Scala in Practice

lab 04

Acceptance criteria:

Create Scala program with:

- Package cards with abstractions to represent a deck of cards:
 - Standard deck consists of thirteen cards for each of the four colors: Clubs ♣, Diamonds ♠, Hearts ♥ and Spades ♠ (52 cards total). The thirteen cards for each color have the values Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King. This should be a valid definition: val exampleCard = Card(Hearts, Queen)

```
Package deck with:
  class Deck(cards: List[Card]) {
     def pull() = ??? //creates new deck without first card
     def push(c: Card) = ??? //creates new deck with given card
     pushed on top
     def push(color: ..., value: ...) = ??? //creates new deck with new
     card(color, value) pushed on top
     val isStandard: Boolean = ??? // checks if deck is a standard
     deck
     def duplicatesOfCard(card: ...): Int = ??? //amount of
     duplicates of the given card in the deck
     def amountOfColor(color: ...): Int = ??? //amount of cards
     in the deck for the given color
     def amountOfNumerical(numerical: ...): Int = ??? //amount of
     cards in the deck for given numerical card (2, 3, 4, 5, 6,
     7, 8, 9, 10)
     val amountWithNumerical: Int = ??? //amount of all numerical
     cards in the deck (2, 3, 4, 5, 6, 7, 8, 9, 10)
     def amountOfFace(face: ...) : Int = ??? //amount of cards in
     the deck for the given face (Jack, Queen & King)
     val amountWithFace: Int = ??? //amount of all cards in the
```

deck with faces (Jack, Queen & King)

Scala in Practice

lab 04

```
object Deck implementing method:
      def apply() = ??? //creates the standard deck with random
      order of cards. Check Random.shuffle<sup>1</sup> function
    }
Package games with:
    class Blackjack(deck: Deck) {
       // Points calculation:
      1. Numerical cards as their numerical value = 2 - 10.
      2. Face cards (Jack, Queen, King) = 10
      3. Ace = 1 or 11 (player could choose)
      def play(n: Int): Unit = ??? // loop taking n cards from the
      deck, pretty-printing them with points & printing the sum of
       points on the end
      lazy val all21: List[List[Cards]] = ??? // finds all
       subsequences of cards which could give 21 points
      def first21(): Unit = ??? // finds and pretty-prints the
      first subsequence of cards which could give 21 points
    }
    object Blackjack {
      def apply(numOfDecks: Int) = ??? // creates Blackjack game
      having numOfDecks-amount of standard decs with random order
      of cards. For example, with Blackjack(3) deck would have 156
      cards
    }
```

• Create *application entry-point* object with some example tests for the above implementation

Michał Kowalczykiewicz

¹ https://www.scala-lang.org/api/current/scala/util/Random\$.html