

### **TeSP - Desenvolvimento de Jogos Digitais**

Técnico Superior Profissional

Plano: Aviso de Registo nº R/Cr 55/2017 de 13-07-2017

#### **Ficha da Unidade Curricular: Programação Orientada por Objetos**

ECTS: 5; Horas - Totais: 135.0, Contacto e Tipologia, TP:15.0; PL:45.0;

Ano | Semestre: 1 | S2; Ramo: Tronco comum;

Tipo: Obrigatória; Interação: Presencial; Código: 639013

Área de educação e formação: Ciências informáticas

#### **Docente Responsável**

Fernando Sérgio Hortas Rodrigues

Professor Adjunto

#### **Docente e horas de contacto**

Fernando Sérgio Hortas Rodrigues

Professor Adjunto, TP: 15; PL: 45;

### **Objetivos de Aprendizagem**

Adquirir conhecimentos sobre o paradigma da orientação a objetos.

### **Objetivos de Aprendizagem (detalhado)**

1. Adquirir conhecimentos sobre:
  - 1.1. - Paradigma da orientação a objetos, tais como hereditariedade, abstração, encapsulamento e polimorfismo.
  - 1.2. - Linguagem de programação C++.
2. Aprender a linguagem e o paradigma de orientação a objetos através da sua aplicação prática ao desenvolvimento de pequenos jogos.

### **Conteúdos Programáticos**

C++: Tipos; Expressões; Apontadores; Referências; Classes; Funções; Funções Membro; Sobreposição de Funções; Construtores; Destrutores; Hereditariedade; Polimorfismo; Classes Abstratas.  
SFML

### **Conteúdos Programáticos (detalhado)**

1. Conceitos gerais
  - 1.1. Paradigma das linguagens compiladas: Pré-processamento, compilação e linkagem.
  - 1.2. Breve introdução à linguagem C++
  - 1.3. O ambiente de desenvolvimento MS Visual Studio
  - 1.4. Soluções, Projetos e Templates
2. Fundamentos da linguagem C++
  - 2.1. Perceber os fundamentos da linguagem C++
  - 2.2. Os vários tipos predefinidos do C++
  - 2.3. Declaração e inicialização de variáveis e constantes



- 2.4. Enumerados
- 2.5. Conversões implícitas e explícitas entre tipos
- 2.6. Expressões
- 2.7. Instruções Condicionais
  - 2.7.1. *if..else*
  - 2.7.2. Operador ternário *?:*
  - 2.7.3. *switch*
- 2.8. Instruções Iterativas
  - 2.8.1. *for*
  - 2.8.2. *while*
  - 2.8.3. *do..while*
  - 2.8.4. Instruções *break* e *continue*
- 2.9. Arrays
  - 2.9.1. *Declaração e inicialização de arrays unidimensionais e multidimensionais*
  - 2.9.2. *O template array<T, N>*
  - 2.9.3. *O template vector<T>*
- 2.10. Ciclo *for* para coleções ou *for* melhorado (*enhanced for*)
  
- 3. Apontadores e Referências (*Pointer* e *References*)
  - 3.1. Apontadores para *char*
  - 3.2. Apontadores e arrays
  - 3.3. Aritmética de apontadores
  - 3.4. Alocação dinâmica de memória
    - 3.4.1. Operadores *new* e *delete*
    - 3.4.2. Alocação dinâmica de arrays
  - 3.5. Apontadores primitivos e Apontadores Inteligentes (*Raw Pointers* e *Smart Pointers*)
    - 3.5.1. Apontadores inteligentes únicos
    - 3.5.2. Apontadores inteligentes partilhados
  - 3.6. Referências
    - 3.6.1. Definição de referências
    - 3.6.2. Utilização de uma referência num ciclo *for* melhorado
  
- 4. Utilização de Strings
  - 4.1. Melhor classe de strings
  - 4.2. Conversão de strings para numéricos
  - 4.3. *Streams* de strings
  
- 5. Funções
  - 5.1. Definição de funções
  - 5.2. Declaração de funções (protótipos de funções)
  - 5.3. Passagem de argumentos para funções
    - 5.3.1. Passagem por valor
      - 5.3.1.1. Passagem de apontadores
      - 5.3.1.2. Passagem de arrays
  - 5.4. Passagem por referência
    - 5.4.1. Referência vs. Apontadores
    - 5.4.2. Parâmetros de entrada (input) vs. parâmetros de saída (output)
    - 5.4.3. Passagem de arrays por referência
  - 5.5. String Views e parâmetros String Views
  - 5.6. Valores por omissão para parâmetros
  - 5.7. Valores de retorno
    - 5.7.1. Retorno de um apontador
    - 5.7.2. Retornar uma referência

- 5.7.3. Parâmetro de retorno vs. Parâmetros de saída
- 5.8. Variáveis estáticas
- 5.9. *Overload* de funções
  - 5.9.1. *Overload* e parâmetros apontador
  - 5.9.2. *Overload* e parâmetros referência
  - 5.9.3. *Overload* e parâmetros constantes
- 5.10. Fundamentos de *Templates* de funções
  
- 6. *Namespaces*
  - 6.1. O namespace *Global*
  - 6.2. Definir um *namespace*
  - 6.3. Utilização de diretivas *using*
  - 6.4. Funções e *namespaces*
  - 6.5. *Namespaces* anónimos
  - 6.6. *Namespaces* aninhados
  - 6.7. Pseudónimos (*Aliases*), de *namespaces*
  
