

Projeto

Objetivo

Você deve utilizar os conhecimentos de orientação a objetos e Java para implementar um jogo computacional com as mesmas características do Pac-man.

Especificações do Jogo

- 1) O jogo deve conter três fases distintas, onde cada uma é caracterizada por um cenário diferente. Você é livre para definir o cenário como quiser, porém a grade da tela deve conter no mínimo 15 x 15 elementos no total.
- 2) Você deve implementar os elementos do jogo listados abaixo. As características de cada um são detalhadas a diante.
 - a. Pacman
 - b. Fantasmas
 - c. Bolinhas comuns (Pac-Dots)
 - d. Bolinhas de poder (Power Pellets)
 - e. Frutas
 - f. Objetos do cenário
- 3) A tela do jogo deve conter, além dos elementos listados acima, a pontuação atual do Pacman, a quantidade de vidas restantes do Pacman e a fase atual do jogo (1, 2 ou 3).
- 4) Você deve implementar uma funcionalidade que permita que o jogo seja salvo a qualquer momento, na situação em que estiver, para que seja carregado posteriormente. Para isso, utilize Serialização. A funcionalidade de salvar pode ser chamada através de uma combinação de teclas, como por exemplo "Ctrl+s". A funcionalidade de carregar um jogo salvo anteriormente só poderá estar disponível antes de iniciar um jogo. Pode ser também através de uma combinação de teclas ou um botão na interface gráfica.

Características dos Elementos do Jogo

Pacman

- Principal elemento do jogo, é dirigido pelo usuário através das setas do teclado (esquerda, direita, cima, baixo) e barra de espaço (parar).
- Considere que a velocidade do Pacman é a mesma em todas as situações.
- Ao comer todas as pac-dots e power pellets do cenário, a fase termina.
- Ao encostar em um fantasma, em condições normais, o Pacman morre.
- Ao comer uma power pellet, o Pacman entra em um estado capaz de comer os fantasmas.
- A cada 10.000 pontos, o Pacman ganha uma vida.

Fantasmas

- Existem quatro fantasmas no jogo, que possuem particularidades quanto à movimentação no cenário. Nenhum dos fantasmas deve parar de se movimentar durante o jogo, porém a decisão de qual direção seguir a cada instante deve seguir regras diferentes em cada fantasma.

- Blinky (vermelho): é o mais esperto dos fantasmas e persegue o Pacman obstinadamente. Seus movimentos deverão ser baseados na direção atual do Pacman. Contudo, adicione um pequeno componente aleatório na tomada de decisão do Blinky para que o jogo não fique sem graça.
- Pinky (rosa): este fantasma tenta encurralar o Pacman, movendo-se paralelamente a ele. A movimentação na outra direção (perpendicular) é aleatória. Adicione também um pequeno componente aleatório pequeno na movimentação paralela ao Pacman, para que os movimentos não sejam idênticos.
- Inky (ciano): este fantasma movimenta-se aleatoriamente enquanto o Blinky estiver distante dele (distância > D). Quando o Blinky se aproxima dele (distância < D), ele se encoraja e passa a perseguir o Pacman de maneira similar ao Blinky.
- Clyde (laranja): este fantasma persegue o Pacman, de maneira similar ao Blinky, quando está a uma distância grande Pacman (distância > D). Porém, quando sua distância ao Pacman diminui (distância < D), seu movimento passa a ser aleatório.
- Assuma um limiar de distância $D=8$ unidades. Isso significa que o elemento1 é considerado perto do elemento2 quando o elemento2 estiver dentro de um raio de 8 unidades de distância do elemento1. A unidade de distância é o tamanho da grade ($D=8$ equivale a 8 elementos da grade).
- Quando o Pacman come uma power pellet, todos os fantasmas mudam de estado: ficam azuis, mais lentos (decidam o quão mais lento) e a direção dos seus movimentos é invertida. Em situação normal, todos os fantasmas movimentam-se com a mesma velocidade. A queda na velocidade dos fantasmas devido a uma power pellet é igual em todos os fantasmas.
- Ao comer um fantasma, o Pacman ganha 200 pontos com o primeiro, 400 com o segundo, 800 com o terceiro e 1600 com o quarto. Isso dentro de um mesmo evento de power pellet.

Bolinhas comuns (Pac-Dots)

- São as bolinhas de pontuação que preenchem todo o cenário.
- O cenário (fase) muda quando o Pacman come todas as pac-dots e power pellets.
- Cada pac-dot vale 10 pontos.

Bolinhas de poder (Power Pellets)

- São bolinhas especiais que dão poder temporário ao Pacman de comer os fantasmas.
- Faz com que o estado dos fantasmas mude (ver item dos fantasmas).
- O poder de uma power pellet deve durar 7 segundos.
- A quantidade de power pellets é fixa em uma fase: são apenas quatro. Sua localização é próxima aos cantos do cenário.
- Cada power pellet comida pelo Pacman vale 50 pontos por si só.

