

# LEMURYA

Trabalho acadêmico da disciplina  
de técnicas de programação.

Gabriel Henrique Linke  
Pedro Sodré dos Santos

# TABELA DE REQUISITOS

- 100% cumpridos

N.	Requisitos Funcionais	Situação	Implementação
1	Apresentar menu de opções aos usuários do Jogo.	Requisito previsto inicialmente e realizado.	Requisito cumprido via classe Menu (abstrata) e os respectivos objetos de suas subclasses (Menu Principal, Menu Pause e Menu Morte).
2	Permitir um ou dois jogadores aos usuários do Jogo, sendo que no último caso seria para que os dois joguem de maneira concomitante.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via classe Player cujos objetos são agregados em jogo, podendo ser um ou dois efetivamente.
3	Disponibilizar ao menos duas fases que podem ser jogadas sequencialmente ou selecionadas.	Requisito previsto inicialmente e realizado.	Requisito cumprido, inclusive com 3 fases, sendo através de objetos das classes (FaseAquatica1, FaseAquatica2 e FaseNoturna3).
4	Ter três tipos distintos de inimigos (o que pode incluir 'Chefão', vide abaixo), sendo que pelo menos um dos inimigos deve ser capaz de lançar projétil contra o(s) jogador(es).	Requisito previsto inicialmente e realizado.	Requisito cumprido, criando objetos das classes, Tritão, Esqueleto e Mago, sendo este último, o chefão capaz de lançar bolas de fogo (projétil).

# TABELA DE REQUISITOS

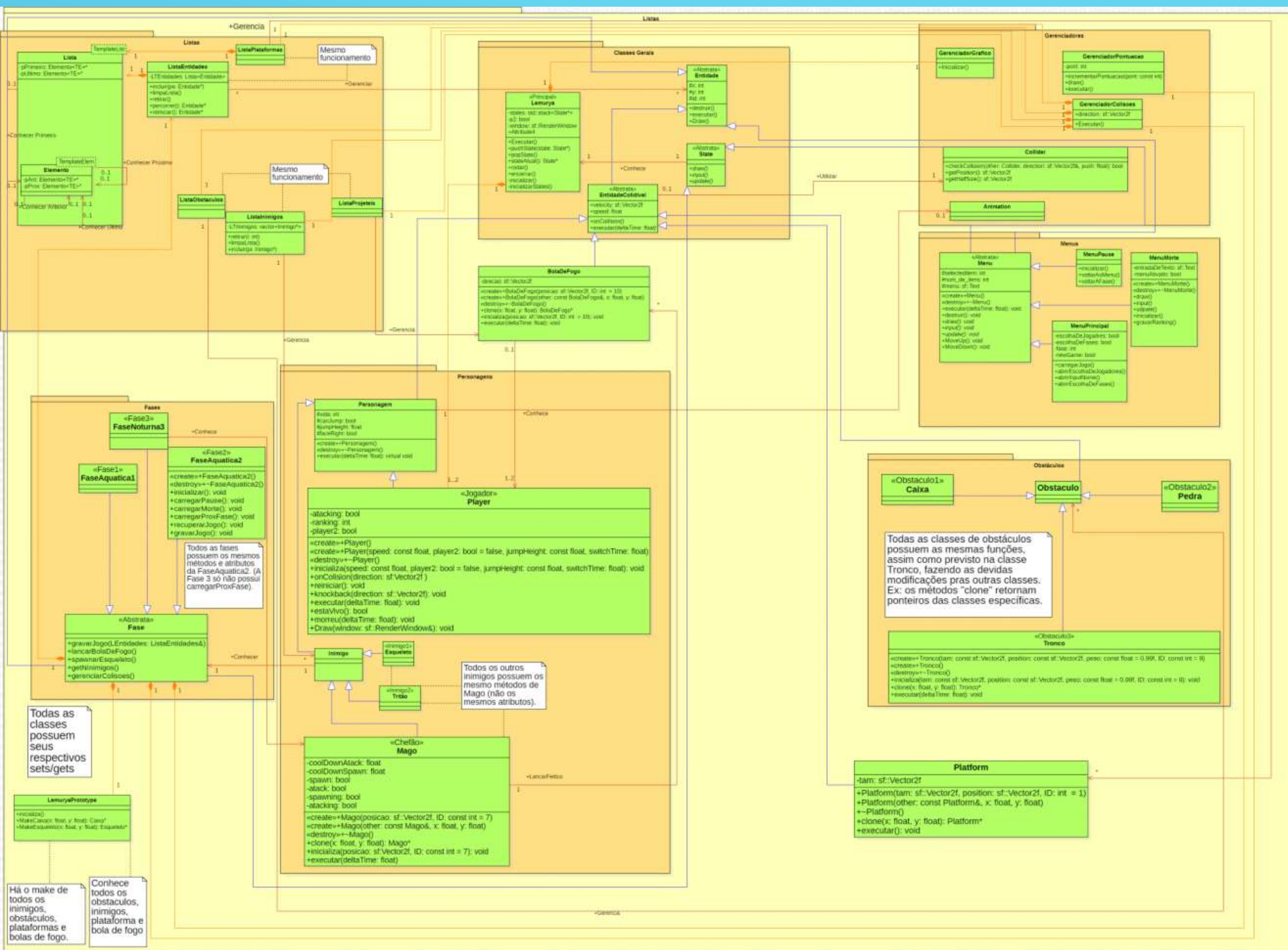
5	Ter a cada fase ao menos dois tipos de inimigos com número aleatório de instâncias, podendo ser várias instâncias e sendo pelo menos 5 instâncias por tipo.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via método gerarInimigos, definido na classe abstrata Fase e redefinido nas suas classes derivadas.
6	Ter inimigo "Chefão" na última fase	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via objeto da classe Mago que é instanciado na última fase (FaseNoturna3).
7	Ter três tipos de obstáculos.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via classes Tronco, Pedra e Caixa cujos objetos são agregados na fase através da classe LemuryaPrototype.
8	Ter em cada fase ao menos dois tipos de obstáculos com número aleatório de instâncias (i.e., objetos) sendo pelo menos 5 instâncias por tipo.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via método gerarObstaculos definido na classe Fase e redefinido nas suas classes derivadas.
9	Ter representação gráfica de cada instância.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive com a utilização da biblioteca gráfica SFML e via classe GerenciadorGrafico.
10	Ter em cada fase um cenário de jogo com os obstáculos.	Requisito previsto inicialmente e realizado.	Requisito cumprido via classe abstrata Fase e suas derivadas, com seus obstáculos sendo agregados através dos métodos gerarObstaculos, novoJogo e recuperarJogo.

