MC322/MC336
Segundo semestre de 2015

Laboratório 2

Professor: Fábio Luiz Usberti (fusberti@ic.unicamp.br)
PED: Rafael Arakaki (rafaelkendyarakaki@gmail.com)

---

1 Objetivo

O objetivo desta atividade será o aprendizado da criação de classes e métodos em Java, variáveis e métodos estáticos, uso de *arrays* e a classe *Random* da biblioteca padrão do Java.

2 Atividade

Nesta atividade o principal foco será a programação de uma classe chamada **Baralho**. Para trabalhar com a classe Baralho, logicamente você precisará ter a classe **Carta**.

Para facilitar, crie um novo projeto no Eclipse chamado "Lab2" e adicione a este duas classes: Carta e Baralho. A classe Carta deve ser igual à implementada no laboratório 1, e a classe Baralho será programada neste laboratório. Coloque as duas classes no mesmo pacote (*package*), que pode ser o padrão.

As atividades serão:

1. Criação do projeto e das duas classes no mesmo pacote.
2. Declaração dos atributos da classe Baralho.
3. Criação do construtor da classe Baralho.
4. Implementação dos métodos adicionarCarta() e comprarCarta() da classe Baralho.
5. Implementação do método embaralhar() da classe Baralho.
6. Implementação da função main() criando um baralho, adicionando algumas cartas e embaralhando.
7. Responder às questões usando comentários no código.

3 Classe Baralho

A classe Baralho deve ter os seguintes atributos:

- vetorCartas (array de objetos Carta)
- nCartas (número inteiro)
- gerador (atributo **estático** da classe Random[1]).

Um exemplo da declaração dos objetos é mostrada a seguir:

[1] Observe que é necessário importar a classe Random da biblioteca padrão do Java, para isso use: import java.util.Random;

```java
public class Baralho {

    Carta[] vetorCartas;
    int nCartas;
    static Random gerador = new Random();
    //...
}
```

atributos.java

Durante a construção do objeto Baralho, é necessário atentar-se de que a variável nCartas deve receber 0, e os atributos vetorCartas e gerador precisam ser inicializados corretamente. Repare na maneira como o array é inicializado.

Já o objeto gerador, como é um atributo estático, foi inicializado já na declaração do atributo. Veja o exemplo abaixo:

```java
public Baralho() {
    vetorCartas = new Carta[10];
    nCartas = 0;
}
```

construtor.java

A classe Baralho irá possuir dois métodos que adicionam e retiram cartas, respectivamente: adicionarCarta() e comprarCarta(). Como convenção, iremos adicionar e remover cartas sempre do final do *array* **vetorCartas**. O código destas classes é muito simples, veja abaixo:

```java
public void adicionarCarta (Carta card){
    vetorCartas[nCartas] = card;
    nCartas++;
}

public Carta comprarCarta (){
    nCartas--;
    return vetorCartas[nCartas];
}
```

adicionarcomprar.java

Por fim a classe Baralho também irá possuir um método para embaralhar as cartas. Para isso, existe um algoritmo, que será descrito a seguir: tendo como entrada um baralho com posições variando de 1 a n, a partir da 2a carta sorteie uma posição dentre 1 até 2 para esta carta, se a posição for diferente da atual (diferente de 2) deve-se trocar as cartas da posição 1 e 2 (a 2a carta vai para a posição 1 e a carta que estava na posição 1 vem para a posição 2). Faça o mesmo para as cartas 3, ..., n, só que a posição sorteada para a i-ésima carta deve ser sempre entre $[1, i]$. O algoritmo acima considera que as posições iniciam em 1, adaptando para a linguagem Java, onde as posições do array iniciam em 0, o código fica como abaixo:

```java
public void embaralhar(){
    for(int i = 1; i < nCartas; i++){
        int j = gerador.nextInt(i+1); // Sorteia um numero entre [0,i]
        if(j != i){ // Se posicao diferente, troque
            Carta a = vetorCartas[i];
            Carta b = vetorCartas[j];
            vetorCartas[i] = b;
            vetorCartas[j] = a;
        }
    }
```

```
11      // Imprima as cartas em ordem reversa do array aqui
12  }
```

embaralhar.java

Após embaralhar as cartas, o método embaralhar deve imprimir as cartas na **ordem reversa do array** (de $n-1$ até 0 no array). Que é a ordem que, segundo nossa convenção, o usuário irá comprar as cartas. Para imprimir as cartas, deve-se utilizar **System.out.println(<objeto carta>)** porque assim será utilizado o método toString() das cartas implementado no Lab1.

4 Saída do programa

Por fim, crie uma função main e nela declare novos objetos[2]: um baralho e três cartas diferentes quaisquer. Adicione as três cartas no baralho, e chame o método embaralhar. Verifique se de fato as cartas estão sendo embaralhadas (rode o programa algumas vezes até a ordem mudar).

5 Questões

Algumas questões devem ser respondidas no código, como comentários e de maneira sucinta. As questões são:

1. **(Q1):** No método construtor, o que acontece se adicionarmos mais de 10 cartas, do jeito que está ? Que tipo de erro o Java acusará ? (Se não souber, teste!)
2. **(Q2):** No método comprarCarta, o que acontece se chamarmos este método com o baralho vazio ? Que tipo de erro o Java acusará ? (Se não souber, teste!)
3. **(Q3):** Na declaração do atributo gerador, porque este atributo pode ser estático ? Qual a diferença de escopo de um atributo "estático" para um "não-estático" ?
4. **(Q4):** Por que o atributo gerador não é inicializado no construtor da classe Baralho ? Em nosso programa, quantas vezes o comando new Random() será executado ?

Marque os comentários para cada questão x iniciando com "(Qx): ...", como mostra o exemplo abaixo:

```
1 // (Q3): E mais eficiente porque blablabla.
2 static Random gerador = new Random();
```

comentarios.java

Algumas questões extras, que não precisam ser respondidas (não vale ponto), apenas para pensar:

- No método embaralhar, é garantida a distribuição uniforme no embaralhamento ? Isto é, para uma dada carta em um baralho de n posições, ela possui $1/n$ de chance de ficar em cada posição ?
- Como ficaria a rotina comprarCarta se desejássemos que a carta comprada fosse aquela do índice 0 do array ? A rotina ficaria computacionalmente mais pesada ?

[2] Se é um novo objeto, você deve utilizar o "new".

6 Submissão

Para submeter desta atividade utilize a página da disciplina no Ensino Aberto. Utilize o recurso de portfólio para submeter a atividade. Para isso, crie uma subpasta dentro de seu portfólio com o nome **Lab2** e dentro desta subpasta submeta o arquivo Baralho.java com a sua implementação da classe Baralho. Ao submeter, marque a opção "Compartilhado com Formadores" na opção de compartilhamento.