TRABALHO PARA A DISCIPLINA DE
TÉCNICAS DE PROGRAMAÇÃO DO CURSO DE
ENGENHARIA DE COMPUTAÇÃO DA UTFPR:

Desenvolvimento do Jogo: “Ludi”

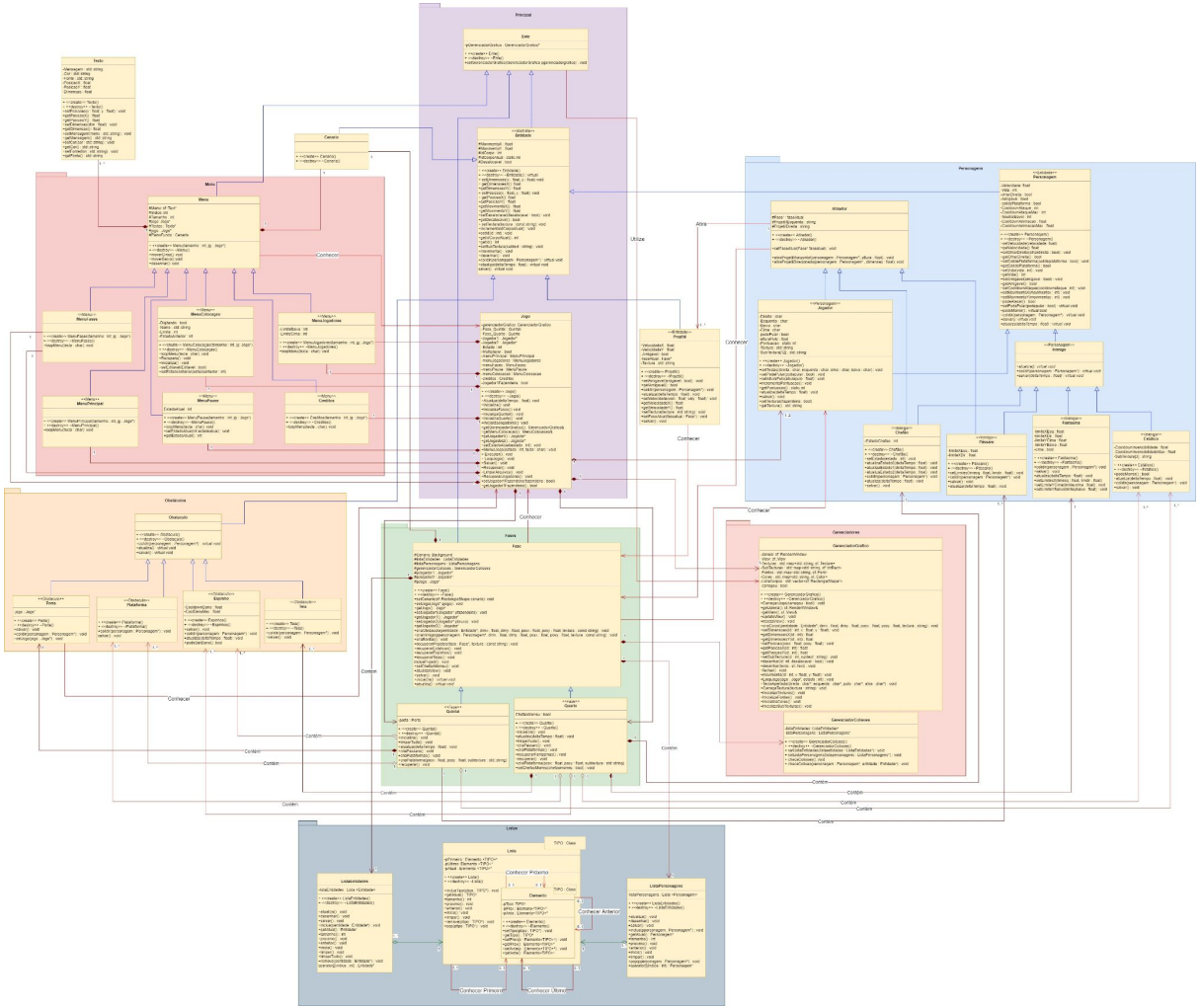
Professor Dr. Jean M. Simão
Turma S71
Daniel Augusto Pires de Castro
Francisco Cardoso Becheli

