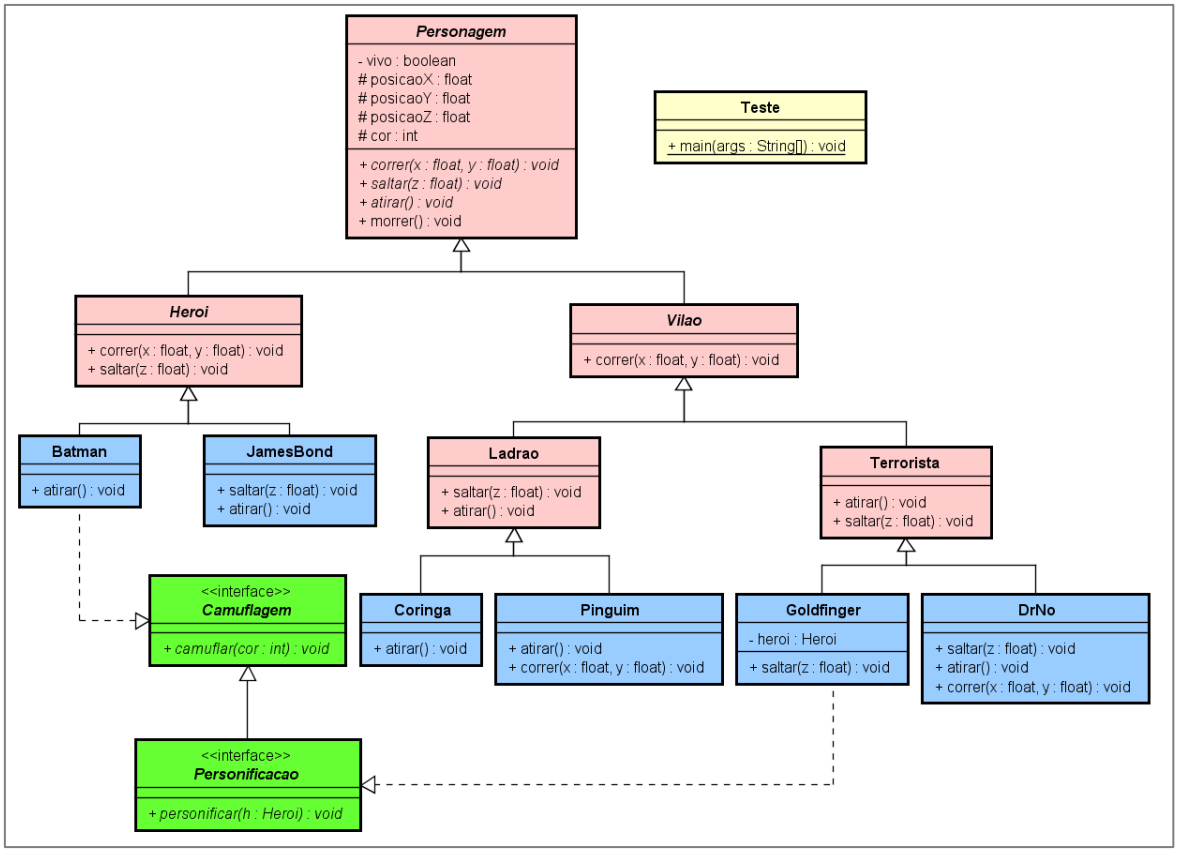
Pontifícia Universidade Católica do Paraná - PUCPR
Programação Orientada a Objetos

RA3: Codificar programas baseados em objetos definidos por hierarquia de classes.

ID11: Codifica classes abstratas e interfaces.

POO – Lista de Exercícios 3



Implemente as classes e interfaces representadas no diagrama acima, de acordo com a seguinte definição de atributos e métodos.

a. **Atributos**

- boolean **vivo**: indica se um personagem está vivo (true) ou morto (false)
- float **posicao_x** : posição de um personagem no eixo-x do espaço
- float **posicao_y**: posição de um personagem no eixo-y do espaço
- float **posicao_z**: posição de um personagem no eixo-z do espaço
- int **cor**: código da cor de um personagem
- Heroi **heroi**: referência para a instância de Heroi personificada pelo objeto da classe Goldfinger

b. **Métodos**

- **correr(float x, float y):** atualiza os atributos posicao_x e posicao_y e imprime a mensagem "C correndo", onde C é o nome da classe que implementa o método. Por exemplo, "Heroi correndo ..."

