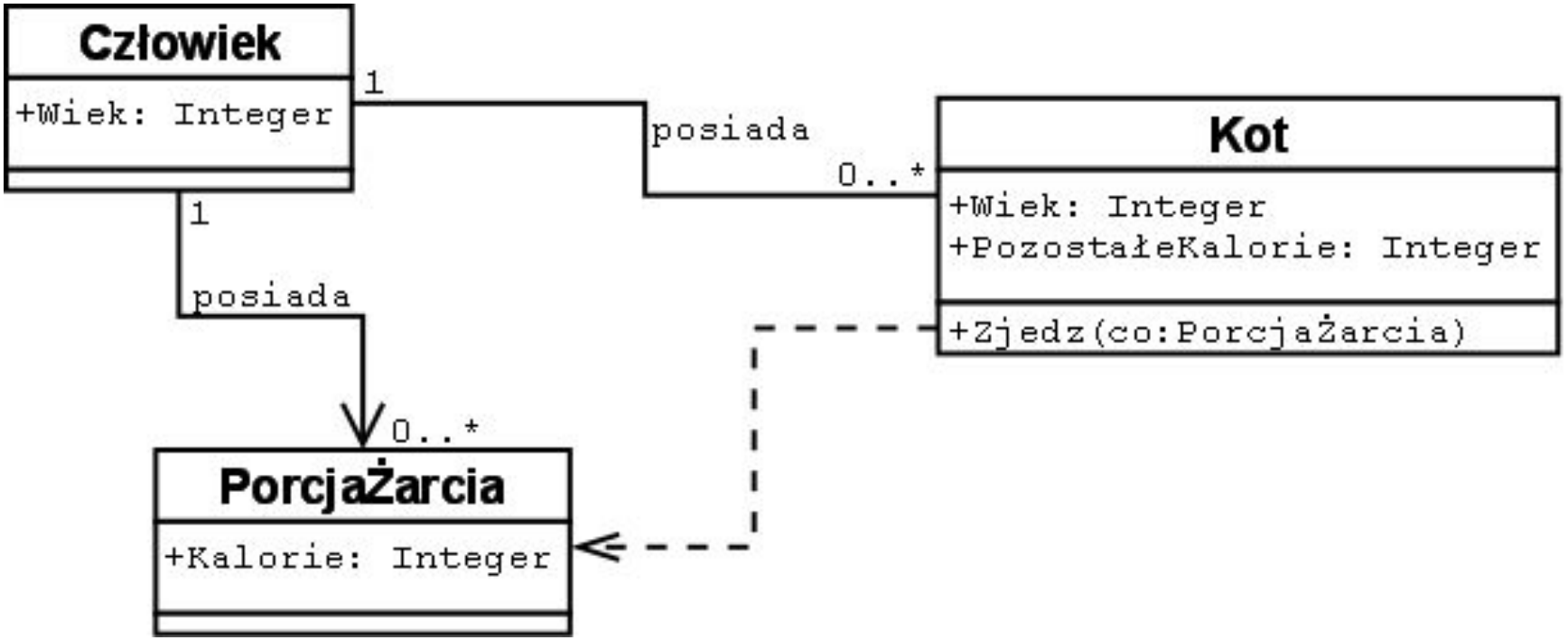
- Agregacja w przeciwieństwie do relacji opisuje zależność typu Całość/Część.
- Odcinki są elementami Łamanej.
- Ani dom nie jest częścią człowieka ani człowiek domu.
- Generalne zalecenie jest takie, żeby w przypadku wątpliwości używać relacji zamiast agregacji.

Kompozycja



- Zapisywana jako linia zakończona czarnym rombem po stronie klasy zawierającej
- Jest to mocna relacja Całość/Część. W tym przypadku Punkt jest elementem Koła.
- Kompozycja jako mocna relacja Całość/Część określa także czas istnienia elementów. Gdy Koło zostanie usunięte, usunięty zostanie także Punkt będący jego środkiem.

Zależności



- Zależność jest bardzo słabą relacją.
- Zależności zapisujemy przerywaną linią.
- W tym przypadku jedna z Operacji w klasie Kot przyjmuje jako argument obiekt klasy PorcjaŻarcia. Prawdopodobnie po to, aby sprawdzić ile kalorii kot zyska zjadając daną porcję.

Interfejsy

- Interfejs to klasa która nie posiada obiektów, jest tylko ogólnym typem. Nie istnieją Humanoidy jako takie, ale istnieje wiele istot, które są humanoidami.
- Nazwa takiej klasy jest pisana kursywą.
- Klasa taka nie może posiadać atrybutów a jej wszystkie funkcje muszą być wirtualne.
- Funkcje wirtualne są także zapisywane kursywą. Funkcje takie muszą być implementowane w klasach dziedziczących po klasie interfejsu.