Frutas

- Nesta versão do Pac-man, você deve implementar duas frutas apenas: cereja e morango.
- Cerejas valem 100 pontos e morangos valem 300 pontos.
- O aparecimento das frutas é de tempos em tempos e deve acontecer em posições aleatórias no cenário (em uma posição transitável).
- Você pode decidir se o aparecimento da fruta deverá substituir uma pac-dot/power pellet, ou se será sobreposta a esta. No caso de substituição, cuide

para que quando só reste uma pac-dot/power pellet no cenário a fruta não a substitua, o que geraria uma inconsistência no jogo.

- Cerejas devem surgir no cenário a cada 50 segundos e o morango a cada 75 segundos. Cada fruta ficará no cenário por 15 segundos. Se o Pacman não comer a fruta, ela desaparece.

Objetos do cenário

- São os elementos que definem o cenário, que irão determinar os caminhos onde o Pacman e os fantasmas podem se movimentar.
- Nenhum destes elementos do cenário pode se movimentar. Porém, alguns devem ser transponíveis (apenas fundo) e outros não (paredes).
- Defina a aparência destes objetos de forma que o cenário fique bonito e de fácil identificação dos caminhos.


Framework para Desenvolvimento do Projeto





No Tidia está disponível um framework (conjunto de classes) sobre o qual o projeto pode ser desenvolvido. Nele já estão implementadas várias funcionalidades que facilitarão o desenvolvimento.


A tela do jogo foi definida como uma grade de $N \times N$ elementos, sendo que cada elemento da grade irá conter um objeto do cenário. O valor de N pode ser alterado. O Pacman e os fantasmas podem caminhar pela grade com passos menores do que N , isto é em posições não inteiras da grade. Isso dá uma aparência mais agradável à dinâmica do jogo. Note porém, que os elementos só poderão se movimentar quanto os objetos inteiros não se chocarem (objetos definidos como não transponíveis).

Abaixo está o diagrama de classes simplificado do conjunto de classes disponibilizado. Você pode alterar a estrutura destas classes como quiser, se achar conveniente.

Pacote *control*


 GameController
<ul style="list-style-type: none">+ Graphics drawAllElements(ArrayList<Element> elemArray, Graphics g)+ ArrayList<Element> processAllElements(ArrayList<Element> e)+ Element isValidPosition(ArrayList<Element> elemArray, Element elem)




 GameScreen (JFrame)
<ul style="list-style-type: none"> - Lolo lolo - ArrayList<Element> elemArray - GameController controller
<ul style="list-style-type: none">◆ +GameScreen (JFrame)()+ Element addElement(Element elem)+ Element removeElement(Element elem)+ Graphics paint(Graphics gOld)+ void go()+ KeyEvent keyPressed(KeyEvent e)+ KeyEvent keyTyped(KeyEvent e)+ KeyEvent keyReleased(KeyEvent e)

 Main
<ul style="list-style-type: none">+ <u>static String[] main(String[] args)</u>








Pacote *utils*

Drawing














 ~static GameScreen (JFrame) screen

-  +static GameScreen (JFrame) getGameScreen()
-  +static GameScreen (JFrame) setGameScreen(GameScreen newScreen)
-  +static double draw(Graphics g, ImageIcon imageIcon, double y, double x)