- **saltar(float z):** atualiza o atributo posicao_z e imprime a mensagem "C saltando", onde C é o nome da classe que implementa o método. Por exemplo, "Goldfinger saltando ..."
- **atirar():** imprime a mensagem "C atirando ...", onde C é o nome da classe que implementa o método. Por exemplo, "Batman atirando ...
- **morrer():** atualiza o atributo vivo para false e imprime a mensagem "Morto".
- **camuflar(int cor):** atualiza a cor do personagem e imprime a mensagem "C camuflando", onde C é o nome da classe que implementa o método. Por exemplo, "Batman camuflando ..."
- **personificar(Heroi h):** atualiza a referência heroi da classe Goldfinger e imprime a mensagem "Personificando"

Deve ser implementada uma classe Teste, contendo o método **main**, o qual deverá criar um objeto de cada classe concreta do diagrama e chamar cada um dos seus métodos ao menos uma vez, como exemplificado a seguir:

```
public class Teste {

	public static void main(String[] args) {
		Batman     bat     = new Batman (1);
		Coringa    coring  = new Coringa(2);
		Pinguim    ping    = new Pinguim(2);

		JamesBond  james   = new JamesBond (1);
		DrNo       dr      = new DrNo (2);
		Goldfinger gold    = new Goldfinger(2);

		bat.atirar();          // chamada de método do Bataman
		bat.correr(20, 30);    // chamada de método herdado
		bat.camuflar(2);       // chamada de método do Bataman
		bat.saltar(20);        // chamada de método herdado
		System.out.println("----------------------");

		coring.correr(20, 30);
		coring.atirar();
		coring.saltar(5);
		System.out.println("----------------------");

		ping.correr(20, 30);
		ping.atirar();
		ping.saltar(4);
		System.out.println("----------------------");

		james.atirar();
		james.correr(20, 30);
		james.saltar(5);
		System.out.println("----------------------");

		dr.correr(20, 30);
		dr.atirar();
		dr.saltar(3);
		System.out.println("----------------------");

		gold.personificar(james);
		gold.camuflar(1);
		gold.correr(20, 30);
		gold.atirar();
		gold.saltar(2);
	}
}
```

Saída na console:

```
Batman atirando...
Heroi correndo...
Batman camuflando...
Heroi saltando...
----------------------
Vilao correndo...
Coringa atirando...
Ladrao saltando... 5.0
----------------------
Pinguim correndo...
Pinguim atirando...
Ladrao saltando... 4.0
----------------------
JamesBond atirando...
Heroi correndo...
JamesBond saltando...  5.0
----------------------
DrNo correndo...
DrNo atirando...
DrNo saltando... 3.0
----------------------
Goldfinger personoficando James Bond...
Goldfinger camuflando....
Vilao correndo...
Terrorista atiando...
Goldfinger saltando...  2.0
```

Exemplo de classes:

```java
public class DrNo extends Terrorista {

    public DrNo(int cor) {
        super(cor);
    }

    public void saltar(float z) {
        System.out.println("DrNo saltando... " + z);
        this.posicaoZ = z;
    }

    public void atirar() {
        System.out.println("DrNo atirando... ");
    }

    public void correr(float x, float y) {
        System.out.println("DrNo correndo... ");
        this.posicaoX = x;
        this.posicaoY = y;
    }

}
```

```java
public abstract class Heroi extends Personagem {

    public Heroi(int cor) {
        super(cor);
    }

    public void correr(float x, float y) {
        System.out.println("Heroi correndo...");
        this.posicaoX = x;
        this.posicaoY = y;
    }

    public void saltar(float z) {
        System.out.println("Heroi saltando...");
        this.posicaoZ = z;
    }

}
```