- 7. Classes e Objetos
  - 7.1. Classes e orientação a objetos
  - 7.2. Encapsulamento
  - 7.3. Classes e Objetos
  - 7.4. Modificadores de acesso: *public*, *private* e *protected*
  - 7.5. Construtores
    - 7.5.1. Construtor por omissão (explícito e implícito)
      - 7.5.1.1. A opção *default*
    - 7.5.2. Construtores com parâmetros
    - 7.5.3. Definição de construtores e funções fora da classe
    - 7.5.4. Inicializadores de membros (*Member Initializer List*)
    - 7.5.5. A opção *explicit*
    - 7.5.6. Construtores delegados ou inicialização de construtores
  - 7.6. O construtor cópia (*Copy Constructor*)
  - 7.7. Funções *Setters* e *Getters* para acesso a membros de classe privados
  - 7.8. Objetos e Funções membro constantes
  - 7.9. *Overloading* sobre constantes
  - 7.10. A opção *mutable*
  - 7.11. Membros estáticos (*static*)
  - 7.12. Destrutores
  - 7.13. Membros de classe do tipo apontador
  
- 8. Herança (*Inheritance*)
  - 8.1. Herança vs. Agregação
  - 8.2. Classes derivadas
  - 8.3. Acesso a construtores da classe base nas classes derivadas
  - 8.4. Herança múltipla
  - 8.5. Classes base Virtuais
  
- 9. Polimorfismo
  - 9.1. Apontador de classe base (*Base Class Pointer*)
  - 9.2. Chamar funções herdadas
  - 9.3. Funções virtuais
  - 9.4. As opções *override* e *final*
  - 9.5. Funções virtuais e Hierarquia de classes
  - 9.6. Funções virtuais e Modificadores de acesso

- 9.7. Coleções polimórficas
- 9.8. Funções virtuais puras
  - 9.8.1. Classes abstratas
  - 9.8.2. Classes abstratas como interfaces

### **Metodologias de avaliação**

Ép. Normal de Frequência:

- Teste escrito (30%)
- Todos os alunos são automaticamente admitidos a exame.

Ép. Normal de Exame:

- Nota da frequência ou exame escrito (30%)
- Trabalho Prático (70%)

Ép. de Recurso:

- Exame escrito (30%)
- Trabalho Prático (70%)

Todas as épocas:

- Nota mínima testes e exame: Sete (7) valores
- Nota mínima trabalho prático: Dez (10) valores

### **Software utilizado em aula**

MS Visual Studio

Framework SFML - Simple and Fast Multimedia Library

### **Bibliografia recomendada**

- Ivor Horton and P. Van Weert, *Beginning C++17: From Novice to Professional, Fifth ed.* Apress, 2018.  
ISBN: 978-1-4842-3365-8
- John Horton, *Beginning C++ Game Programming.* Birmingham-Mumbai: Packt Publishing, 2016.  
ISBN: 978-1-7864-6619-8

### **Coerência dos conteúdos programáticos com os objetivos**

Aprendizagem inicial da linguagem de programação orientada a objeto C++, que cumpre dois objetivos: Em primeiro lugar a aprendizagem dos conceitos de orientação a objetos, presentes nas modernas linguagens de programação e em segundo lugar a aprendizagem e a prática de uma linguagem de programação, largamente utilizada no desenvolvimento de jogos, de forma direta ou através de motores de jogos.

O Objetivo 1 é endereçado por todos os capítulos do programa do 1 ao 9.

O objetivo 2 é implementado através da aplicação prática dos conceitos aprendidos, ao desenvolvimento jogos simples, utilizando a linguagem juntamente com uma biblioteca externa (SFML).

### **Metodologias de ensino**

Aulas teórico-práticas expositivas onde se descrevem os conceitos fundamentais e são utilizados pequenos exemplos demonstrativos.

Aulas práticas de desenvolvimento de jogos simples, de forma apoiada, para utilização prática dos conceitos adquiridos nas aulas teórico-práticas.

**Coerência das metodologias de ensino com os objetivos**

Os conceitos teóricos são lecionados com recurso a exemplos simples e ilustrativos. Na componente prática os alunos efetuam jogos simples de forma assistida, para uma consolidação profunda e persistente dos conceitos transmitidos.

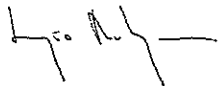
**Língua de ensino**

Português e Inglês

**Pré-requisitos**

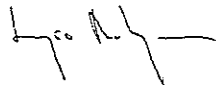
Conceitos básicos de algoritmos

**Docente Responsável**



Sérgio Hortas Rodrigues  
cn=Sérgio Hortas Rodrigues, o=IPT, ou=ESTA,  
email=sergio.rodrigues@ipt.pt, c=PT  
2019.02.15 16:41:29 Z

**Diretor de Curso, Comissão de Curso**



Sérgio Hortas Rodrigues  
cn=Sérgio Hortas Rodrigues, o=IPT, ou=ESTA,  
email=sergio.rodrigues@ipt.pt, c=PT  
2019.02.15 16:41:54 Z

**Conselho Técnico-Científico**