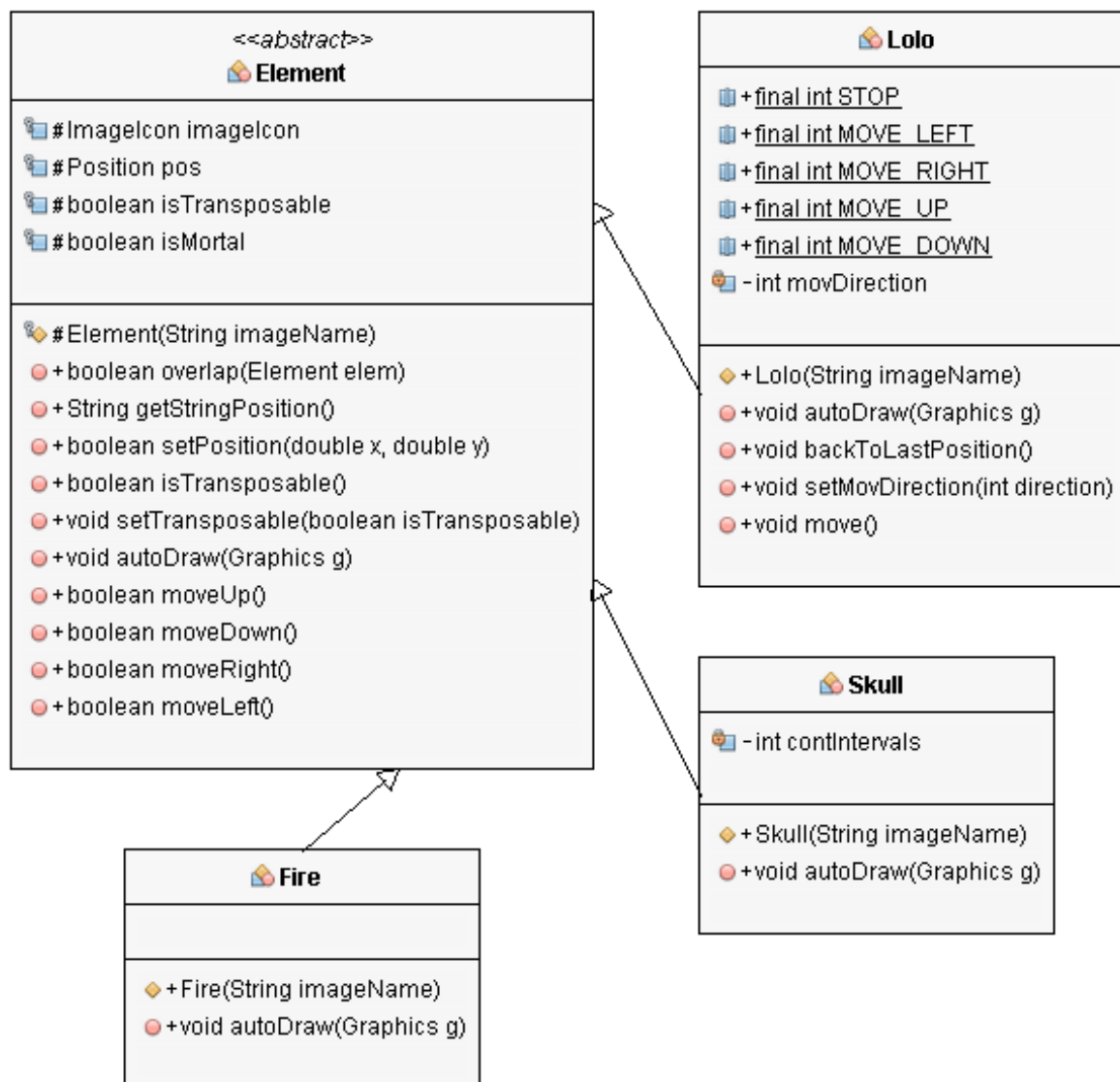
Consts

-  +final int CELL_SIZE
-  +final int NUM_CELLS
-  +final int WALK_STEP_DEC_PLACES
-  +final double WALK_STEP
-  +final String PATH
-  +final int DELAY
-  +final int TIMER_FOGO

Position

-  - double x
-  - double y
-  - double previousX
-  - double previousY
-  +Position(double x, double y)
-  +double setPosition(double x, double y)
-  +double getX()
-  +double getY()
-  +boolean comeBack()
-  +boolean moveUp()
-  +boolean moveDown()
-  +boolean moveRight()
-  +boolean moveLeft()

Pacote *elements*



Classes Obrigatórias

- 1) O projeto tem uma classe abstrata *Element*, da qual devem ser derivados todos os objetos do jogo. Crie classes para o Pacaman e para os fantasmas que herdem da classe *Element*. Crie também uma classe *BackgroundElement* para representar os elementos do cenário. Analise como deve ser a estrutura dessas classes para que se aproveite bem os conceitos de POO. Se achar necessário, você pode alterar a estrutura da classe *Element*.
- 2) Crie uma classe *Stage* para representar uma fase. Ela deve conter os objetos do cenário e os métodos que achar necessário.

Dicas

- 1) Você pode utilizar screenshots do próprio jogo e recortá-los em algum editor de imagens (paint, gedit, photoshop, ...) para obter as imagens do jogo, ou obtê-las diretamente na internet.
- 2) Estude com calma a relação entre as classe no framework fornecido. O bom entendimento facilita a criação do seu projeto.
- 3) Teste cada nova funcionalidade. É muito mais difícil identificar erros depois de inserir muitas funcionalidades.
- 4) Seja criativo! Tente colocar mais funcionalidades além das exigidas, tais como animações, sons, etc.

Composição dos Grupos

O trabalho deve ser feito em grupos de 2 a 4 pessoas. Informar por email até o dia 26/09/2017 os componentes dos grupos.

Data limite para entrega: 10/12/2017.

Local de entrega: Tidia 4.0 (Atividades).

Forma de entrega:

Um arquivo zip contendo

- 1) Projeto do Netbeans (ou em outro formato, com instruções de compilação)
- 2) Arquivo PDF contendo:
 - Nome e número USP de cada integrante do grupo.
 - Diagrama de classes simplificado (heranças, implementações, principais métodos/atributos de cada classe)
 - Quaisquer esclarecimentos que acharem necessários.

A discussão de ideias entre colegas é estimulada, porém a solução do trabalho é individual (do grupo). A identificação de plágio acarretará em nota zero para todos os envolvidos.