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

N.	Requisitos Funcionais	Situação	Implementação
1	Apresentar graficamente menu de opções aos usuários do Jogo, no qual pode se escolher fases, ver colocação (*ranking*) de jogadores e demais opções pertinentes.	Requisito previsto inicialmente e realizado.	Requisito cumprido via classe Menu e suas respectivas derivadas, cujos objetos foram implementados na classe principal Jogo, com suporte da SFML.
2	Permitir um ou dois jogadores com representação gráfica 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 Jogador cujo objeto é agregado em Jogo.
3	Disponibilizar ao menos duas fases que podem ser jogadas sequencialmente ou selecionadas, via menu, nas quais jogadores tentam neutralizar inimigos por meio de algum artifício e vice-versa.	Requisito previsto inicialmente e realizado.	Requisito foi realizado completamente porque a classe Fase deriva as classes Quintal e Quarto, que constituem diferentes cenários.
4	Ter pelo menos três tipos distintos de inimigos, cada qual com sua representação gráfica, sendo que ao menos um dos inimigos deve ser capaz de lançar projétil contra o(s) jogador(es) e um dos inimigos deve ser um 'Chefão'.	Requisito previsto inicialmente e realizado.	Requisito foi realizado completamente como se observa na hierarquia da classe Inimigo em que há quatro tipos de inimigos, sendo um deles o Pássaro que atira projéteis e o Bicho Papão de Chefão.
5	Ter a cada fase ao menos dois tipos de inimigos com número aleatório de instâncias, podendo ser várias instâncias e sendo pelo menos 3 instâncias por tipo.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via função rand() no instante de inicializar inimigos em cada Fase.

6	Ter três tipos de obstáculos, cada qual com sua representação gráfica, sendo que ao menos um causa dano em jogador se colidirem.	Requisito previsto inicialmente e realizado.	Requisito foi realizado completamente como se observa na hierarquia da classe Obstáculo há quatro tipos de obstáculos, sendo um deles o Espinho que causa dano ao colidir.
7	Ter em cada fase ao menos dois tipos de obstáculos com número aleatório de instâncias (i.e., objetos), sendo pelo menos 3 instâncias por tipo.	Requisito previsto inicialmente e realizado.	Requisito cumprido inclusive via função rand() no instante de inicializar obstáculos em cada Fase.
8	Ter em cada fase um cenário de jogo constituído por obstáculos, sendo que parte deles seriam plataformas ou similares, sobre as quais pode haver inimigos e podem subir jogadores.	Requisito previsto inicialmente e realizado.	Requisito cumprido ao inicializar plataformas e Cenário em cada Fase.
9	Gerenciar colisões entre jogador para com inimigos e seus projéteis, bem como entre jogador para com obstáculos.	Requisito previsto inicialmente e realizado.	Requisito cumprido via classe Gerenciador de Colisões que invoca método virtual colidir() em cada Personagem polimorficamente.
10	Permitir: (1) salvar nome do usuário, manter/salvar pontuação do jogador (incrementada via neutralização de inimigos) controlado pelo usuário e gerar lista de pontuação (*ranking*). **E** (2) Pausar e **Salvar** Jogada.	Requisito previsto inicialmente e realizado.	Requisito cumprido via classes de Menus e funções de objetos ofstream e ifstream.
Total de requisitos funcionais apropriadamente realizados. *(Cada tópico vale 10%, sendo que para ser contabilizado deve estar realizado efetivamente e não parcialmente)*			**100%** (cem por cento).

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ



- Pacote Roxo: Principal;
- Pacote Vermelho: Menus;
- Pacote Verde: Fases;
- Pacote Azul Claro: Personagens;
- Pacote Laranja: Obstáculos;
- Pacote Rosa: Gerenciadores;
- Pacote Azul Cinzento: Listas;
- Classes de Fora: Texto, Projétil, Cenário.