# TABELA DE REQUISITOS

11	Gerenciar colisões entre jogador e inimigos, bem como seus projeteis (em havendo).	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via classes Collider, que verifica a colisão, e o Gerenciador de Colisões, que verifica as colisões percorrendo todas as listas de entidades que são colidíveis.
12	Gerenciar colisões entre jogador e obstáculos.	Requisito previsto inicialmente e realizado.	Idem para o requisito acima.
13	Permitir cadastrar/salvar dados do usuário, manter pontuação durante jogo, salvar pontuação e gerar lista de pontuação (ranking).	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via classes GerenciadorDePontuação, atributo ranking do jogador, método do menu principal que gera a lista de pontuação.
14	Permitir Pausar o Jogo	Requisito previsto inicialmente e realizado	Requisito cumprido inclusive via classe MenuPause, que é instanciada caso o usuário pressione a tecla ESC.
15	Permitir Salvar Jogada.	Requisito previsto e realizado	Requisito realizado via classe MenuPause, e método checkSalvarJogo das classes derivadas de Fase, que permite salvar a jogada.

# DIAGRAMA DE CLASSES UML






# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- 89% dos conceitos foram utilizados


N.	Conceitos	Uso	Onde / O quê
1	Elementares:		
	- Classes, objetos. & - Atributos (privados), variáveis e constantes. & - Métodos (com e sem retorno).	Sim	Todos .h e .cpp
	- Métodos (com retorno const e parâmetro const). & - Construtores (sem/com parâmetros) e destrutores	Sim	Todos .h e .cpp
	- Classe Principal.	Sim	main.cpp & Lemurya.h/.cpp
	- Divisão em .h e .cpp.	Sim	No desenvolvimento como um todo.

# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS


- Relações de:
    - Associação direcional & biderional
    - Agregação via associação & Agregação propriamente dita
    - Herança Elementar & Herança em diversos níveis
    - Herança múltipla
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.




# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Ponteiros, generalizações e exceções
    - Operador this.
    - Alocação de memória (new & delete)
    - Gabaritos/Templates criada/adaptados pelos autores
    - Uso de Tratamento de Exceções (try catch)
- 
- A series of three parallel white diagonal lines in the bottom right corner of the slide.

# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Virtualidade
    - Métodos virtuais
    - Polimorfismo
    - Métodos virtuais puros / Classes Abstratas
    - Coesão e desacoplamento
- 
- A series of several parallel white diagonal lines in the bottom right corner of the slide, pointing towards the bottom right.

# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Organizadore e estáticos:
    - Namespace
    - Nested criada pelos autores
    - Atributos estáticos e métodos estáticos
    - Uso extensivo de constante(const)  
parâmetro, retorno, método
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide.


# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Standard Template Library(STL) e String OO:
  - A classe Pré-definida String ou equivalente. &
  - - Vector e/ou List da STL (p/ objetos ou ponteiros de objetos de classes definidos pelos autores)
- Programação Concorrente:
  - - *Threads*


# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Biblioteca Gráfica / Visual:
  - Funcionalidades Elementares.
  - Funcionalidades Avançadas como:
    - tratamento de colisões
    - duplo buffer
    - Programação orientada a evento em algum ambiente gráfico.
- Interdisciplinaridades por meio da utilização de Conceitos de Matemática e/ou Física
  - Ensino Médio
  - Ensino Superior

# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Engenharia de Software
    - Compreensão, melhoria e rastreabilidade de cumprimento de requisitos.
    - Diagrama de Classes em *UML*.
    - Uso efetivo de padrões de projeto (particularmente GOF).
    - *Testes a luz da Tabela de Requisitos e do Diagrama de Classes.*
- 

