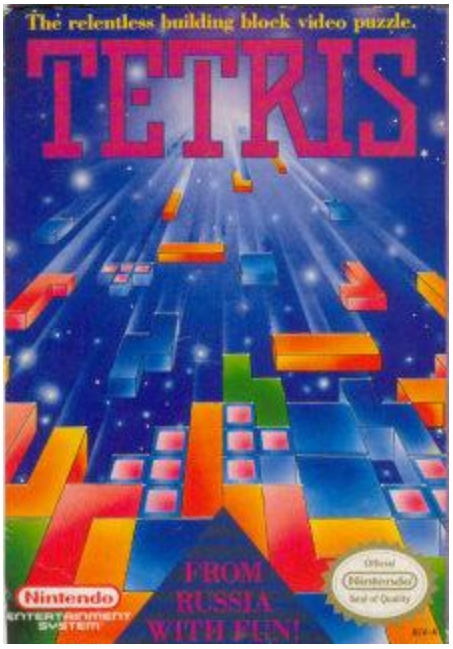


Fonte: Wikipedia

Trabalho 2: data limite para submissao: **06/04/2021 as 17:59**
O trabalho deverá ser feito em dupla

Obs: caso o aluno queira fazer algum trabalho diferente (exemplo: snake, algum jogo similar ao Mario, pacman, jogo de nave 2D, etc) favor conversar com o professor (o mais importante é que o trabalho desenvolvido use recursos similares aos que vamos utilizar no Tetris -- transformações, visualização, viewport, projeções, menu simples, etc). Usem a criatividade.

Regras gerais de trabalhos de INF390:
https://docs.google.com/document/d/1yvvfR6TrK3B9wBg7rHQNwHIJZ34I01rBtq0P7wFSrRA/edit (atenção: **leia TUDO**, mesmo se já tiver cursado alguma disciplina comigo)

Um jogo simples que fez muito sucesso (e continua fazendo) e' o Tetris (https://en.wikipedia.org/wiki/Tetris). Esse jogo foi criado nos anos 80 por Alexey Pajitnov e rapidamente se espalhou pelo mundo em computadores, videogames, aparelhos portáteis, etc.

Na disciplina de Estrutura de Dados (alguns de) vocês criaram uma versão em modo texto do Tetris utilizando a biblioteca *ncurses*. Neste trabalho iremos criar uma versão gráfica dele.

Leia este roteiro com bastante atenção antes de começar sua implementação. Antes de submeter a versão final do seu trabalho leia este roteiro novamente e confira se sua implementação seguiu toda a especificação e se você está fazendo a entrega de forma correta.

Aviso: apesar desta especificação ter ficado grande (devido aos detalhes), sua implementacao do trabalho provavelmente ficará pequena! Ele é mais simples do que aparenta.

Aviso 2: **comece sua implementacao com antecedencia! Cuidado com o deadline!**

A seguir há a especificação do trabalho. Você pode adicionar outras funcionalidades ao seu trabalho caso queira.

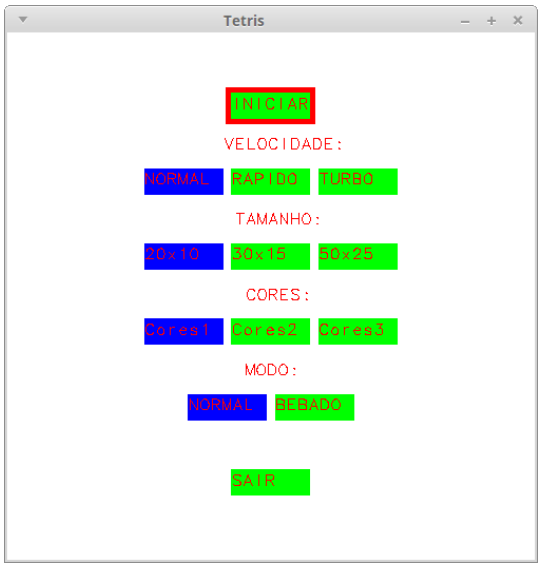
O Menu

Ao abrir o programa, ele deve entrar primeiramente em um menu. O jogo só deverá se iniciar após o usuário selecionar o botão “Iniciar jogo”. Além disso, se durante o jogo o usuário pressionar a tecla “ESC” o menu deverá ser exibido (não é necessário voltar ao estado anterior do jogo caso você vá ao menu e volte para ele) e os botões marcados como selecionados no menu devem corresponder as opções selecionadas anteriormente.

