SWO3	**Übung zu Softwareentwicklung mit klassischen Sprachen und Bibliotheken 3**	**WS 2019/20, ÜZ 8**

☐ Gruppe M. Hava

☒ Gruppe J. Heinzelreiter Name: Neuhold Michael Aufwand [h]: _____

☐ Gruppe P. Kulczycki Feedback von: _____

Beispiel	Lösungsidee (max. 100%)	Implement. (max. 100%)	Testen (max. 100%)
1 (25 P + 60 P + 15 P)			

Beispiel 1: Schach (`src/chess/`)

Implementieren Sie in C++ unter Anwendung des objektorientierten Programmierparadigmas eine einfache Version des Brettspiels „Schach“. Beachten Sie dabei die folgenden Anforderungen und Hinweise.

Für alle Spielfiguren gibt es eine gemeinsame, abstrakte Basisklasse `chessman`, die eine Schnittstelle für mindestens die folgenden Funktionalitäten bietet:

1. sie liefert die Farbe einer Spielfigur,
2. sie liefert eine symbolische Darstellung einer Spielfigur in Form eines ASCII-Zeichens,
3. sie beantwortet die Frage, ob eine Spielfigur „essentiell“ ist (ein Verlust einer essentiellen Spielfigur bedeutet das Ende des Spiels) und
4. sie beantwortet die Frage, ob sich eine Spielfigur auf einem Schachbrett von Position $\{z_0, s_0\}$ auf Position $\{z_1, s_1\}$ bewegen kann.

Alle konkreten Spielfiguren (wie z.B. Bauer und Springer) sollen dann, insbesondere für die letzte Funktionalität, eigene Implementierungen besitzen. Erstellen Sie dazu von `chessman` abgeleitete Klassen für die typischen Schachfiguren.

1. Der König („king“ – K) kann sich in alle Richtungen um ein Feld bewegen. Das Zielfeld darf aber nicht durch eine Spielfigur der eigenen Farbe besetzt sein. Der König ist eine essentielle Spielfigur.
2. Die Dame („queen“ – Q) kann sich beliebig weit diagonal, horizontal oder vertikal auf dem Spielbett bewegen, solange keine andere Spielfigur im Weg ist.
3. Der Läufer („bishop“ – B) kann sich beliebig weit diagonal auf dem Spielbett bewegen, solange keine andere Spielfigur im Weg ist.
4. Der Turm („rook“ – R) kann sich beliebig weit horizontal oder vertikal auf dem Spielbett bewegen, solange keine andere Spielfigur im Weg ist.
5. Der Springer („knight“ – N) kann sich um jeweils zwei Felder in eine Richtung (horizontal oder vertikal) und gleichzeitig um ein Feld in die andere Richtung (vertikal oder horizontal) bewegen. Dabei kann er beliebig andere Spielfiguren überspringen.
6. Der Bauer („pawn“ – P) kann sich um ein Feld nach vor bewegen. Bei seinem ersten Zug können dies auch zwei Felder nach vor sein. Der Bauer kann nur seitlich schlagen.

Alle Spielfiguren können auf einer feindlichen Spielfigur landen, was diese aus dem Spiel befördert und zum eigenen Sieg beitragen kann. Die Spielfiguren befinden sich auf einem Spielbrett was z.B. mit einer Klasse `chessboard` implementiert werden kann. Dieses Spielfeld muss mindestens die folgenden Funktionalitäten aufweisen:

1. Es muss vorgesehen sein, ein beliebig großes Spielbrett anlegen zu können (jedoch mit einer sinnvollen Mindestgröße).
2. Es muss erkannt werden, wann ein Spiel beendet ist (nämlich dann, wenn einer der Spieler eine essentielle Figur verloren hat).