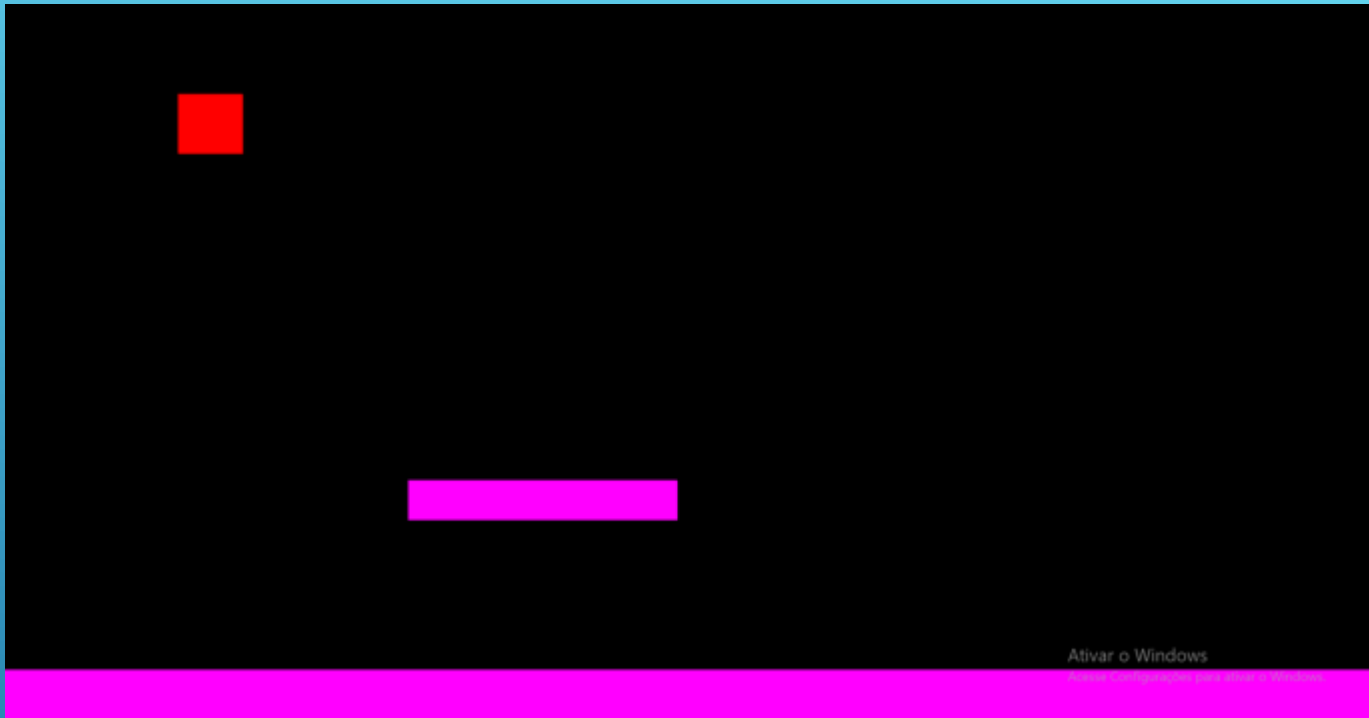
# TABELA DE CONCEITOS UTILIZADOS E NÃO UTILIZADOS

- Execução do Projeto
    - Controle de versão de modelos e códigos automatizado (via SVN e/ou afins) OU manual (via cópias manuais).
    - Uso de alguma forma de cópia de segurança (backup).
    - *Reuniões com o professor*
    - *Reuniões com o monitor*
    - *Revisão do trabalho escrito por outra equipe*
- 

# DESENVOLVIMENTO DO JOGO







INÍCIO DO JOGO



COLOCADO PERSONAGEM NA  
TELA



ADICIONADO MAPA E  
BACKGROUND



ADICIONADOS OBSTÁCULOS



ADICIONADOS INIMIGOS

# Lemurya

New Game

Load Game

Exit

Ativar o Windows  
Antes Configurações para ativar o Windows.

ADICIONADO O PRIMEIRO  
MENU





ADICIONADO PROJÉTIL



ADICIONADA PONTUAÇÃO



**RESULTADO FINAL**





# MENU PRINCIPAL



RANKING



Select number of players:

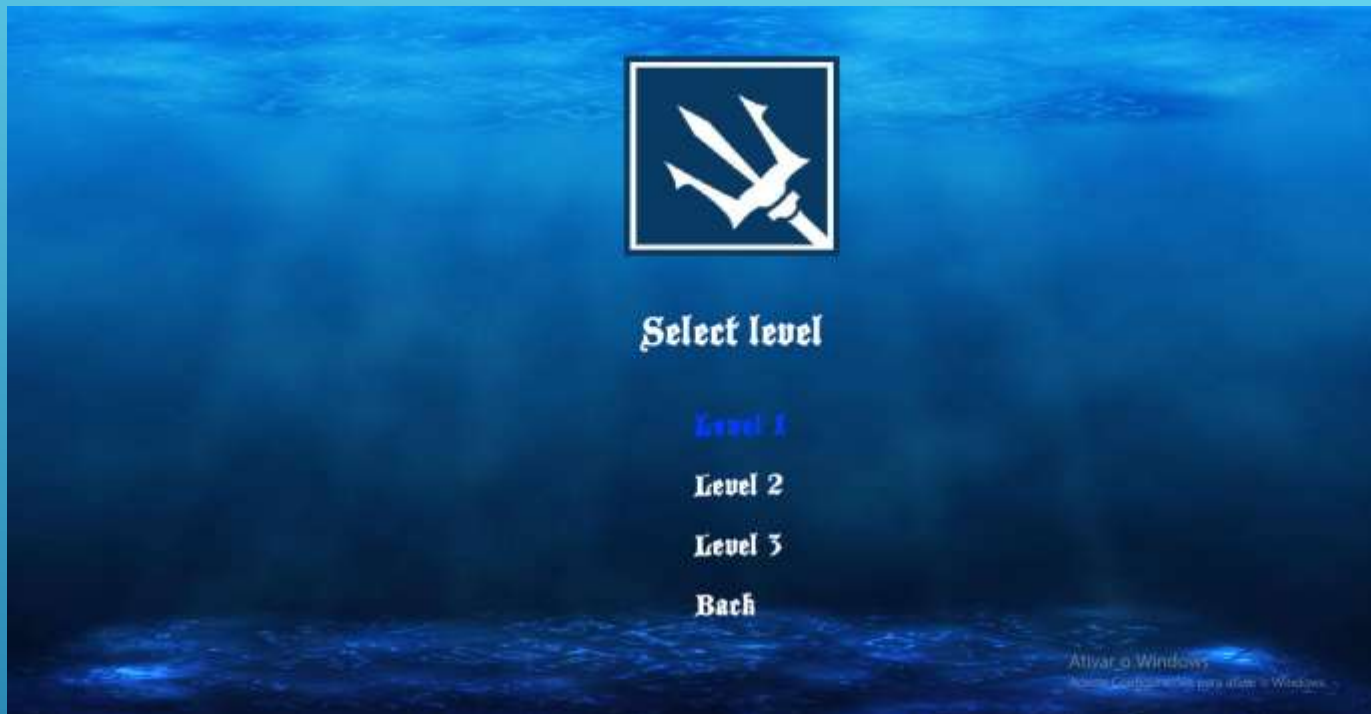
Solo

Coop

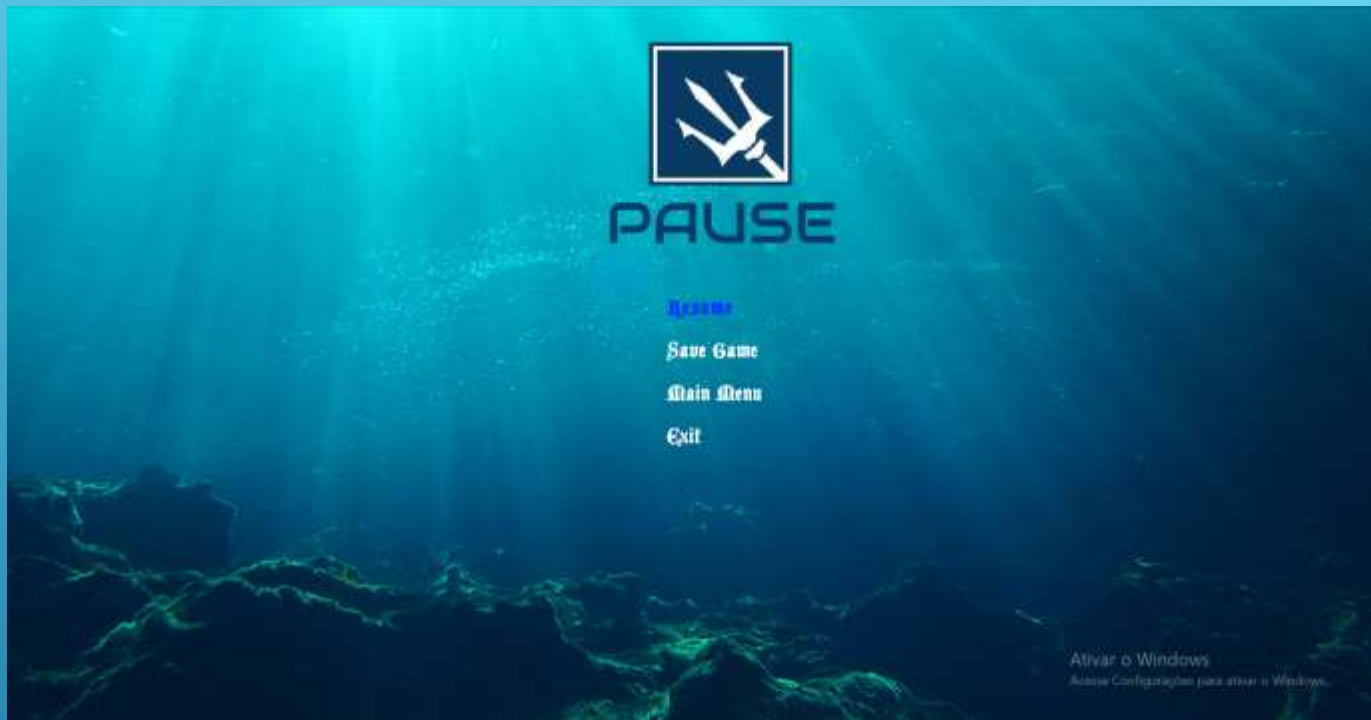
Back

Ativar o Windows  
Selecione as configurações para ativar o Windows

# ESCOLHA DE PLAYERS



ESCOLHA DA FASE



# MENU DE PAUSE



FASE 1





FASE 2





FASE 3



MENU DE MORTE



# MENU DE VITÓRIA

CONCLUSÃO

