PGO 05

Zadanie 1

Utwórz następujący system:

- Utwórz 3 klasy: Figura, Prostokąt i Koło.
- Klasa Figura definiuje pewną figurę geometryczną. Musimy tylko zapisać jej nazwę.
- Prostokąt ma również nazwę oraz 2 wartości: długość boku A i boku B.
- Oraz koło, dla którego oprócz nazwy wpisujemy promień.
- Wiedząc, że Prostokąt i Koło są Figurami, stwórz dziedziczenie między nimi, aby Prostokąt i Koło poszerzyły Figurę. Zwróć uwagę, co się teraz dzieje z atrybutem nazwa.
- Dla każdej klasy utwórz metody: getArea() i getPerimeter(), które obliczają i zwracają odpowiednio powierzchnię i obwód figur. W klasie Figura pole i obwód są 0.
  `Prostokąt => obwód: (a + b) * 2, powierzchnia: (a * b)`
  `Koło => obwód: 2 * PI * r, powierzchnia: PI * r ^ 2`
- W metodzie głównej, utwórz Listę figur i wypełnij ją obiektami klas Figure, Rectangle i Circle.
- Dla każdego z obiektów wywołaj metody getPerimeter() i getArea() i wydrukuj ich wynik wraz z nazwą figury.

Przykładowe dane wyjściowe.

```
Figury:
Prostokąt1: obwód = 10; powierzchnia = 6;
Koło1: obwód = 12,56; powierzchnia = 12,56;
Figura1: obwód = 0; powierzchnia = 0;
```

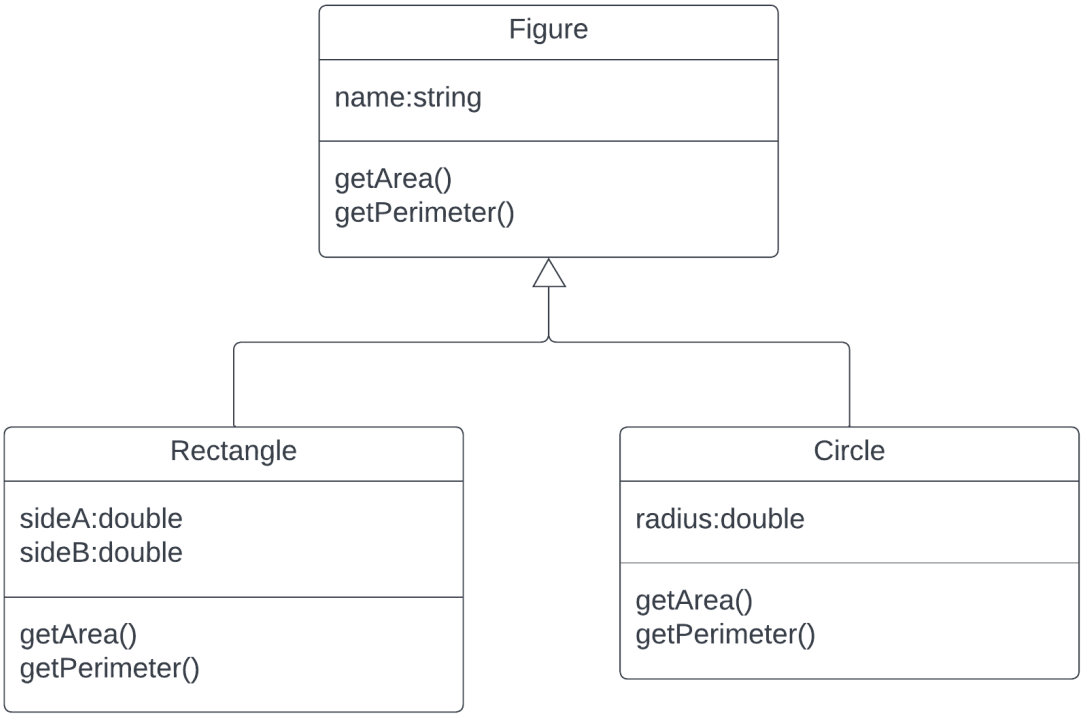
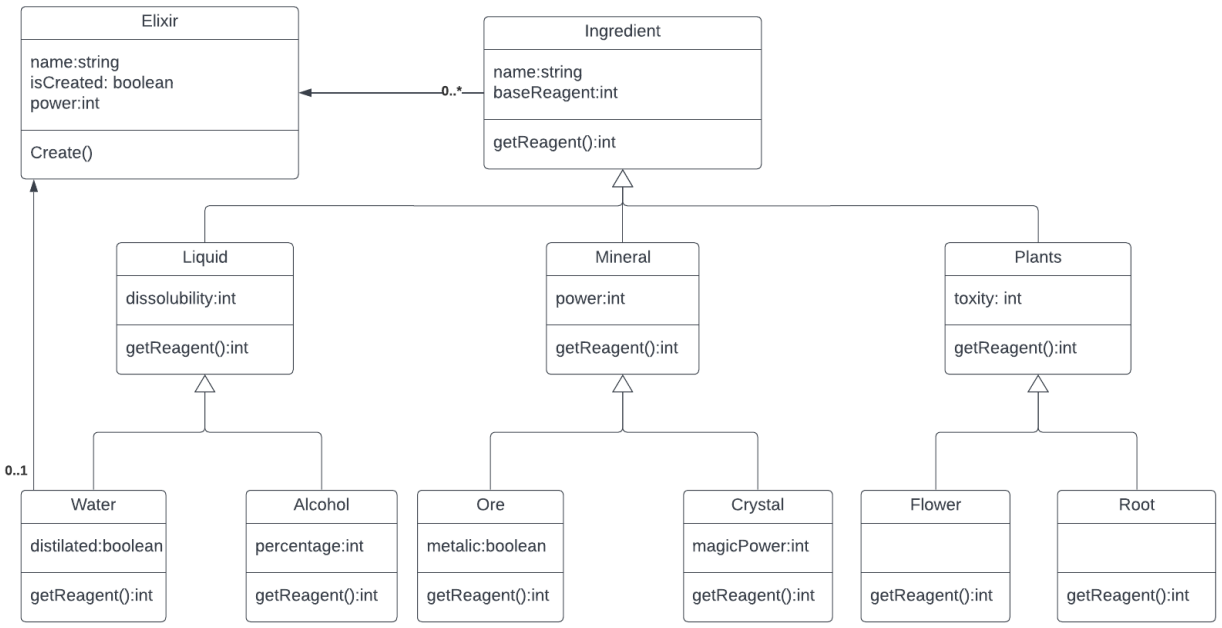