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Jogo

-gerenciadorGrafico: GerenciadorGrafico
-Fase_Quintal : Quintal
-Fase_Quarto : Quarto
-Jogador1 : Jogador*
-Jogador2 : Jogador*
-Estado : int
-Multiplayer : bool
-menuPrincipal : MenuPrincipal
-menuJogadores : MenuJogadores
-menuFases : MenuFases
-menuPause : MenuPause
-menuColocacao : MenuColocacao
-creditos : Creditos
-Jogador1Fazendeira : bool

+ <<create>> Jogo()
+ <<destroy>> ~Jogo()
+Atualiza(deltaTempo : float): void
+Inicializa(): void
+InicializaFases(): void
+InicializaQuintal() : void
+InicializaQuarto() : void
+InicializaJogadores(): void
+getGerenciadorGrafico() : GerenciadorGrafico&
+getMenuColocacao() : MenuColocacao&
+getJogador1() : Jogador*
+getJogador2() : Jogador*
+setEstadoAtual(estado : int) : void
+MenusJogo(estado : int, tecla : char) : void
+ Executar() : void
+ LoopJogo() : void
+Salvar() : void
+Recuperar() : void
+LimparArquivos() : void
+RecuperarJogadores() : void
+setJogador1Fazendeira(fazendeira : bool)
+getJogador1Fazendeira() : bool

Ente

-pGerenciadorGrafico : GerenciadorGrafico*

+ <<create>> Ente()
+ <<destroy>> ~Ente()
+setGerenciadorGrafico(GerenciadorGrafico : pgerenciadorgrafico) : void

<<Abstrata>>
Entidade

#MovimentoX : float
#MovimentoY : float
#IdCorpo : int
#IdCorpoAtual : static int
#Desalocavel : bool

+ <<create>> Entidade()
+ <<destroy>> ~Entidade() : virtual
+ setDimensoes(x : float, y : float) void
+ getDimensoesX(): float
+getDimensoesY() : float
+ setPosicao(x : float, y : float): void
+ getPosicaoX(): float
+getPosicaoY() : float
+getMovimentoX() : float
+getMovimentoY() : float
+setDesalocavel(desalocavel : bool) : void
+getDesalocavel() : bool
+ setTextura(textura : const string): void
+incrementaIdCorpoAtual() : void
+setId(id : int) : void
+getIdCorpoAtual() : int
+getId() : int
+setSubTextura(subtext : string) : void
+movimenta() : void
+desenhar() : void
+colidir(personagem : Personagem*) : virtual void
+atualiza(deltaTempo : float) : virtual void
salvar() : virtual void

GerenciadorGrafico

-Janela: sf::RenderWindow
-View: sf::View
-Texturas : std::map<std::string, sf::Texture>
-SubTexturas : std::map<std::string, sf::IntRect>
-Fontes : std::map<std::string, sf::Font>
-Cores : std::map<std::string, sf::Color>
-ListaCorpos : std::vector<sf::RectangleShape*>
-carregou : bool

+ <<create>> GerenciadorGrafico()
+ <<destroy>> ~GerenciadorGrafico()
+CarregarJogo(carregou : bool) : void
+getJanela(): sf::RenderWindow&
+getView(): sf::View&
+updateView() : void
+resetaView() : void
+criaCorpo(pentidade : Entidade*, dimx : float, dimy : float, posx : float, posy : float, textura : string) : void
+setDimensoes(id : int, x : float, y : float) : void
+getDimensoesX(id : int) : float
+getDimensoesY(id : int) : float
+setPosicao(posx : float, posy : float) : void
+getPosicaoX(id : int) : float
+getPosicaoY(id : int) : float
+setSubTextura(id : int, subtext : string) : void
+desenhar(id: int, desalocavel : bool) : void
+desenhar(texto : sf::Text) : void
+fechar() : void
+movimenta(id : int, x :float, y: float) : void
+LoopJogo(jogo : Jogo*, estado : int) : void
+TeclaApertada(direita : char*, esquerda : char*, pulo : char*, atira : char*) : void
+CarregaTextura(textura : string) : void
+InicializaTexturas() : void
+InicializaFontes() : void
+InicializaCores() : void
+InicializaSubTexturas() : void

Menu

#Menu: sf::Text*
#Indice: int
#Tamanho : int
#jogo: Jogo*
#Textos : Texto*
#jogo : Jogo*
#PlanoFundo : Cenario

+ <<create>> Menu(tamanho : int, jg : Jogo*)
+ <<destroy>> ~Menu()
+moverCima(): void
+moverBaixo():void
+desenhar(): void

