Evaluating a hand of cards

We consider an imaginary game in which each hand of cards is scored according to the number of pairs, three-of-a-kind, and four-of-a-kind sets it contains:

- **Four of a kind** (e.g. 7♠ 7♥ 7♣ 7♦): +100 points
- **Three of a kind** (e.g. 8♥ 8♣ 8♦): +10 points
- **Pair** (e.g. 9♠ 9♣): +1 point

For example, the following hand of 10 cards:

5♠ 5♣ 5♦ 7♥ 7♦ J♦ A♠ A♥ A♣ A♦

evaluates as:

10 + 1 + 0 + 100 = 111

Step-by-step implementation:

1. Using the provided classes `Card` and `Deck`, write a function `deal(n)` that creates a randomly shuffled deck and deals a hand of `n` cards, which are returned as a list.
2. Write a function `evaluate(hand)`, which, given a list of card objects, evaluates it according to the rules described in the previous section and returns the score. (Exercise 6 from Unit 5 can be helpful for implementing this.)

Write a text user interface that repeatedly asks the user how many cards should be dealt, creates a hand of the requested size and evaluates it. The program should check that the user input is an

integer (use `isdigit`) and is in the range $0 \leq n \leq 52$. Example:

```
Number of cards: 5
     10 of hearts
     6 of spades
     8 of diamonds
     ace of clubs
     jack of hearts
     -----------> Score: 0

Number of cards: 7
     2 of diamonds
     10 of diamonds
     10 of spades
     10 of clubs
     king of diamonds
     ace of clubs
     9 of diamonds
     -----------> Score: 10

Number of cards: 20
     6 of hearts
     8 of diamonds
     8 of spades
     10 of hearts
     2 of clubs
     2 of diamonds
     7 of hearts
     6 of diamonds
     4 of diamonds
     4 of hearts
     queen of spades
     6 of spades
     3 of spades
     9 of spades
     7 of diamonds
     8 of hearts
     2 of spades
     4 of clubs
     8 of clubs
     5 of diamonds
     -----------> Score: 131
```

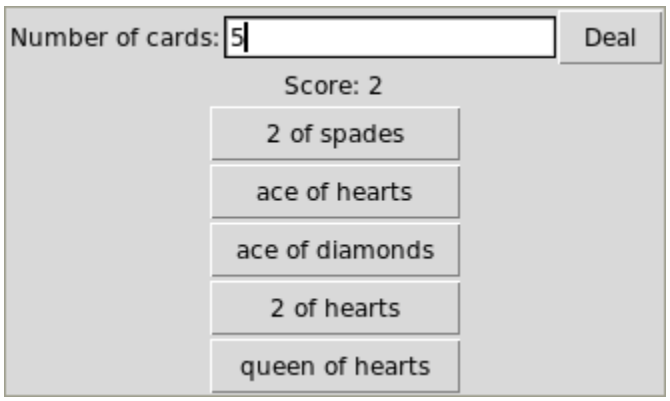
```
Number of cards: 3
     king of clubs
     9 of hearts
     jack of hearts
     -----------> Score: 0

Number of cards: 10
     ace of spades
     king of hearts
     jack of diamonds
     queen of spades
     8 of diamonds
     8 of spades
     9 of clubs
     jack of hearts
     ace of clubs
     king of diamonds
     -----------> Score: 4
```

