HI4 オブジェクト指向プログラミング
前期期末レポート

人間情報システム工学科 4 年 36 号
松山　京介

2021 年 7 月 18 日

課題 3-1: Card クラスの実装

課題内容

スペード (spade)，ハート (heart)，ダイヤ (diamond)，クラブ (club) のいずれかの「スート」(suit) と 1 から 13 までの「数字」(number) を持つトランプのカードを模したクラス Card を作成したい．Card クラスは，次のようなフィールドおよびメソッドを持つものとする．

Card のフィールド

- `private int type`
  カードの「スート」を示す整数のフィールド．
- `private int number`
  カードの「強さの値」を示すフィールド．1 以上 13 以下の整数である

Card のメソッド

- `boolean isStrongerThan(Card c)`
  Card クラスのインスタンス c を受け取り，次のルールで自分 (インスタンス自身) と c とを勝負させる．その結果，自分が「勝ち」であれば true，そうでなければ false を返す．
  勝負のルール:
  - 数字が異なる場合，数字が大きい方を「勝ち」とする．
  - 数字が同じである場合，スートを比較する．スートは，スペード，ハート，ダイヤ，クラブの順番で強いものとする．
  - 数字もスートも同じ場合は，「引き分け」とする．
- `boolean isSameAs(Card c)`
  Card クラスのインスタンス c を受け取り，c のスートと数字が自分と同じ (すなわち，上記「勝負のルール」での引き分けの関係) であれば true，そうでなければ false を返す．
- `public String toString()` (オーバーライド)
  自分のスートと数字の値を含んだ文字列 (たとえば，「HEART [10]」といったもの) を返す．トランプらしく，1 を A，11 から 13 をそれぞれ，J，Q，K で表してもよい (たとえば，「DIAMOND[J]」)

Card のコンストラクタ

- `Card()`
  ランダムに決定された「スート」と「数字」で初期化する．
- `Card(int suit, int number)`
  引数に指定された「スート」と「数字」で初期化する．

プログラム

次のようなプログラムを作成した.

ソースコード 1: Card.java

```
1  import java.util.Random;
2
3  class Card{
4      //スートを示す定数値.「Card.SUIT_SPADE」といった表現で取得できる.
5      public static final int SUIT_SPADE=0;
6      public static final int SUIT_HEART=1;
7      public static final int SUIT_DIAMOND=2;
8      public static final int SUIT_CLUB=3;
9
10     private int suit;
11     private int number;
12
13     Card(){
14         Random random=new Random();
15         suit=random.nextInt(4);
16         number=random.nextInt(13);
17     }
18
19     Card(int suit,int number){
20         this.suit=suit;
21         this.number=number;
22     }
23
24     boolean isStrongerThan(Card c){
25         if(c.number<number){
26             return true;
27         }
28         else if(c.number>number){
29             return false;
30         }
31         else{
32             if(c.suit>suit){
33                 return true;
34             }
35             else{
36                 return false;
37             }
38         }
39     }
40
```

```
41      boolean isSameAs(Card c){
42          if(c.number==number && c.suit==suit){
43              return true;
44          }
45          else{
46              return false;
47          }
48      }
49
50      @Override
51      public String toString(){
52          String result="";
53          switch(suit){
54              case SUIT_SPADE:
55                  result="SPADE   ["+number+"]";
56                  break;
57
58              case SUIT_CLUB:
59                  result="CLUB    ["+number+"]";
60                  break;
61
62              case SUIT_DIAMOND:
63                  result="DIAMOND ["+number+"]";
64                  break;
65
66              case SUIT_HEART:
67                  result="HEART   ["+number+"]";
68                  break;
69          }
70          return result;
71      }
72  }
```

