TP 2 - Problem 14 - I

```cpp
#include <cassert>
#include <iostream>
#include <random>
#include <vector>

#define SHOW(arg) std::cout << "Macro SHOW “" #arg "”: " << (arg) << '\n';

// This enumeration class represents the different values within a color.
enum class Value {
  v1,
  v2,
  v3,
  v4,
  v5,
  v6,
  v7,
  v8,
  v9,
  v10,
```

TP 2 - Problem 14 - II

```
20      jack,
21      queen,
22      king
23    };
24    // This enumeration class represents the different suits.
25    enum class Color { club, diamond, heart, spade };
26    // This class represents a card.
27    struct Card {
28      Value value;
29      Color color;
30    };
31    // Returns an external representation of a card.
32    std::string to_string(Card card) {
33      // “rvo” stands for return value optimisation.
34      std::string rvo{"["};
35      switch (card.value) {
36      case Value::jack:
37        rvo += "Jack";
38        break;
39      case Value::queen:
```

TP 2 - Problem 14 - III

```
40        rvo += "Queen";
41        break;
42      case Value::king:
43        rvo += "King";
44        break;
45      default:
46        // Use a “static_cast” to get the numeric value of the enumerator.
47        rvo += std::to_string(static_cast<int>(card.value) + 1);
48        break;
49      }
50      rvo += " of ";
51      switch (card.color) {
52      case Color::club:
53        rvo += "clubs";
54        break;
55      case Color::diamond:
56        rvo += "diamonds";
57        break;
58      case Color::heart:
59        rvo += "hearts";
```

TP 2 - Problem 14 - IV

```
60        break;
61      case Color::spade:
62        rvo += "spades";
63        break;
64      }
65      rvo += ']';
66      return rvo;
67    }
68    // Displays the cards of the game, assuming 4 gamers.
69    void display_game(std::vector<Card> const &game) {
70      assert(game.size() % 4 == 0);
71      int i{};
72      for (char const *const &char_ptr : {"South", "West", "North", "East"}) {
73        std::cout << char_ptr << '\n';
74        do {
75          std::cout << to_string(game.at(i)) << '\n';
76        } while (i++ % (game.size() / 4) != (game.size() / 4 - 1));
77      }
78    }
79    // Shuffles the cards of the game, using Fischer-Yates (or Knuth) method.
```

TP 2 - Problem 14 - V

```
80  void mix_game(std::vector<Card> &game) {
81    if (game.size() <= 1)
82      return;
83    // A pseudo random number generator.
84    std::mt19937 prng;
85    // “i” is always > 0: it is safe to decrement it.
86    for (size_t i{game.size() - 1}; i >= 1; --i) {
87      // Get an pseudo random index in [0, i] using the remainder of the
88      // euclidian division.
89      size_t const j{prng() % (i + 1)};
90      Card tmp{game.at(i)};
91      game.at(i) = game.at(j);
92      game.at(j) = tmp;
93    }
94  }
95
96  int main() {
97    int answer;
98    while (true) {
99      std::cout << "32 or 52 cards game (1|2)?\n";
```

TP 2 - Problem 14 - VI

```
100       std::cin >> answer;
101       if ((answer == 1) || (answer == 2))
102         break;
103     }
104     int const cards_nr{(answer == 1) ? 32 : 52};
105     std::vector<Card> game;
106     // Reserves the memory for the cards, to avoid reallocation.
107     game.reserve(cards_nr);
108     // For each color.
109     for (Color const &color :
110          {Color::club, Color::diamond, Color::heart, Color::spade}) {
111       if (cards_nr == 32) {
112         for (Value const &value :
113              {Value::v1, Value::v7, Value::v8, Value::v9, Value::v10,
114               Value::jack, Value::queen, Value::king})
115           game.push_back(Card{value, color});
116       } else {
117         for (Value const &value :
118              {Value::v1, Value::v2, Value::v3, Value::v4, Value::v5,
119               Value::v6, Value::v7, Value::v8, Value::v9, Value::v10,
```

TP 2 - Problem 14 - VII

```
120                 Value::jack, Value::queen, Value::king})
121             game.push_back(Card{value, color});
122         }
123     }
124     display_game(game);
125     mix_game(game);
126     std::cout << "== Shuffled game ==\n";
127     display_game(game);
128     return 0;
129 }
```

TP 2 - Problem 14 - VIII

Possible output:

```
32 or 52 cards game (1|2)?
South
[1 of clubs]
[7 of clubs]
[8 of clubs]
[9 of clubs]
[10 of clubs]
[Jack of clubs]
[Queen of clubs]
[King of clubs]
West
[1 of diamonds]
[7 of diamonds]
[8 of diamonds]
[9 of diamonds]
[10 of diamonds]
[Jack of diamonds]
[Queen of diamonds]
```

TP 2 - Problem 14 - IX

```
19  [King of diamonds]
20  North
21  [1 of hearts]
22  [7 of hearts]
23  [8 of hearts]
24  [9 of hearts]
25  [10 of hearts]
26  [Jack of hearts]
27  [Queen of hearts]
28  [King of hearts]
29  East
30  [1 of spades]
31  [7 of spades]
32  [8 of spades]
33  [9 of spades]
34  [10 of spades]
35  [Jack of spades]
36  [Queen of spades]
37  [King of spades]
38  == Shuffled game ==
```

TP 2 - Problem 14 - X

```
39  South
40  [1 of diamonds]
41  [9 of spades]
42  [Queen of spades]
43  [Queen of clubs]
44  [7 of diamonds]
45  [Jack of spades]
46  [Queen of hearts]
47  [8 of clubs]
48  West
49  [10 of diamonds]
50  [King of hearts]
51  [9 of diamonds]
52  [10 of hearts]
53  [Jack of hearts]
54  [8 of spades]
55  [10 of clubs]
56  [King of clubs]
57  North
58  [King of spades]
```

TP 2 - Problem 14 - XI

```
59   [Jack of clubs]
60   [9 of clubs]
61   [1 of clubs]
62   [7 of hearts]
63   [7 of spades]
64   [9 of hearts]
65   [8 of hearts]
66   East
67   [8 of diamonds]
68   [Jack of diamonds]
69   [7 of clubs]
70   [1 of hearts]
71   [1 of spades]
72   [Queen of diamonds]
73   [King of diamonds]
74   [10 of spades]
```