Departamento de Ciência da Computação – Universidade de Brasília
CIC 113956 Programação Sistemática – 1/2016
Data de entrega: 14/04/2016, 23:59h
Trabalho Prático - Parte I

Introdução

Planejamento e organização são práticas indispensáveis no cotidiano de um bom cientista ou engenheiro. Sabemos que, em um ambiente profissional, essas características devem estar presentes da forma mais precisa possível. O motivo se dá pelo fato de que, em eventos de grande escala, tem-se um grande número de informações que precisam ser manipuladas. Assim sendo, um dos focos do T1 da disciplina é a organização do trabalho em equipe. Visando a aplicação dos conceitos aprendidos na disciplina de Programação Sistemática, a criação de um jogo como o **Tetris** irá guiá-los no amadurecimento de suas habilidades, não só como programadores, mas também como grupo. Sempre aplicando **todos** os conceitos aprendidos durante o andamento da disciplina.

Regras e Funcionamento do Jogo

1. Dimensões: Os limites da tela do tetris devem estar claros, ou seja, a borda que delimita o tamanho do mesmo deve ser visível. Tamanho: matriz 15x25 ([y][x]). Além das bordas, o tetris conterá um *limite superior* localizado na linha 5 da matriz do tetris (considerando a primeira linha de cima para baixo como a linha zero). Tal limite deve estar indicado na interface do jogo.

2. Peças: Esta versão do jogo conterá apenas peças retas. Tais peças poderão estar em duas posições: vertical e horizontal.

3. Cores: As peças devem ser coloridas (A cada nova peça, uma cor diferente). Lembre-se que uma peça não pode possuir a mesma cor do fundo ou da tela(bordas) do jogo em execução. Por último, a cor da tela de *início* e *fim de jogo* também deverá ser diferente da cor da tela em execução.

4. Movimentação: Direita, esquerda e baixo. Não pode haver a opção de subir. *Entrada:* Setas direcionais do teclado. Obs.: O movimento de rotação não está permitido nesta versão do tetris.

5. As peças não devem ultrapassar os limites laterais, inferior ou superior do jogo durante a movimentação.

6. Atualização e posição: Quando a peça atingir a borda inferior ou uma outra peça, ela automaticamente deve se fixar nesta posição. Assim que uma peça for fixada, uma nova peça deverá surgir automaticamente no centro superior da tela.

7. As peças deverão aparecer de forma aleatória. Assim, uma mesma peça poderá aparecer

infinitas vezes até que o jogador perca a partida. A posição da peça também será de forma aleatória, ou seja, uma mesma peça pode aparecer horizontalmente ou verticalmente. Surgimento Vertical: início da peça em [0][13]. Horizontal: [0][x] … x varia com o tamanho da peça, mas com o foco no centro da linha. *Notação: [linha][coluna].*

8. Quando um jogador completar todas as colunas de uma linha da tela, esta linha deverá ser limpa dando lugar para que as peças de cima caiam. Só após toda esta movimentação ocorrer que outra peça aparecerá. *Saída:* quando o jogador conseguir completar uma linha, ele ganha automaticamente 100 pontos. A pontuação deve ser atualizada em tempo real.

9. Término da partida: A partida acabará somente quando o jogador atingir o *limite superior* [5][x] (onde a contagem do eixo y começa de cima para baixo) da área onde as peças caem.

10- Inicio: O jogo conterá uma tela de início. O jogo só iniciará quando uma tecla for apertada.

Fim de jogo: O jogo deverá mostrar uma tela de fim de jogo, contendo a pontuação, o tempo da partida e uma mensagem de encerramento.

Modularização e Instalação da Ncurses

Nesta primeira parte do trabalho, uma versão simplificada do Tetris será implementada, e, para cada módulo de implementação criado (arquivo .c), deverá ser criado também um módulo de definição (arquivo .h), de forma que a lincagem ocorra sem problemas quando o makefile for criado. Lembrem-se também de utilizar diretivas de controle para evitar múltiplas inclusões!

Para auxiliar na usabilidade do jogo, vocês deverão utilizar a biblioteca **ncurses** para realizar a movimentação das peças, manipular cores de caracteres e fundos, mostrar o jogo na tela, construir a borda do tetris, etc. Ela pode ser instalada através do seguinte comando no terminal (em ambiente Linux):

sudo apt-get install libncurses-dev

Mais informações sobre a biblioteca podem ser encontradas no slides disponibilizados no Aprender da disciplina de Programação Sistemática e também aqui:

http://www.gnu.org/software/ncurses/

Módulo 1(Tela)

O módulo 1 será responsável por criar a matriz que servirá como tela para o jogo, onde as peças se movimentarão.