Os botões poderão ser selecionados usando tanto o mouse quanto as teclas direcionais/enter do teclado.

O menu deverá possuir 6 grupos de botões:
1) Velocidade: onde o jogador poderá escolher a velocidade do jogo (lenta, normal, rápida).
2) Esquemas de cor: onde o jogador poderá escolher uma de 3 opções de estilos de cor no jogo (pelo menos a cor das peças e cor do fundo do jogo deverão ser alteradas com os diferentes estilos selecionados).
3) Tamanho do jogo: teremos pelo menos 3 opções para o tamanho (número de linhas e colunas) do jogo: 20x10, 30x15 e algum outro tamanho bem grande.
4) Modo: normal e bêbado (no modo bêbado a tela do jogo (não a do menu) deverá girar enquanto o jogador estiver jogando -- o professor mostrará um exemplo disso em sala). Se quiser, pode fazer tal modo usando algum outro efeito similar mais interessante (o importante é utilizar transformações do OpenGL).
5) Iniciar: inicia o jogo.
6) Sair: fecha o programa.

Veja o exemplo a seguir:



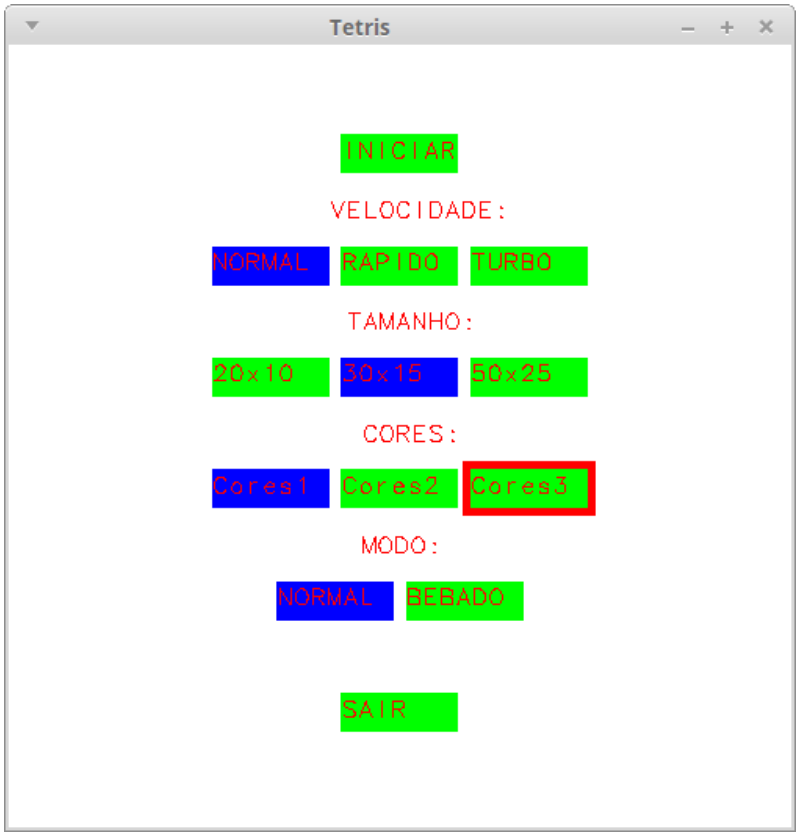
Os botões deverão ser selecionados utilizando ou o mouse (clicando neles) ou o teclado. Ao utilizar o teclado, as teclas “para cima” e “para baixo” devem ser utilizadas para selecionar cada um dos 4 grupos de botões. As teclas “esquerda” e “direita” devem ser utilizadas para selecionar uma das opções (quando existirem mais de uma) dentro do mesmo grupo. Se o usuário estiver no último grupo de botes e pressionar a tecla “para baixo” → o cursor deve ir ao primeiro grupo (e vice-versa) (algo similar deve ocorrer quando o usuário estiver no último/primeiro botão de um grupo).

Um retângulo maior que o botão deve ser utilizado para destacar a opção na qual “o teclado está” (na figura acima tal retângulo é vermelho). A tecla “Enter” deverá funcionar de forma similar a um clique do mouse no botão destacado.

