Упражнение 3: Обработка на поведението на агента

Следващият пример демонстрира създаване на агент, който имплементира поведение от типа *SimpleBehaviour*. За целта е необходимо да се пренапишат методите *action*() и *done*():

```
public class SimpleBehaviourAgent extends Agent {

	public SimpleBehaviourAgent() {
	}

	protected void setup()
	{
		addBehaviour(new MySimpleBehaviour( this ));
	}

	class MySimpleBehaviour extends SimpleBehaviour {
		private boolean finished = true;
		private int round = 0;

		public MySimpleBehaviour(Agent agent)
		{
			super(agent);
		}

		@Override
		public void action() {
		  System.out.println("Inside ACTION:: " + round++);
		}

		@Override
		public boolean done() {
			return finished;
		}
	}
}
```

Дефинирани са две променливи: *finished*, която има стойност *true* и *round*, която си увеличава стойността с единица всеки път, когато се изпълнява метода *action*().

Резултатът от изпълнението на горния пример ще е „Inside ACTION:: 0”, защото методът *action*() се изпълнява само един път тъй като методът *done*() връща стойност *true*.

Задача 1:

Модифицирайте задачата 3 от упражнение 2 така, че да се дефинира поведение на агента *MealAgent* за обхождане и претърсване на колекцията с рецепти и намиране на съвпадение.

Задача 2:

Създайте софтуерен агент с поведение, който организира брояч, увеличаващ се с единица всеки път, когато се изпълнява поведението на агента, докато брояча недостигне стойност 20. Това се реализира чрез имплементирането на две различни поведения от типа *CyclicBehaviour*. Първото поведение проверява дали брояча е със стойност 10 и ако е изпълнено условието прекратява работата на агента. Второто поведение разпечатва стойността на брояча и го увеличава с единица.