Die Aufgaben könnt ihr nur machen, wenn ihr Aufgabe 1 und Aufgabe 2 fertig gestellt habt!

Aufgabe 3 – Implementierung der Gegner- und Rätselwahl

Schwierigkeit **: Damit die Spieldynamik im Main-Programm vernünftig gegliedert werden kann, müssen zwei wesentliche Funktionen in eigene Methoden ausgelagert werden: Die Auswahl des Gegners und die Auswahl der Rätsel. Schreibe zwei solche Methoden, welche entweder zufällig ein Rätsel oder Gegner auswählt oder aber einen festen Gegner oder festes Rätsel wählt. Verwendet zur Fallunterscheidung eine switch-case-Anweisung.

Aufgabe 4 – Implementierung der Rätsel

Unser Dungeon Crawl hat unterschiedliche Quest. Diese müssen für unser Spiel entsprechend implementiert werden.

Schwierigkeit **:

1. Wir verwenden das Konzept der abstrakten Klasse, um uns unsere Klasse *Mystery* als Vorlage für unsere tatsächlichen Rätsel zu gestalten. Überlege dir welche Attribute, welchen Konstruktor, welche abstrakten Methoden und welche Methoden du benötigst und implementiere dies.
2. Ergänze nun die einzelnen Rätsel, indem jedes neue Rätsel von *Mystery* erbt. Denke daran, was du bei der Vererbung von abstrakten Klassen beachten musst.

Aufgabe 4 – Implementierung der Spieler und Gegner

Damit wir auch verschiedene Gegner haben und selbst auch als Spieler fungieren können, benötigen wir hierfür jeweils verschieden Objekte der Gegner und Spieler. Hierfür müssen wir die verschiedenen Klassen konstruieren.

Schwierigkeit **:

1. Wir verwenden das Konzept der abstrakten Klasse, um uns unsere Klasse *Figure* als Vorlage für unsere tatsächlichen Charaktere bzw. Spieler zu gestalten. Überlege dir welche Attribute, welchen Konstruktor, welche abstrakten Methoden und welche Methoden du benötigst und implementiere dies.
2. Als nächstes benötigen wir einen Spieler. Die Klasse *Player* muss mit extends von *Figure* erben. Überlege dir, wie der Konstruktor aufgebaut werden muss und wie sich bei einem Schaden (ist der Wert des Angriffs) die Gesundheit reduziert.
3. Lass auch deinen Gegner (*Enemy*) von deiner Klasse *Figure* erben. Überlege dir, welche abstrakte Methode benötigt werden könnte, und überschreibe die Methode für den Angriff passend.
4. Jetzt fehlen uns noch die konkreten Gegner. Jeder der Gegner erbt von *Enemy*. Mögliche Gegener können bspw. die Venom creature, der fire dragon, der ghost warrior, das shadow golem und der Titan sein. Alle müssen in ihrem

Konstruktor ihren Namen, einen ausgewählten Gesundheitsstand und einen festen Wert für ihren Angriff übergeben. Die Methode specialSkill(Player player) muss passend überschrieben werden. Es muss eine Information für den Spieler als Konsolenoutput generiert werden und eine Veränderung des Gesundheitszustands des Gegners oder des übergeben Players erfolgen.

Aufgabe 4 – Implementierung des Spielablaufs

Schwierigkeit **:

Für die Implementierung des Spielablaufs brauchst du deinen Entscheidungsbaum, die Main-Methode in *DungenCrawl*, einen *Scanner* und einen *Player*.

Mit Hilfe von verschachtelten if-else-Unterscheidungen kannst du nun den Spielverlauf erstellen.

Wichtig: Denke dran deinem Spieler ein Storytelling zu geben.

Aufgabe 5 – Spiele dein Spiel

Aufgabe 6 – Implementierung von Variationen im Spielablauf

Du kannst nun deine Implementierung des Spiels so anpassen, dass der Spielablauf nach deinen Wünschen geregelt wird. Eine Möglichkeit wäre z.B. noch weitere Rätsel mit einzubeziehen.