Os botões relativos a opções que estiverem ativas devem ser coloridos com outra cor (para que o jogador sempre saiba quais opções estão atualmente selecionadas -- na figura acima eles estão coloridos de azul). Por exemplo, se ao abrir o jogo a velocidade padrão for “Normal”, tal botão deverá estar em uma cor diferente das outras. Se o usuario mudar a velocidade, o botão de tal velocidade deverá ter a cor alterada.

Veja os exemplos abaixo (seu Menu deverá ser parecido com os exemplos, mas não precisa ser exatamente igual (notem que não houve muito capricho no design dos exemplos...)):

(notem acima que o menu deve ficar centralizado e não distorcido quando a janela é redimensionada -- mais detalhes sobre isso adiante)



O Jogo

Nos arquivos de exemplo há uma implementação do Tetris (em modo texto) utilizando uma classe criada na disciplina de Estrutura de Dados (a compilação exige a biblioteca ncurses).

Tal implementação não é eficiente (foi propositalmente feita para forçar a alocação e desalocação de memória e usar muito construtores de cópia, etc) e não segue muitas práticas boas de programação, porém ela é aceitável para este trabalho.

Algumas informações sobre a implementação fornecida podem ser obtidas no seguinte arquivo: https://docs.google.com/document/d/1VUnNmL6WBgtL4LTtXeqJuAHu9Y0wBN0Q7wIiO8aakxE

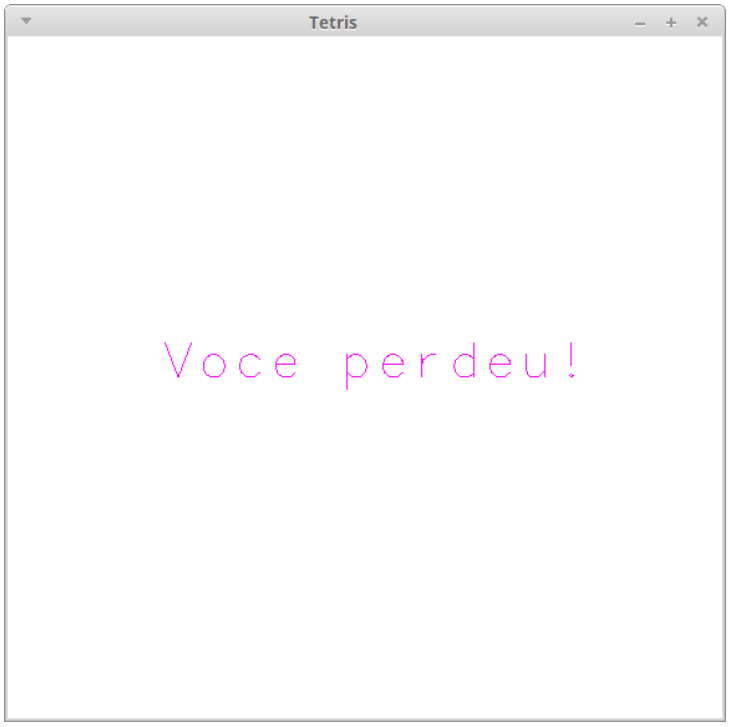
(se você não cursou Estrutura de Dados com o professor atual sinta-se à vontade para pedir mais informações sobre tal implementação)

Você poderá utilizar o que quiser da implementação fornecida (dica: deixe que a classe trate tudo relacionado a lógica do jogo -- seu programa em opengl precisará basicamente usar as funções get() para verificar o estado de cada quadrado que compõe as peças e a tela do jogo). Graças a isso seu código provavelmente ficará relativamente pequeno (basicamente você precisará apenas se encarregar de desenhar o jogo na tela, tratar eventos do teclado/mouse, trabalhar com o menu e manipular a janela).

A posição da peça deverá ser controlada pelas teclas (setas) de direção do teclado (a tecla “para baixo” deverá ser utilizada para acelerar o jogo). Se o usuário pressionar “shift” juntamente com as teclas “direita” ou “esquerda” a peça deverá se deslocar na horizontal de forma mais rápida do que o normal (isso permitirá que a peça alcance posições mais afastadas da tela quando a largura for maior do que o normal). A tecla “espaço” deverá ser utilizada para rotacionar as peças.