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ListaEntidades

-listaEntidades : Lista <Entidade>

+ <<create>> ListaEntidades()
+ <<destroy>> ~ListaEntidades()

+atualiza() : void
+desenhar() : void
+salvar() : void
+inclua(pentidade : Entidade*) : void
+getAtual() : Entidade*
+tamanho() : int
+proximo() : void
+anterior() : void
+inicio() : void
+limpar() : void
+limparTudo() : void
+remove(pentidade : Entidade*) : void
operator[](indice : int) : Entidade*

ListaPersonagens

-listaPersonagens : Lista <Personagem>

+ <<create>> ListaEntidades()
+ <<destroy>> ~ListaEntidades()

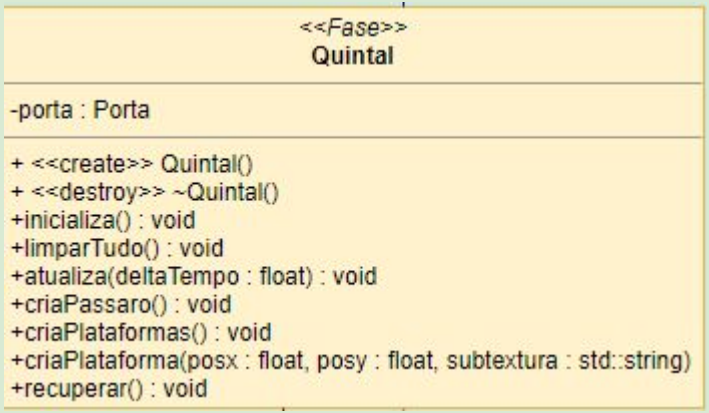
+atualiza() : void
+desenhar() : void
+salvar() : void
+inclua(ppersonagem : Personagem*) : void
+getAtual() : Personagem*
+tamanho() : int
+proximo() : void
+anterior() : void
+inicio() : void
+limpar() : void
+pop(ppersonagem : Personagem*) : void
+operator[](indice : int) : Personagem*

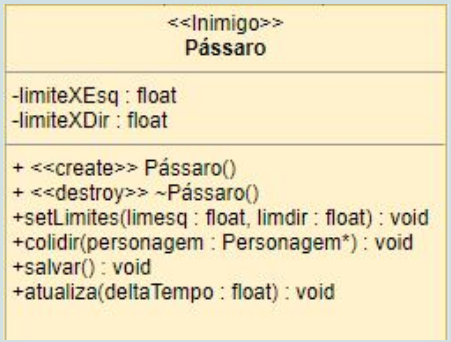
GerenciadorColisoes

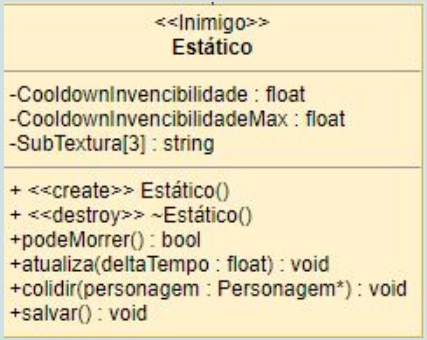
-listaEntidades: ListaEntidades*
-listaPersonagens : ListaPersonagens*

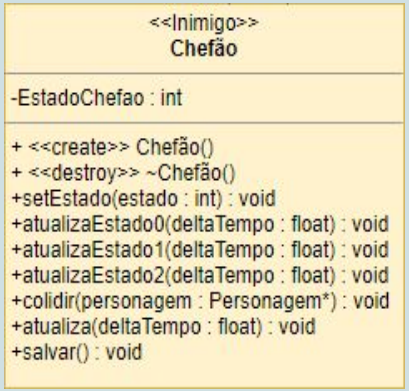
+ <<create>> GerenciadorColisoes()
+ <<destroy>> ~GerenciadorColisoes()
+ setListaEntidades(listaentidades : ListaEntidades*) : void
+ setListaPersonagens(listapersonagens : ListaPersonagens*) : void
+ checaColisoes(): void
+ checaColisao(personagem : Personagem*, entidade : Entidade*) : void

