	<p>Centro Tecnológico</p> <p>Departamento de Informática</p>
<p>Disciplina: Computação Gráfica</p>	<p>Código: INF09282 e INF09284</p>
<p>Prof. Thiago Oliveira dos Santos</p>	

Trabalho Curto 2

1 Introdução

Esse trabalho tem como objetivo aprimorar o conhecimento dos alunos em relação ao tópico de formato de arquivos gráficos e interatividade usando dispositivos gráficos e representação da informação visual.

Para isso, o aluno deverá implementar um programa que lerá informações descrevendo uma arena de um arquivo do tipo Scalable Vector Graphics (SVG), e desenhará a arena com seus respectivos elementos na tela. A arena deverá ser estática, exceto pelo personagem do jogador que deverá se mover pela pista se colidir com os objetos. O trabalho deverá ser implementado em C++ (ou C) usando as bibliotecas gráficas OpenGL e GLUT (freeglut).

2 Especificação das Funcionalidades


Ao rodar, o programa deverá ler, de um arquivo de configurações (denominado “config.xml”), as configurações necessárias para suas tarefas. O arquivo de configurações deverá estar no formato xml e será fornecido juntamente com a especificação do trabalho. A localização do arquivo “config.xml” será fornecida pela linha de comando ao chamar o programa. Por exemplo, se o arquivo estiver dentro de uma pasta chamada “Test1” localizada na raiz, basta chamar o programa com “/Test1/” como argumento (outros exemplos de caminhos possíveis “../Test1/”, “../../Test1/”, etc.). As informações contidas nesse arquivo servirão para ler o arquivo SVG contendo as informações da arena.

O arquivo de configurações deverá conter uma tag xml global <aplicacao> com uma sub-tag específica para descrever o arquivo de entrada da arena, denominada <arquivoDaArena>. A tag <arquivoDaArena> terá atributos para descrever o nome, a extensão, e o caminho do arquivo descrevendo a arena (“nome”, “tipo” e “caminho” respectivamente). Perceba que o nome e o caminho do arquivo SVG descrevendo a arena será especificado por esses 3 atributos combinados, e que os caminhos do arquivo podem contemplar combinações do tipo: “../Test1/”, ou “./Test1/”, ou “~/Test1/”, entre outros.

Exemplo do arquivo config.xml com um arquivo SVG dado por: ../Test1/arena.svg

```
<aplicacao>
  <arquivoDaArena nome="arena" tipo="svg" caminho="../Test1/"></arquivoDaArena>
</aplicacao>
```

Após ler as informações do arquivo de configurações, o programa deverá ler e interpretar os elementos da arena do arquivo do tipo SVG respectivo e desenhá-los na tela. A arena será composta por uma série de elementos (ver Figura 1): uma pista descrita por dois círculos, um círculo azul descrevendo o limite exterior e um círculo branco descrevendo o limite interior; um círculo verde representando o personagem do jogador; pelo menos um círculo vermelho representando os personagens dos inimigos; e um retângulo preto representando o local da largada e da chegada. Todos esses elementos estarão contidos no arquivo SVG descrevendo a arena. Um arquivo SVG será fornecido como exemplo juntamente com a descrição do trabalho.

	Centro Tecnológico Departamento de Informática	
Disciplina: Computação Gráfica		Código: INF09282 e INF09284
Prof. Thiago Oliveira dos Santos		

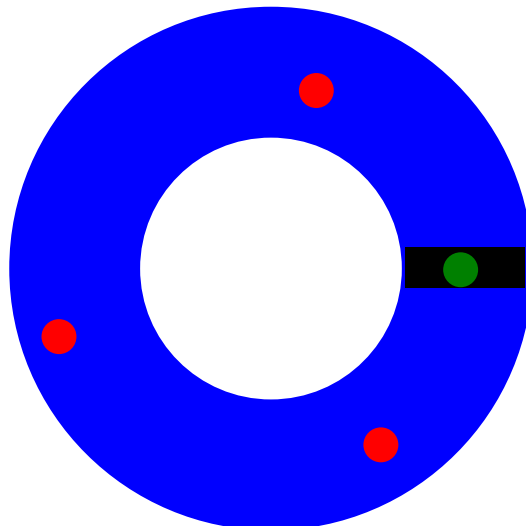


Figura 1: Exemplo de arena com seus elementos.