Adicionalmente, seu programa deverá suportar “zoom”: a ideia é que poderemos ter jogos grandes e, assim, o jogador poderá fazer zoom para ver melhor a peça. As teclas “z” e “x” deverão, respectivamente, fazer um “zoom in” e “zoom out” na tela. As teclas “a”,“s”,“d” e “w” deverão ser utilizadas como teclas direcionais para que o jogador possa mover a “câmera” na tela. (peça para o professor mostrar o uso desses recursos em sala).

Quando o usuário perder o jogo você deverá exibir alguma mensagem informando isso. A qualquer momento o usuario deve ser capaz de voltar ao menu utilizando a tecla ESC.



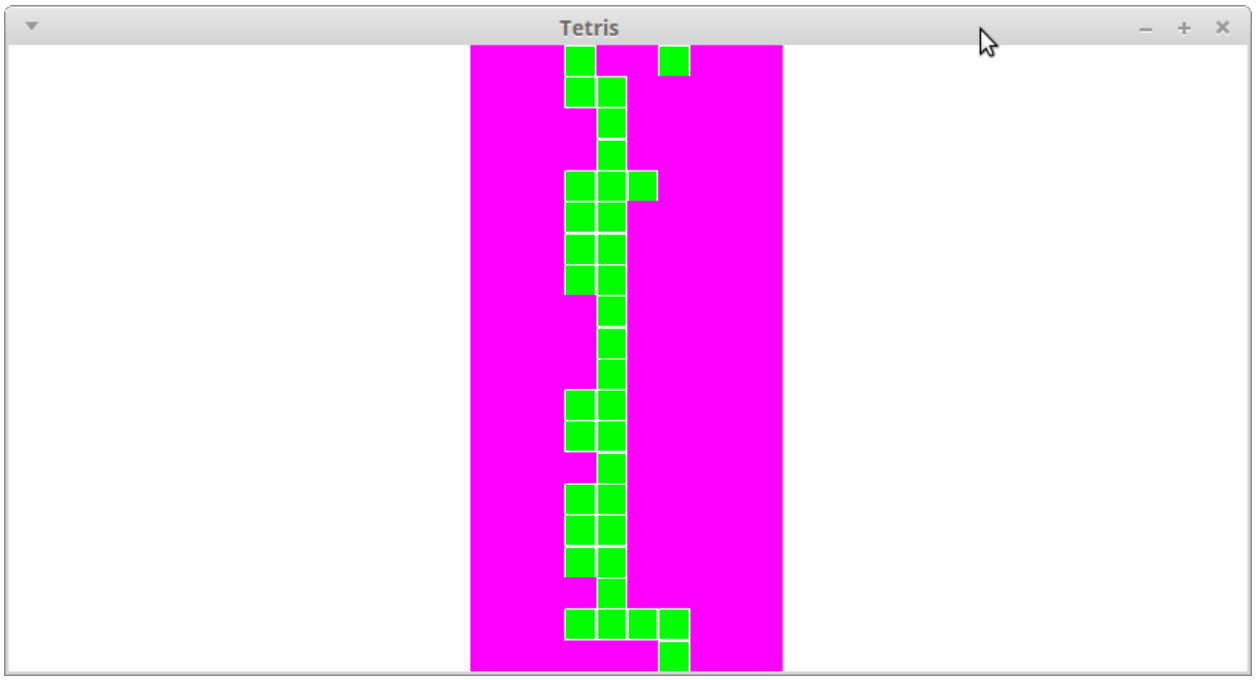
A janela do jogo/menu

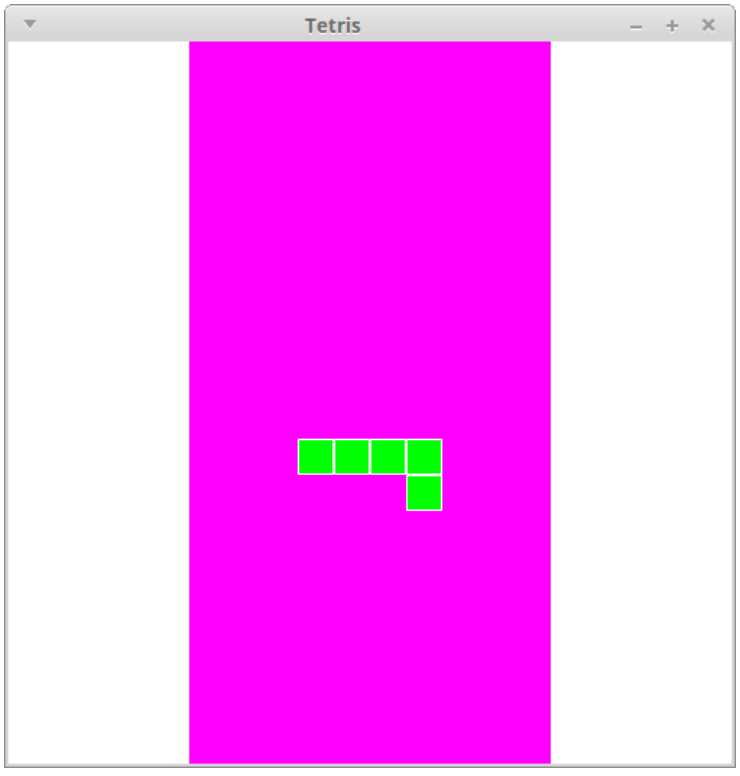
Suas peças devem ser formadas por quadrados e eles devem manter a forma (de quadrado) na tela (ou seja, não devem parecer retângulos).

A parte principal do seu jogo (e o menu) devem ficar centralizados na janela (do sistema operacional).

Tais propriedades devem ser mantidas mesmo se a janela for redimensionada pelo usuário (ou seja, mudanças na janela não devem distorcer o seu jogo).

Veja exemplos:


Tetris


Tetris

Dicas

Apesar disso em geral não ser uma boa prática de programação, devido ao estilo go OpenGL (com funções de callback) você provavelmente usará algumas variáveis globais para armazenar o estado do jogo. Isso é ok neste trabalho. Porem, tente manter seu codigo organizado!

Tente separar a parte responsável pela “lógica do jogo” da parte responsável por desenhá-lo na tela e manipular a entrada/saída.

Não misture a “renderização do jogo” com o gerenciamento da janela. Ou seja, ao desenhar o jogo o desenhe utilizando um espaço de coordenadas “real” e apenas se preocupe com redimensionamento de janela, projeção do jogo no viewport, etc em uma parte separada do seu programa!