3. Für jede Position muss die darauf befindliche Figur ermittelt werden können.
4. Der aktuellen Spieler sowie die Größe des Spielfeldes müssen ebenfalls abgefragt werden können.
5. Die folgenden Fragen müssen für jedes Feld des Spielbretts beantwortet werden können:
   a. Kann es „befahren“ werden bzw. ist es frei?
   b. Kann eine Spielfigur mit einer bestimmten Farbe geschlagen werden?
   c. Kann eine Spielfigur einer bestimmten Farbe darauf landen? Ist es also frei oder kann dort eine andere Figur geschlagen werden?
   d. Kann man auf diesem Feld eine Figur momentan „in die Hand nehmen“? Also ist dort eine Figur und wenn ja, von der aktuell am Zug befindlichen Farbe?
   e. Kann man die aktuell aktive („in der Hand befindliche“) Figur an diesem Feld abstellen? Darf die Figur also von ihrem aktuellen Startpunkt aus dort hinfahren und kann auch dortbleiben?
6. Kann man von einem Feld $\{z_0, s_0\}$ auf ein Feld $\{z_1, s_1\}$ fahren?
7. Folgende Aktionen müssen außerdem möglich sein:
   a. Die Spielfigur an einer bestimmten Stelle aktivieren, also „in die Hand nehmen“.
   b. Die aktive Spielfigur auf einem bestimmten Feld wieder abstellen (sofern das erlaubt ist).
8. Der aktuelle Spielstand (das Spielbrett mit den Spielfiguren) muss, genau so wie unten gezeigt, als ASCII-Grafik auf der Konsole ausgegeben werden.

Erledigen Sie nun die folgenden Aufgabenstellungen:

(a) Implementieren Sie alle klassischen Schachfiguren und testen Sie deren Bewegungsmuster ausführlich. Gehen Sie in der Lösungsidee auch darauf ein, wieviel Speicher pro Spielfigur benötigt wird.

(b) Schreiben Sie ein Hauptprogramm, mit dem zwei Spieler interaktiv Schach spielen können. Dabei müssen alle Zugmöglichkeiten der jeweils aktiven Spielfigur angezeigt werden.

Zwei Beispiele: Im linken Beispiel ist der weiße Springer (N) in Zeile 1, Spalte b aktiv. Im rechten Beispiel ist die die weiße Dame (Q) in Zeile 1, Spalte d, die einen gegnerischen schwarzen Bauern [p] in Zeile 4, Spalte d schlagen könnte, aktiv.

```
  | a  b  c  d  e  f  g  h |
--+-----------------------+--
8 | r  n  b  q  k  b  n  r | 8
7 | p  p  p  p  p  p  p  p | 7
6 | .  *  .  *  .  *  .  * | 6
5 | *  .  *  .  *  .  *  . | 5
4 | .  *  .  *  .  *  .  * | 4
3 |[*] .  [*] .  *  .  *  . | 3
2 | P  P  P  P  P  P  P  P | 2
1 | R (N) B  Q  K  B  N  R | 1
--+-----------------------+--
  | a  b  c  d  e  f  g  h |
```

```
  | a  b  c  d  e  f  g  h |
--+-----------------------+--
8 | r  n  b  q  k  b  n  r | 8
7 | p  p  p  p  *  p  p  p | 7
6 | .  *  .  *  .  *  .  * | 6
5 | *  .  *  .  *  .  * [.]| 5
4 |[.] *  . [p] .  * [.] * | 4
3 | * [.] P [.] P [.] *  . | 3
2 | P  P [.][*][.] P  P  P | 2
1 | R  N  B (Q) K  B  N  R | 1
--+-----------------------+--
  | a  b  c  d  e  f  g  h |
```

Abbildung: Die unterschiedlichen Farben der Spielfiguren sind als Groß- bzw. Kleinbuchstaben, die unterschiedlichen Farben der Felder durch Punkt bzw. Stern dargestellt. Die aktive Spielfigur ist in runde Klammern gesetzt. Die Felder, die die aktive Spielfigur betreten kann, sind in eckige Klammern gesetzt.

(c) Schreiben Sie ein alternatives Hauptprogramm (oder eine über die Kommandozeile auswählbare Variante), das bzw. die es erlaubt, einen zufälligen, und natürlich nur aus gültigen Zügen bestehenden, Spielablauf zu generieren.