Figure
name:string
getArea()
getPerimeter()
Rectangle
sideA:double
sideB:double
getArea()
getPerimeter()
Circle
radius:double
getArea()
getPerimeter()

Zadanie 2 (20p)

Stwórz aplikację do symulacji reakcji alchemicznych.



Dla aplikacji musimy przechowywać następującą strukturę składników:

Ingredient

`Ingredient` class - przechowuje abstrakcyjny składnik, z następującymi możliwościami:

- **`String name`** - nazwa składnika
- **`int baseReagent`** - podstawowa wartość składnika alchemicznego
- **`int getReagent()`** - zwraca wartość reagentu

Składniki są podzielone na typy: Płynne, Minerały i Rośliny.

Liquid (Płyn)

`Liquid` class

- **`int dissolubility`** [od 0 do 100] - ilość jak dobrze katalizator rozpuszcza składniki
- **`int getReagent()`** - zwraca wartość reagentu w zależności od procentu rozpuszczalności

`Water(Woda)` - podklasa Liquid:

- **`boolean distilated:`** oznacza, czy woda jest destylowana
- **`int getReagent()`** - zwraca wartość reagentu, który pobiera wartość z klasy Liquid i dzieli wartość przez 2, jeśli woda nie była destylowana.

`Alcohol` - podklasa Liquid:

- **`int percent`** [od 0 do 100]: reprezentuje procent alkoholu w płynie

- **`int getReagent()`** - zwraca wartość reagentu, który pobiera wartość z klasy Liquid i zmniejsza otrzymaną wartość zgodnie z percentem.

Minerals

`Mineral class`:

- **`int power`** - moc minerału
- **`int getReagent()`** - zwraca wartość baseReagent z Ingredient i dodaje do niej power

`Ore` - podklasa Mineral:

- **`boolean metallic`**: określa czy ruda jest metaliczna czy nie
- **`int getReagent( )`** - zwraca wartość reagentu, który pobiera wartość z klasy Mineral i zmniejsza ją w 2 razy, jeśli ruda nie jest metaliczna.

`Crystal` - podklasa Mineral:

- **`int magicPower`**: reprezentuje magiczną moc zawartą w krysztale
- **`int getReagent()`** - zwraca wartość reagentu, który pobiera wartość z klasy Mineral i dodaje magicPower.

Plants

`Plant class`:

- **`int toxity`** - jak toksyczna jest roślina
- **`int getReagent()`** - zwraca wartość baseReagent ze składnika i mnoży ją przez toksyczność.

`Flower` - podklasa Plant:

- **`int getReagent()`** - zwraca wartość reagentu, który pobiera wartość z klasy Plant i mnoży wartość przez 2

`Root` - podklasa Plant:

- **`int getReagent()`** - zwraca wartość reagentu, który przyjmuje wartość z klasy Plant i dzieli wartość przez 2

Eliksir

`Eliksir class` :

- **`String name`** : nazwa eliksiru (tylko ta zmienna jest potrzebna do konstruktora)
- **`boolean isCreated`** : określa, czy eliksir został utworzony, czy nie. Nie można zmienić nigdzie, tylko jeden raz w metodzie Create. Wartość domyślna to false.
- **`int power`**: reprezentuje moc eliksiru (dostępne tylko wtedy, gdy isCreated ma wartość false)
- **`List<Ingredient> ingredients`** : lista składników w eliksirze
- **`void addIngredient(Ingredient)`** : dodaje składnik do listy (nie można dodać, jeśli eliksir został już utworzony)

- **`void removeIngredient( Ingredient)`**: usuwa składnik z listy (nie można go usunąć, jeśli eliksir został już utworzony)
- **`Liquid catalyst:`** reprezentuje katalizator eliksiru (nie można go zmienić, jeśli eliksir został już utworzony)
- **`void Create()`** : tworzy eliksir. isCreated staje true. Moc wszystkich składników jest obliczana i przechowywana w polu power i dzieli się przez wartość reagentu katalizatora (catalyst).

Powinna być klassa **`Main`**, która demonstruje działającą funkcjonalność systemu.
Bez klasy Main rozwiązanie **nie będzie ocenione**.

Pamiętaj, że wszystkie atrybuty powinny być private (lub protected w razie potrzeby), dlatego należy zaimplementować funkcjonalność seterów i getterów z odpowiednimi sprawdzeniami.