Os arquivos INF390.cpp e INF390.h (disponíveis juntamente com o trabalho) foram criados pelo monitor Gustavo (que atuou na disciplina em períodos anteriores) e possui uma função que facilita a exibição de texto na tela. Teste-o e use-o para, por exemplo, colocar texto nos botões.

A função “glutTimerFunc” da biblioteca glut pode ser utilizada para chamar alguma função de callback de tempos em tempos (veja a documentação). Isso provavelmente será MUITO útil para seu trabalho. Por exemplo, você deverá “descer” com as peças de tempos em tempos (e, portanto, você pode utilizar um timer para atualizar o jogo e chamar a função de “display” de tempos em tempos). Além disso, você provavelmente precisará de um timer adicional com intervalo de tempo menor para fazer o efeito na qual a tela do jogo gira (no modo “bebado”).

Seu programa provavelmente usará os seguintes conceitos/funções:
- Transformacoes
- Desenho de poligonos
- Funcoes de callback do teclado/mouse
- gluOrtho2D
- glViewport
- glutTimerFunc
- Cores
- etc

Arquivo README

Seu trabalho deverá incluir um arquivo README.

Tal arquivo contera:

- Nome/matricula dos dois alunos
- Informacoes sobre fontes de consulta utilizadas no trabalho
- Informações adicionais sobre o jogo (exemplo: informações sobre alguma funcionalidade extra que você adicionou ao seu trabalho)

Makefile

Seu trabalho deverá ter um Makefile. Ao digitar “make” deverá ser gerado um arquivo executável com o nome “main.out”.(ele deverá compilar no linux)

Entrega

Envie seus arquivos pelo Submitty (código fonte, README, Makefile, etc).

Avaliacao

Seu programa deverá compilar/funcionar no Linux (sugere-se que você o desenvolva no Linux -- você deverá pelo menos testá-lo nesse Sistema Operacional antes de entregá-lo).

Duvidas

Dúvidas sobre este trabalho deverão ser postadas no sistema Piazza. Se esforce para implementá-lo e não hesite em postar suas dúvidas!

Avaliacao do codigo

O código do seu trabalho também será avaliado. Tente mante-lo bem organizado, com comentarios, etc.