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

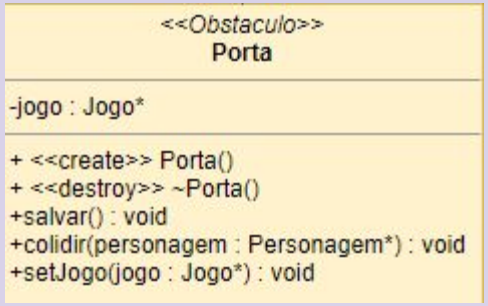


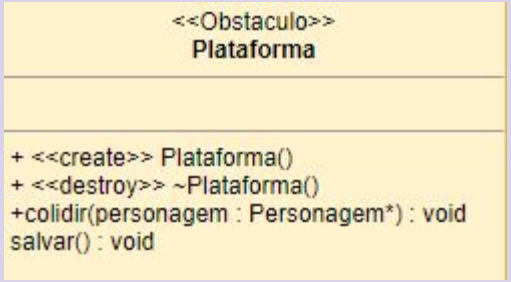


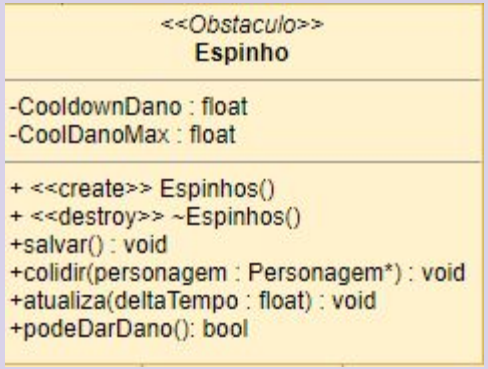


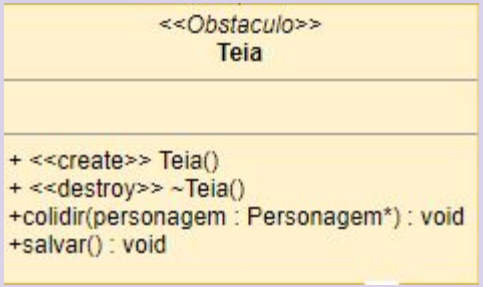


UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ









UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

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	No desenvolvimento como um todo
	- Classe Principal.	Sim	Main.cpp & Jogo.h/.cpp
	- Divisão em .h e .cpp.	Sim	No desenvolvimento como um todo.
2	**Relações de:**		
	- Associação direcional. & - Associação bidirecional.	Sim	No desenvolvimento como um todo.
	- Agregação via associação. & - Agregação propriamente dita.	Sim	Classes Quarto e Quintal agregam objetos da classe Espinho
	- Herança elementar. & - Herança em diversos níveis.	Sim	Classe Entidade elementar, que chega a três níveis de herança.
	- Herança múltipla.	Sim	Classe Atirador
3	**Ponteiros, generalizações e exceções**		
	- Operador *this* para fins de relacionamento bidirecional.	Sim	Jogo.h/.cpp com Menu.h/.cpp
	- Alocação de memória (*new* & *delete*).	Sim	Inclusão e exclusão de entidades via classe ListaEntidades
	- Gabaritos/*Templates* criada/adaptados pelos autores (e.g. Listas Encadeadas via *Templates*).	Sim	Template de de Lista Encadeada adaptada da prova do professor[1]
	- Uso de Tratamento de Exceções (*try catch*).	Sim	Cálculo do versor da direção do projétil de pássaro para evitar divisão por 0.
4	**Sobrecarga de:**		
	- Construtoras e Métodos.	Meio	Método desenhar da classe GerenciadorGrafico
	- Operadores (2 tipos de operadores pelo menos).	Sim	Classe gabarito Lista
	Persistência de Objetos (via arquivo de texto ou binário)		
	- Persistência de Objetos.	Sim	Salvamento polimórfico de cada entidade.
	- Persistência de Relacionamento de Objetos.	Sim	Salvamento polimórfico de cada entidade.
5	**Virtualidade:**		
	- Métodos Virtuais.	Sim	colidir(); movimentar(); ...
	- Polimorfismo	Sim	Colisão e movimento de Personagem; ...
	- Métodos Virtuais Puros / Classes Abstratas	Sim	Classe Entidade; ...
	- Coesão e Desacoplamento	Sim	No desenvolvimento como um todo.
6	**Organizadores e Estáticos**		
	- Espaço de Nomes (*Namespace*) criada pelos autores.	Sim	Organizando menus do jogo.
	- Classes aninhadas (*Nested*) criada pelos autores.	Sim	Classe ListaEntidades, adaptada de uma prova anterior do professor[1]
	- Atributos estáticos e métodos estáticos.	Sim	Id de entidade atual. Pontuação simultânea de jogadores.
	- Uso extensivo de constante (*const*) parâmetro, retorno, método...	Sim	No desenvolvimento como um todo

