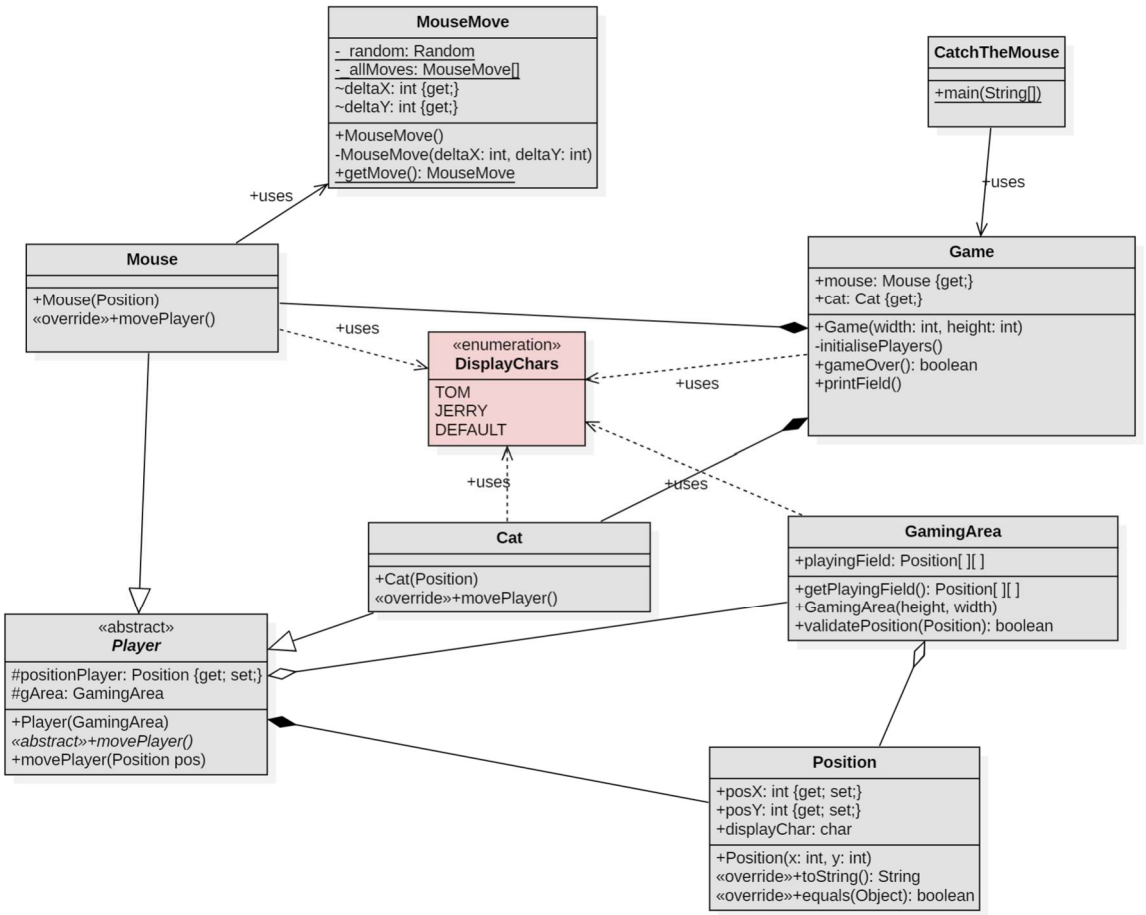
Projekt *Catch The Mouse*

AUFGABENSTELLUNG:

In dieser Aufgabenstellung erstellen wir ein Spiel, das unter dem Namen *Catch The Mouse* bekannt ist. Auf einem 10x10 Spielfeld bewegen sich eine Maus und eine Katze. Die Maus wird durch das Programm zufällig bewegt, die Katze wird vom Spieler gesteuert. Ziel ist es, die Maus zu fangen. Die Maus kann sich genau ein Feld (vertikal, horizontal oder diagonal) bewegen, die Katze kann beliebige Bewegungen machen.

Sprint 1

Aufgabe 1 – Spiellogik: Zunächst programmieren wir die Spiellogik gemäß dem folgenden Klassendiagramm:



Zur einfachen (und zentralen) Definition der Spielfeldinhalte verwenden wir ein enum – allerdings ein „enum of chars“ (wir wollen ja für die Spielfiguren und den Hintergrund jeweils ein char zeichnen).

Um das zu erreichen müssen wir die bisher bekannte Definition/Verwendung eines enum erweitern – finde selbständig heraus wie das funktioniert 😊

Aufgabe 2 – User Interface & Unit Tests:

Zunächst sind (zumindest) für die Klassen **Game** und **GamingArea** Unit Tests zu erstellen, um die korrekte Funktionalität der einzelnen Methoden zu überprüfen.

