MC322
Segundo semestre de 2018

Laboratório 1

Professor: Fábio Luiz Usberti (fusberti@ic.unicamp.br)
PED: Luis Henrique Pauleti Mendes (luishpmendes@gmail.com)
PAD: Pedro Barros Bastos (p204481@dac.unicamp.br)

---

1 Objetivo

O objetivo desta atividade consiste na familiarização com o ambiente de desenvolvimento integrado (IDE, *Integrated Development Environment*) chamado Eclipse[1] e a linguagem de programação Java[2].

2 Atividade

Nesta atividade o principal foco será a familiarização com o Eclipse e a programação de uma classe chamada **Card**. A primeira tarefa será configurar o ambiente com a criação de um novo projeto e de uma nova classe para então programar.

Os seguintes passos podem ser tomados para a criação do projeto:

1. Abra o Eclipse.
2. Crie um novo projeto (File -> New -> Project... -> Java Project).
3. Digite o nome do projeto (ex: Lab1).
4. Clique em Finish.

Para criar uma nova classe faça:

1. Utilize a aba 'Package Explorer'.
2. Crie uma nova classe no projeto (Botão direito no projeto -> New -> Class).
3. Digite o nome da classe.
4. Programe a classe.

3 Classe Card

A classe Card deste laboratório é baseada em um jogo de cartas chamado Svoyi Koziri[3].

A classe Card deve ter os seguintes atributos:

- suit (cadeia de caracteres - *String*)
- rank (caractere)

[1] https://eclipse.org
[2] https://www.java.com
[3] https://en.wikipedia.org/wiki/Svoyi_Koziri

O exemplo abaixo apresenta a declaração da classe Dog e seus atributos. Note que todas as variáveis são declaradas como privadas (**private**). Note também a implementação dos métodos de acesso get() e set(), esses métodos são comumente utilizados na linguagem Java para acessar os atributos dos objetos.

```java
public class Dog {

    private String name;
    private String breed;
    private String color;

    // Metodo construtor aqui

    // Demais metodos aqui
    pulic String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }

}
```

Dog.java

Além disso a classe Card deve conter um método construtor, que recebe como argumentos os atributos para inicializar o objeto. Para ilustrar esse conceito melhor, veja o exemplo abaixo.

```java
public class Dog {

    // Atributos aqui

    public Dog(String name, String breed, String color) {
        this.name = name;
        this.breed = breed;
        this.color = color;
    }

    // Outros metodos...
}
```

construtor.java

Também é necessário que a classe Card possua uma função **toString()** que devolve uma String contendo uma descrição geral dos atributos da carta. Veja o exemplo abaixo:

```java
public class Dog {

    // Atributos e outros metodos aqui

    @Override
    public String toString() {
        String out = "";
        out = out + "Name = " + this.getName() + "\n";
        out = out + "Breed = " + this.getBreed() + "\n";
        out = out + "Color = " + this.getColor() + "\n";
        return out;
    }

}
```

toString.java

O formato do método toString() a ser implementado é livre, mas todos os atributos devem ser impressos.

Faça a implementação do método **construtor**, dos métodos **get()** e **set()** de cada atributo e do método **toString()** para a classe **Card**.

4 Classe Main

Para um programa Java funcionar é requerida a existência de um método main que serve de ponto da partida para o programa ser inicializado. Crie uma nova classe através do Eclipse chamada Main e escolha a opção para gerar automaticamente o método main.

Na função main realize a instanciação de alguns objetos do tipo Carta com valores de atributos quaisquer conforme sua imaginação. Após instanciar os objetos, imprima seus dados utilizando o método System.out.println(). Veja o exemplo a seguir:

```java
public class Main {

    public static void main(String[] args) {
        // Instanciando objetos
        Dog dog1 = new Dog("Fred", "Poodle", "Black");
        Dog dog2 = new Dog("Bob", "Yorkshire", "Brown");
        Dog dog3 = new Dog("Cisco", "Lhasa Apso", "White");

        // Impressao dos objetos
        System.out.println("Primeiro cachorro:\n" + dog1);
        System.out.println("Segundo cachorro:\n" + dog2);
        System.out.println("Terceiro cachorro:\n" + dog3);
    }

}
```

Main.java

Observe que ao imprimir os dados dos objetos da classe Carta, o método toString() que você implementou foi chamado implicitamente.

Após implementar as três classes, para executar o programa e ver o resultado clique no botão “Run” do Eclipse.

5 Documentação Javadoc

Javadoc é uma ferramenta para gerar documentação em HTML a partir dos comentários no código fonte. Um comentário é escrito em HTML e deve preceder uma declaração de classe, de atributo, de construtor ou de método. Ele é constituído de duas partes: uma descrição seguida de tags de bloco.

Veja no exemplo a seguir uma documentação javadoc da classe Dog que utiliza as tags de bloco @author, @param e @return.

```java
/**
  * A classe <code>Dog</code> representa um cachorro.
  *
  * @author Luis H. P. Mendes <luishpmedes@gmail.com>
  */
public class Dog {
    /**
      * O nome do cachorro
      */
    private String name;
    // Outros atributos...

    /**
      * Inicializa um objeto de <code>Dog</code> recem-criado
      *
      * @param name  o nome do cachorro
      * @param breed a raca do cachorro
      * @param color a cor do cachorro
      */
    public Dog(String name, String breed, String color) {
        this.name = name;
        this.breed = breed;
        this.color = color;
    }

    /**
      * Recupera o nome do cachorro
      *
      * @return o nome do cachorro
      */
    public String getName() {
        return this.name;
    }

    /**
      * Define o nome do cachorro
      *
      * @param name o nome do cachorro
      */
    public void setName(String name) {
        this.name = name;
    }
    // Outros getters e setters...

    /**
      * Retorna uma representacao em String do cachorro
      *
      * @return uma representacao em String do cachorro
      */
    @Override
    public String toString() {
        String out = "";
        out = out + "Name = " + this.getName() + "\n";
        out = out + "Breed = " + this.getBreed() + "\n";
        out = out + "Color = " + this.getColor() + "\n";
        return out;
    }
}
```

javadoc.java

Para gerar a documentação em HTML vá em Project -> Generate Javadoc... O eclipse irá criar um diretório "doc"dentro do seu projeto com a documentação em HTML.

Para mais detalhes sobre javadoc, consulte o site da Oracle (`https://www.oracle.com/technetwork/java/javase/tech/index-jsp-135444.html`) e a Wikipedia (`https://pt.wikipedia.org/wiki/Javadoc`).

6 Tarefas

- Criação do projeto e classes.
- Programação do método construtor da classe Card.
- Programação dos métodos get e set da classe Card.
- Programação do método toString da classe Card.
- Programação do método main e impressões de algumas cartas.
- Documentação javadoc da classe Card.

7 Submissão

Para submeter a atividade utilize o Moodle (`https://www.ggte.unicamp.br/ea`). Salve os arquivos dessa atividade em um arquivo comprimido no formato .tar.gz e nomeie-o **Lab1-000000.tar.gz** trocando '000000' pelo seu número de RA. Submeta o arquivo na seção correspondente para esse laboratório no moodle da disciplina MC322.

Datas de entrega

- Dia **08/08** até às 23:59