A matriz deve ter o tamanho 15x25 ([y][x]).

As telas de início e fim de jogo podem ser criadas onde for mais usual.

Funções:

/* A função “cria_tela” deve criar a interface (bordas), lugar onde ocorrerá toda a dinâmica do jogo. */

• Tela* cria_tela();

/* A função “mostra_tela” será chamada sempre que o jogo for iniciado ou alguma mudança ocorrer dentro da matriz principal, pois tal mudança só irá ser percebida pelo jogador quando ela for mostrada na saída. */

• void mostra_tela(Tela* t);

Módulo 2(Peças)

O módulo 2 será responsável por criar as peças do jogo e realizar seus movimentos dentro da tela(matriz).

As peças:

- Terão tamanho variável entre 3 e 5 caracteres.
- Serão guardadas usando vetores.
- Serão formadas por caracteres. Ex: 'ooooo', ou os mesmos caracteres rearranjados na posição vertical.

Funções:

/* A função “nova_peça” mostra uma nova peça na parte superior do tetris. */

• void nova_peça(Tela* tela);

/* A função “move_peça_x” move a peça para a direita e esquerda (eixo x). O parâmetro inteiro x é a coordenada foco, para onde a peça deverá se mover. */

• void move_peça_x(peça* peça, int x);

/* A função “move_peça_y” move a peça para baixo (eixo y). O parâmetro inteiro y é a coordenada foco, para onde a peça deverá se mover. */

• void move_peça_y(peça* peça, int y);

Módulo 3(Engine)

O módulo 3 será responsável por ler o input do teclado a cada jogada e associar com uma ação, para então, mostrar a movimentação na tela, além de inicializar e finalizar o uso da ncurses. Portanto, este será o módulo com maior acoplamento do trabalho, sendo cliente dos outros dois módulos anteriores.

Funções:

/* Todos os recursos da biblioteca ncurses deverão ser declarados/usados entre as duas funções a seguir. */

• void inicia_ncurses();

• void finaliza_ncurses();

/* A função “pega_input”, do tipo inteiro, é a função que será responsável por ler a entrada do teclado. Seu retorno é o valor da leitura que poderá ser usado em outras funções, quando necessário. */

- int pega_input(int input);

Controle de qualidade das funcionalidades

Deve-se criar um módulo controlador de teste (disciplinado) usando o CUnit (C) para testar se as principais funcionalidades e restrições dos módulos atendem a especificação. O teste disciplinado deve seguir os seguintes passos:

1. Antes de testar: produzir um roteiro de teste;

2. Antes de iniciar o teste: estabelecer o cenário do teste;

3. Criar um módulo controlador de teste, usando a ferramenta CUnit (C) para testar as principais funcionalidades de cada módulo;

4. Ao testar: produzir um laudo em que todas as discrepâncias encontradas são registradas. Esse laudo pode ser uma saída da execução do CUnit (C) . Somente termine o teste antes de completar o roteiro, caso observe que não vale mais a pena continuar executando o roteiro, uma vez que o contexto para o resto está danificado;

5. Após a correção: repetir o teste a partir de 2 até o roteiro passar sem encontrar falhas.

Parte Escrita e Observações Finais

Visando facilitar o trabalho em grupo, vocês devem utilizar o repositório **GitHub** para interagir e atualizar o código a cada etapa do desenvolvimento. Vide slides introdutórios sobre o GitHub no Aprender da disciplina de Programação Sistemática.

O arquivo .zip de submissão deve conter:

- Todos os arquivos necessários para a compilação e execução do programa, incluindo os módulos controladores de teste.
- Um ReadMe.txt contendo:
  - As instruções para a compilação e execução correta do trabalho. Lembrem-se que o programa será avaliado em uma distribuição Linux.
  - O link do projeto no GitHub.
- Um documento textual explicando a arquitetura utilizada contendo os modelos conceituais e físicos
- Os gráficos gerados pelo **GitHub** com as estatísticas do desenvolvimento do projeto contendo as horas trabalhadas **por cada membro do grupo** e as respectivas descrições das tarefas que **cada membro realizou**.