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

7	**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)	Sim	Classe Texto utilizando strings. Vector da classe Texto em menus do jogo..
	- Pilha, Fila, Bifila, Fila de Prioridade, Conjunto, Multi-Conjunto, Mapa **OU** Multi-Mapa.	Sim	Mapa para associar strings a cores, texturas e fontes do SFML dentro do Gerenciador Gráfico.
	Programação concorrente		
	- *Threads* (Linhas de Execução) no âmbito da Orientação a Objetos, utilizando Posix, C-Run-Time **OU** Win32API ou afins.	Não	
	- *Threads* (Linhas de Execução) no âmbito da Orientação a Objetos com uso de Mutex, Semáforos, **OU** Troca de mensagens.	Não	
	Biblioteca Gráfica / Visual		
	- Funcionalidades Elementares. & - Funcionalidades Avançadas como: ● tratamento de colisões ● duplo *buffer*	Sim	Classe GerenciadorGrafico trata janela do jogo. Classe GerenciadorColisoes trata as colisões. Classe GerenciadorColisoes trata as colisões.
	- Programação orientada e evento em algum ambiente gráfico. **OU** - *RAD – Rapid Application Development* (Objetos gráficos como formulários, botões etc).	Sim	Classe GerenciadorGrafico trata os eventos.
	Interdisciplinaridades via utilização de Conceitos de Matemática Contínua e/ou Física.		
	- Ensino Médio.	Sim	Implementação de gravidade da Cinemática, para reproduzir movimentos análogos à realidade para o Personagens.
	- Ensino Superior.	Sim	Módulo e versor de vetores da Geometria Analítica para direcionar projéteis à posição do Jogador.

9	**Engenharia de Software**		
	- Compreensão, melhoria e rastreabilidade de cumprimento de requisitos. &	Sim	No desenvolvimento como um todo.
	- Diagrama de Classes em *UML*.	Sim	Plataforma Draw.io
	- Uso efetivo e intensivo de padrões de projeto *GOF*.	Não	
	- Testes à luz da Tabela de Requisitos e do Diagrama de Classes.	Sim	No desenvolvimento como um todo.
10	**Execução de Projeto**		
	- Controle de versão de modelos e códigos automatizado (via SVN e/ou afins). & - Uso de alguma forma de cópia de segurança (backup).	Sim	Git e GitHub do repositório do D.A.P.C[2] e do F.C.B[3]
	- Reuniões com o professor para acompanhamento do andamento do projeto.	Sim	Reuniões dias 29/07, 03/08; 06/08 e 12/08. Total de reuniões: 4.
	- Reuniões com monitor da disciplina para acompanhamento do andamento do projeto.		Reuniões com A. M. C: dias 15/07, 02/08, 05/08, 06/08, 09/08, 10/08 e 13/08; Reunião com L. B. V: dia 13/07. Reuniões com M. K. L.: dias 04/08 e 11/08; Total de reuniões: 10
	- Revisão do trabalho escrito de outra equipe e vice-versa.		Pedro Foresti Leão e Carolina de Souza Fernandes
	Total de conceitos apropriadamente utilizados. (Cada grande tópico vale 10% do total de conceitos. Assim, por exemplo, caso se tenha feito metade de um tópico, então valeria 5%.)		**100%** (setenta por cento).

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Conclusão geral dos resultados alcançados:

- Aprimoramento de novas habilidades e competências;
- Evolução na capacidade de trabalho em equipe;
- Aprendizado sobre o ciclo da engenharia de software;
- Compreensão sobre as diferenças entre programação procedimental e orientada a objetos;
- Introdução ao processo de desenvolvimento profissional.