Danach ist ein einfaches (nicht-grafisches) User Interface (in der Klasse **CatchTheMouse**) zu programmieren. Darin soll das Spiel initialisiert (d.h. Spielfeld angelegt sowie Maus und Katze korrekt positioniert werden) und (mittels der Methode PrintField() ausgegeben werden. Dabei sind für Maus und Katze im enum entsprechende Symbole zu wählen, zB

```
Help Tom to catch Jerry :-)

0   . . . . . .
1   . . . . . .
2   C . . . . . .
3   . . . . . .
4   . . . . M . . . . .
5   . . . . . .
6   . . . . . .
7   . . . . . .
8   . . . . . .
9   . . . . . .
    0 1 2 3 4 5 6 7 8 9


Tom moves to (line-column):
```

Der Benutzer kann nunmehr die neue Position der Maus als Koordinate eingeben, zB

```
Tom moves to (line-column): 5-5



Help Tom to catch Jerry :-)

0   . . . . . .
1   . . . . . .
2   . . . . . .
3   . . . M . . .
4   . . . . . .
5   . . . . C . .
6   . . . . . .
7   . . . . . .
8   . . . . . .
9   . . . . . .
    0 1 2 3 4 5 6 7 8 9


Tom moves to (line-column):
```

Die Katze wird auf das neue Feld bewegt, die Maus verändert davor ebenfalls ihre Position (hier nicht abgebildet) – und es wird anschließend überprüft (und ausgegeben !) falls die Katze die Maus gefangen hat.

Falls kein korrekter Spielzug für die Katze eingegeben worden ist (leere oder falsche Eingabe), wird das Spiel unverzüglich beendet.

Falls die Maus gefangen worden ist, soll der Benutzer die Möglichkeit erhalten eine neue Spielrunde zu starten (dazu muss natürlich das Spielfeld neu aufgebaut werden).

Sprint 2

Aufgabe 3: Schlaue Maus

Zunächst wollen wir die Maus in die Lage versetzen, sich nach Beginn des Spieles nur nach einer definierten Anzahl von Spielzügen – z.B. 3 - zu zeigen (und ansonsten zu verstecken).

Aufgabe 4: Das Protokoll

Zur Kontrolle des korrekten Spielverlaufes ist eine Protokolldatei im Unterordner *protokolle* zu erstellen (im Namen der Datei soll das aktuelle Datum sowie die Beginnzeit des Spieles enthalten sein – d.h. für jedes Spiel wird eine eigene Datei angelegt).

Der Dateiname soll damit wie folgt gestaltet werden:

- DD-MM-YYYY_HHMMSS.txt, also zB 20-10-2023_181535.txt

Für jeden Spielzug sind die Nummer der Spielrunde sowie die Positionen von Katze und Maus vor und nach dem Spielzug sowie ein allfälliges Spielende zu speichern, zB

Runde 1:
Katze zieht von (3,4) auf (5,5)
Maus zieht von (5,4) auf (4,4)

usw.

Aufgabe 5: Graphische Benutzeroberfläche mit AWT/SWING

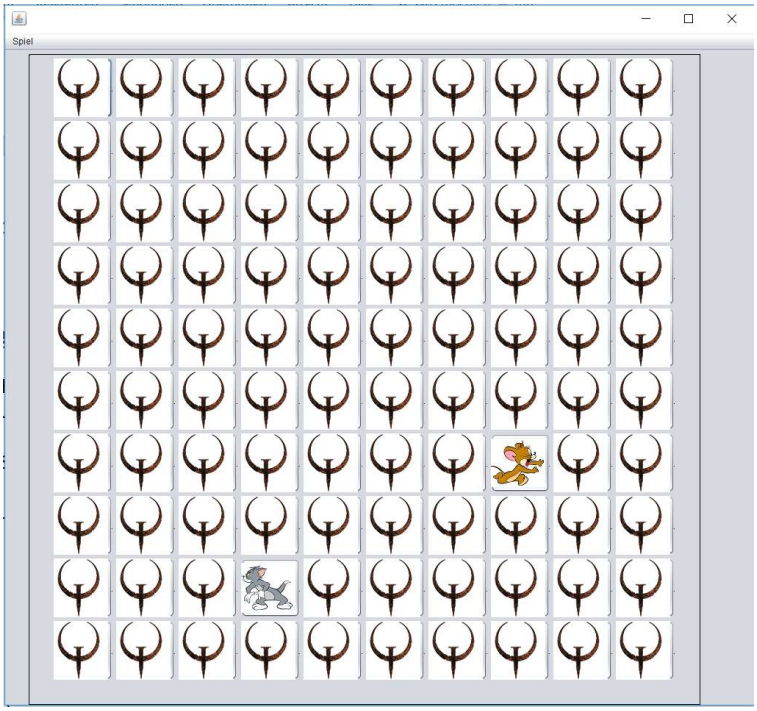
Diese Aufgabe darfst/sollst du mit Hilfe der KI Deines Vertrauens lösen – es gibt allerdings 2 Regeln:

- ***Die grafische Oberfläche muss die bisher programmierten Klassen deines Projektes verwenden***
- ***Du musst in der Lage sein, jede Zeile generierten Codes zu erklären.***

Erstelle mit AWT/SWING ein Formular mit dem 2D Spielfeld (10x10 Buttons). Die bisher verwendeten Klassen und der Spielverlauf bleiben unverändert, lediglich die Spiellogik (aus der Main Klasse) wandert nun ins Formular (zB in die Event-Handling Methoden der Buttons 1).

Die Benutzereingabe erfolgt nun nicht mehr durch Eingabe der neuen Koordinaten für die Katze (als Zahlen) sondern durch Klicken des entsprechenden (Ziel-)Buttons. Das Ende des Spieles (d.h. die Katze hat die Maus gefangen) soll ebenfalls durch eine grafische Komponente angezeigt werden.

Das Formular (zu Beginn des Spieles) könnte beispielsweise so aussehen:



[1] Verwende **eine Methode** als Event Handler für alle Buttons!!

Für die grafische Darstellung von Buttonhintergrund sowie Katze und Maus suche im Internet nach passenden Grafiken – du benötigst Bilder für

- Katze
- Maus
- Hintergrund
- Katze fängt Maus