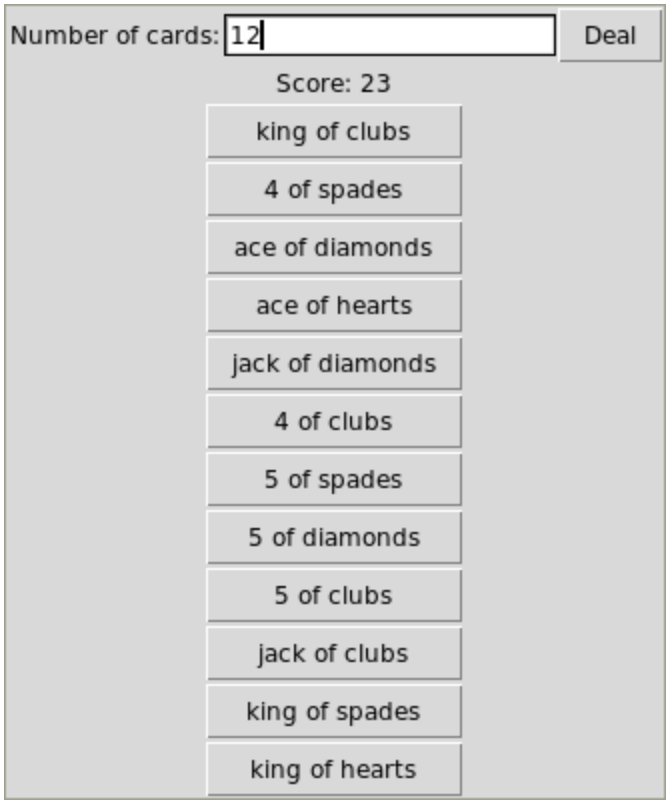
3. The purpose of this step is only to help you write the rest of the program. The program you submit only needs the graphical interface you will create in the later steps, not the text user interface from this step.

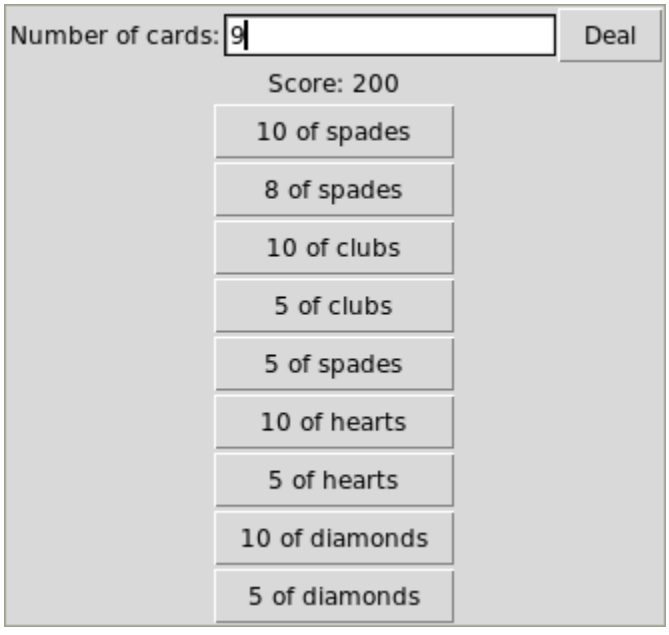
4. Make a widget `CardsFrame` that is a specialized version of `Frame`, which holds a list of buttons with card names on them. Its `__init__` function should receive a list of `Card` objects as a parameter, specifying which cards should be shown:

You don't need to specify the `['command']` options for the buttons, thus clicking a button will do nothing.

5. Make a Tkinter interface for the program, using the `enhancedEntry` and `CardsFrame` widgets. When the user presses the button '***Deal***', a new hand is generated, `CardsFrame` should be updated (you can `destroy` the old widget replacing it with a new one), and the score of the new hand should be shown in the corresponding label:




Number of cards: 12
Deal
Score: 23
king of clubs
4 of spades
ace of diamonds
ace of hearts
jack of diamonds
4 of clubs
5 of spades
5 of diamonds
5 of clubs
jack of clubs
king of spades
king of hearts


Number of cards: 9
Deal
Score: 200
10 of spades
8 of spades
10 of clubs
5 of clubs
5 of spades
10 of hearts
5 of hearts
10 of diamonds
5 of diamonds

One can program a card game using an improved version of the `CardsFrame` widget. It will require setting up `['command']` options on the buttons within the frame.