



Universidade Federal da Fronteira Sul
Ciência da Computação
Programação I

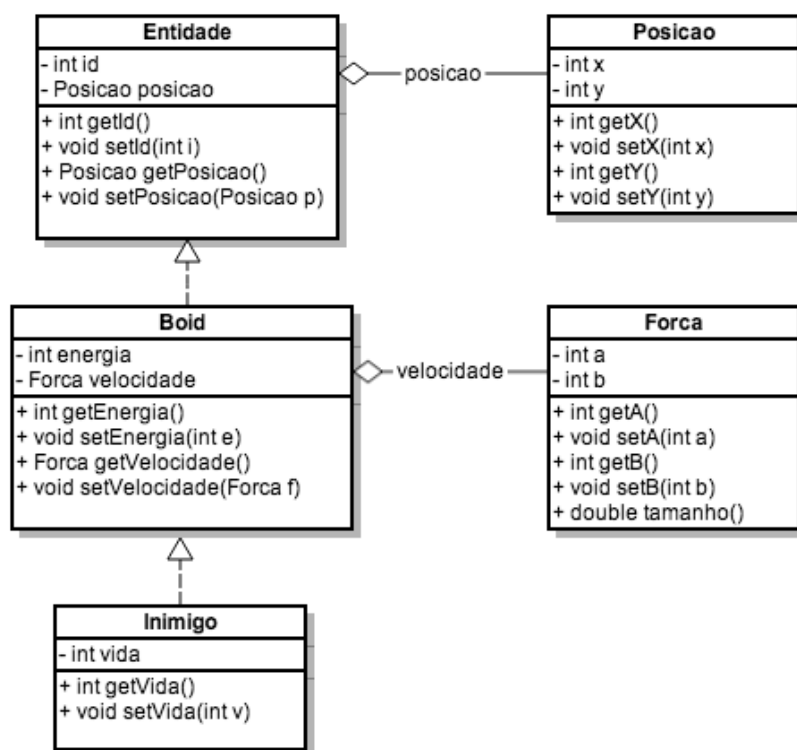
Prova 1

Nome: _____ Data: 13/05/2014

1. A prova pode ser feita a lápis, porém o professor se dará ao direito de não aceitar reclamações relativas à correção.
2. Início da prova 7h30, término 10h00. Manter celulares desligados!
3. Coloque o seu nome nas folhas de resposta.
4. A compreensão das questões faz parte da prova.

Boa prova!

Questões



1) (1,5) Dado o diagrama UML ao lado, implemente na classe `Inimigo` o método: `boolean devoCombater(Inimigo v[])`. O método retorna `true` se no vetor `v` existirem pelo menos 5 inimigos cuja energia seja maior do que a energia do objeto que está invocando o método e a posição (x, y) desses inimigos seja igual à posição (x, y) do objeto invocando o método.

2) (2,0) Utilizando o diagrama UML da questão 1, imagine que você foi incumbido de criar uma classe nova chamada `Barreira`, que é filha da classe `Inimigo`. Ao contrário de sua classe pai, a classe `Barreira` utiliza sua velocidade para calcular a vida. Dessa forma, o método `getVida()` retorna a propriedade `vida` do objeto somada com o tamanho de sua velocidade somada com sua energia. O método `getEnergia()`, por sua vez, retorna o valor da propriedade `energia` dividido pelo atributo `x` da posição da `Barreira`. Implemente a classe `Barreira`. Os métodos herdados que não forem alterados podem ser omitidos no código.

3) (2,5) Faça o diagrama UML com a modelagem de um sistema que represente um jogo de dominó. Nesse jogo, até 4 pessoas podem jogar ao mesmo tempo. Cada jogador recebe uma quantidade determinada de peças, que possuem duas extremidades com números (que vão de zero a seis). Em cada turno, o jogador da vez posiciona uma de suas peças na mesa. Uma peça só pode ser posicionada ao lado de outra que possua o mesmo número na extremidade. Se o jogador não tiver uma peça compatível para jogar, ele pode “comprar” peças do monte. O jogo termina quando um jogador não possui mais peças para jogar (ele é o vencedor).

4) (2,0) Imagine que você foi incumbido de implementar uma classe para uma aplicação matemática. Você recebeu a seguinte explicação. Os números complexos são escritos na forma: $a + bi$. As partes a e b são números reais e o valor de a é a parte real do número complexo e o valor de bi é a parte imaginária do número complexo. Um número complexo z será igual a $a + bi$ ($z = a + bi$). Um exemplo de número complexo é $2 + 5i$ (nesse caso, $a = 2$ e $b = 5$). Dado dois números complexos quaisquer $z_1 = a + bi$ e $z_2 = c + di$, na adição teremos: $z_1 + z_2 = z_3$, onde $z_3 = (a + c) + (b + d)i$. A subtração é similar: $z_1 - z_2 = z_3$, onde $z_3 = (a - c) + (b - d)i$. Você foi incumbido de implementar uma classe que represente números complexos. Os usuários dessa classe querem criar números complexos, editar sua parte real e imaginária (separadamente), além de somar e substrair números complexos. Faça o código completo de uma classe que cumpra essa tarefa.

5) (2,0) Dadas as classes abaixo, mostre o que será impresso na tela quando o programa Main for executado.

<pre> class Sprite { private int x; public int scaleY; public Sprite (int x, int s) { this.x = x; this.scaleY = s; } public int getX() { return this.x; } public void setX(int i) { this.x = i; } public void oi() { imprime(); System.out.println("Soi"); } public void inverte() { this.x = this.scaleY + 1; System.out.println("x = " + getX()); } public String resumo() { return x + " " + scaleY; } public void imprime() { System.out.println(getX() + " " + scaleY); } } </pre>	<pre> class MovieClip extends Sprite { private float alpha; public MovieClip(float h) { super(10, 22); alpha = h; setX(12); } public void negativo() { inverte(); alpha = 2; System.out.println("Negativo!"); } public String resumo() { return "Movie: " + super.resumo(); } public void imprime() { System.out.println(resumo() + " " + alpha); } public void fim() { System.out.println("movie fim"); } public void outro() { fim(); System.out.println("movie: " + alpha); } } </pre>
<pre> class MovieBlock extends MovieClip { public MovieBlock(float h) { super(h); setX(40); } public String resumo() { return "Block: resumo"; } public void imprime() { super.imprime(); System.out.println("Block " + getX()); } public void inverte() { setX(900); System.out.println("b inverte"); } public void fim() { System.out.println("block fim!"); } public void setX(int i) { super.setX(i + 1); System.out.println("setX " + getX()); } } </pre>	<pre> class Main { public static void main(String args[]) { Sprite s = new Sprite(1, 2); MovieClip m = new MovieClip(5.5f); MovieBlock b = new MovieBlock(7.3f); System.out.println(s.resumo()); s.imprime(); m.inverte(); m.imprime(); m.oi(); b.imprime(); b.negativo(); b.outro(); } } </pre>