abstract class Game {
    String name;
    int numPlayers;
    Game(String name, int numPlayers) {
        this.name = name;
        this.numPlayers = numPlayers;
    }
    String getName() { return name; }
    int getNumPlayers() { return numPlayers; }
    boolean hasBoard() { return false; }
    public String toString() {
        String result = "game " + getName() + " for " + getNumPlayers();
        if (hasBoard())
            return "the board " + result;
        else
            return "the " + result;
    }
}

class Checkers extends Game {
    Checkers() {
        super("checkers", 2);
    }
    String firstPlayer() { return "black"; }
    boolean hasBoard() { return true; }
    public String toString() {
        return super.toString() + " starting with " + firstPlayer();
    }
}

class Chess extends Checkers {
    public String getName() { return "chess"; }
    public String firstPlayer() { return "white"; }
}

class Solitaire extends Game {
    Solitaire() {
        super("solitaire", 1);
    }
}

1. In the box, draw a class hierarchy for the above classes.

2. In the following code, put an "X" to the left of each line that would result in a **compilation error**, and put two stars ("**") to the left of each line that would compile but would result in a **runtime error**. In considering each line, assume that all the correct lines above it have executed.

```java
Game g1, g2, g3, g4;
g1 = new Game("Backgammon", 2);
g2 = new Checkers();
g2.firstPlayer();
g3 = new Chess();
g4 = new Solitaire();
Checkers checkers;
checkers = g2;
checkers = (Checkers) g2;
checkers = g3;
checkers = (Checkers) g3;
checkers = g4;
checkers = (Checkers) g4;
checkers = new Checkers();
Chess chess;
chess = checkers;
chess = (Chess) checkers;
Solitaire s = new Solitaire();
chess = s;
chess = (Chess) s;
```

3. What output would be printed by the following statements?

```java
Game game = new Chess();
System.out.println(game.toString());
Checkers checkers = new Checkers();
System.out.println(checkers.toString());
Solitaire sol = new Solitaire();
System.out.println(sol.toString());
Game g = sol;
System.out.println(g.toString().equals(sol.toString()));
```