ソースコード 2: CardTest.java

```
1   class CardTest{
2       public static void main(String[] args){
3           System.out.println(new Card());
4           System.out.println(new Card());
5           System.out.println(new Card());
6           System.out.println();//一行あける
7
8           Card c1= new Card(Card.SUIT_SPADE,4);
9           Card c2= new Card(Card.SUIT_HEART,12);
10          Card c3= new Card(Card.SUIT_CLUB,4);
11          Card c4= new Card(Card.SUIT_SPADE,4);
```

```
12          Card c5= new Card(Card.SUIT_HEART,4);
13          Card c6= new Card(Card.SUIT_DIAMOND,4);
14
15          //isStrongerThan メソッドの判定
16          System.out.println(c1+"  is stronger than "+c2+"? "+c1.isStrongerThan(c2));
17          System.out.println(c1+"  is stronger than "+c3+"? "+c1.isStrongerThan(c3));
18          System.out.println(c1+"  is stronger than "+c4+"? "+c1.isStrongerThan(c4));
19          System.out.println(c1+"  is stronger than "+c5+"? "+c1.isStrongerThan(c5));
20          System.out.println(c1+"  is stronger than "+c6+"? "+c1.isStrongerThan(c6));
21          System.out.println(c3+"  is stronger than "+c5+"? "+c3.isStrongerThan(c5));
22          System.out.println(c3+"  is stronger than "+c6+"? "+c3.isStrongerThan(c6));
23          System.out.println(c5+"  is stronger than "+c6+"? "+c5.isStrongerThan(c6));
24          System.out.println();//一行あける
25
26          //同じかどうかの判定
27          System.out.println(c1+"  is same as "+c2+"? "+c1.isSameAs(c2));
28          System.out.println(c1+"  is same as "+c4+"? "+c1.isSameAs(c4));
29
30      }
31  }
```

実行例

実行例を以下に 2 つ示す.

```
DIAMOND [12]
HEART   [10]
HEART   [6]

SPADE   [4]  is stronger than HEART   [12]? false
SPADE   [4]  is stronger than CLUB    [4]? true
SPADE   [4]  is stronger than SPADE   [4]? false
SPADE   [4]  is stronger than HEART   [4]? true
SPADE   [4]  is stronger than DIAMOND [4]? true
CLUB    [4]  is stronger than HEART   [4]? false
CLUB    [4]  is stronger than DIAMOND [4]? false
HEART   [4]  is stronger than DIAMOND [4]? true

SPADE   [4]  is same as HEART   [12]? false
SPADE   [4]  is same as SPADE   [4]? true
```

```
DIAMOND [7]
CLUB    [9]
DIAMOND [8]

SPADE   [4]  is stronger than HEART   [12]? false
SPADE   [4]  is stronger than CLUB    [4]? true
```

```
SPADE   [4]  is stronger than SPADE   [4]? false
SPADE   [4]  is stronger than HEART   [4]? true
SPADE   [4]  is stronger than DIAMOND [4]? true
CLUB    [4]  is stronger than HEART   [4]? false
CLUB    [4]  is stronger than DIAMOND [4]? false
HEART   [4]  is stronger than DIAMOND [4]? true

SPADE   [4]  is same as HEART   [12]? false
SPADE   [4]  is same as SPADE   [4]? true
```

考察

プログラムを実行するたびに, ランダムにカードが生成されていることが確認できる.
また, isStrongerThan メソッドの各判定パターンについて

- スペードの 4 とハートの 12 では, ハートの 12 の方が数字が大きいため false が出力された.
- スペードの 4 とクラブの 4 では, スペードの方が強いため true が出力された.
- 全く同じカードのときは, false が出力された.
- スペードの 4 とハートの 4 では, スペードの方が強いため true が出力された.
- スペードの 4 とダイヤの 4 では, スペードの方が強いため true が出力された.
- クラブの 4 とハートの 4 では, ハートの方が強いため false が出力された.
- クラブの 4 とダイヤの 4 では, ダイヤの方が強いため false が出力された.
- ハートの 4 とダイヤの 4 では, ハートの方が強いため true が出力された.

以上のことから, 正しく動作していると考えられる.

感想

正しい結果を出力することができたが, 手際よくプログラムを書けなかったのでもっと精進したい.