Cada um dos elementos da arena, contidos no SVG, tem informação suficiente para desenhá-los na tela (ex. posição, tamanho, cor, etc.). A arena deverá ser desenhada em uma tela do tamanho do diâmetro do círculo externo da pista. Os outros elementos da arena deverão ser desenhados em cima da arena considerando as posições, tamanhos e cores lidos do arquivo SVG. Todos os objetos serão estáticos exceto pelo personagem do jogador (i.e. círculo verde). O círculo do jogador deverá se mover ao pressionar as teclas do “w”, “s”, “a” e “d” respectivamente para cima, para baixo, para esquerda e para direita. Os movimentos deverão ser contínuos e permitir combinações de teclas para movimentos na diagonal (ex. ao manter “a” e “w” pressionados, o objeto deverá ser mover na diagonal esquerda para cima) conforme mostrado em laboratório. Colisões deverão ser tratadas para o personagem do jogador. Ele não poderá sair da pista (ou seja, extrapolar o círculo azul ou entrar no círculo branco) e nem deverá sobrepor o círculo dos inimigos, então ele deverá parar antes disso acontecer.

3 Regras Gerais


O trabalho deverá ser feito individualmente. Trabalhos identificados como fraudulentos serão punidos com nota zero. Casos típicos de fraude incluem, mas não se restringem à cópias de trabalhos, ou parte dele, assim como trabalhos feitos por terceiros. Caso seja necessário confirmar o conhecimento do aluno a respeito do código entregue, o professor poderá pedir ao aluno para apresentar o trabalho oralmente em um momento posterior. A nota da apresentação servirá para ponderar a nota obtida no trabalho.

3.1 Entrega do Trabalho

O código deverá ser entregue por email (para: todsantos@inf.ufes.br) dentro do prazo definido no portal do aluno. Trabalhos entregues após a data estabelecida não serão corrigidos.

A entrega do trabalho deverá seguir estritamente as regras a seguir. O não cumprimento acarretará na **não correção do trabalho** e respectivamente na atribuição da nota zero.

- Assunto da mensagem: [CG-2016-2] <tipo do trabalho>. Onde, <tipo do trabalho> pode ser TC1, TC2, TC3 e representa respectivamente trabalho curto 1, 2, 3, etc , ou TF para o trabalho final.
- Anexo da mensagem: arquivo zippado (com o nome do autor, ex. FulanoDaSilva.zip) contendo todos os arquivos necessários para a compilação do trabalho;
- Não enviar arquivos já compilados, inclusive bibliotecas!
- O diretório deverá necessariamente conter um makefile que implemente as seguintes diretivas "make clean" para limpar arquivos já compilados, "make all" para compilar e gerar o executável. O executável deverá ser chamado *trabalhocg*.

	Centro Tecnológico Departamento de Informática	
Disciplina: Computação Gráfica		Código: INF09282 e INF09284
Prof. Thiago Oliveira dos Santos		

Lembre-se que a localização do arquivo config.xml será passada via linha de comando e portanto não se deve assumir que haverá um arquivo desses na pasta do executável. Seja cuidadoso ao testar o seu programa, isto é, não teste com o arquivo no diretório do programa, pois você pode esquecer de testa-lo em outro lugar posteriormente.

3.2 Pontuação

O trabalho será pontuado conforme a tabela abaixo. Bugs serão descontados caso a caso.

Funcionalidade	Peso
Ler e desenhar a arena e seus elementos corretamente	4
Responder corretamente aos movimentos	2
Tratamento da colisão com inimigos	2
Tratamento da colisão com as bordas da pista	2

4 Erratas

Qualquer alteração nas regras do trabalho será comunicada em sala e no portal do aluno. É de responsabilidade do aluno frequentar as aulas e se manter atualizado.