Universidade Federal de Santa Catarina
Centro Tecnológico - CTC
Departamento de Engenharia Elétrica e Eletrônica - EEL

DEPARTAMENTO: *Engenharia Elétrica*
CURSO: *Engenharia Eletrônica*
DISCIPLINA: Programação C++ para Sistemas Embarcados
CÓDIGO: EEL 7323
CRÉDITOS: *04*
CARGA HORÁRIA: *72 horas-aula*
REQUISITOS:
Pré-requisito: *INE5201 - Introdução à Ciência da Computação*
OFERTA: *Engenharia Eletrônica*
Engenharia Elétrica
Engenharia de Controle e Automação
VALIDADE: *a partir de 2019/2*
AUTOR: Prof. Eduardo Augusto Bezerra <Eduardo.Bezerra@ufsc.br>

OBJETIVOS: O cumprimento da disciplina busca dar ao aluno, ao final do semestre, condições de:

1. Dar continuidade aos estudos de programação de sistemas computacionais embarcados.
2. Compreender os conceitos fundamentais do paradigma de programação orientada a objetos.
3. Desenvolver a capacidade de análise de programas em C++ de complexidade baixa ou média.
4. Entender o funcionamento básico de sistemas operacionais para acesso a periféricos.
5. Desenvolver programas em C++ para sistemas embarcados baseados em microprocessadores e microcontroladores, e também sistemas do tipo System-on-a-chip (SoC).

EMENTA:

Conceitos introdutórios; Sistema operacional; Orientação a objetos; Classes; Estruturas de dados; Encapsulamento; Herança; Polimorfismo; Tratamento de exceções; Templates; Entrada e saída; System-on-a-chip; Teste e verificação.

CONTEÚDO PROGRAMÁTICO:

UNIDADE: 01
CONTEÚDO: Introdução

1.1. Paradigma de programação orientada a objetos
1.2. UML e diagrama de classes
1.3. Linguagem C e Linguagem C++
1.4. Ferramentas de desenvolvimento
1.5. Comandos básicos

Nº DA UNIDADE: 02
CONTEÚDO: Conceitos Básicos

2.1. Declarações e definições
2.2. Ponteiros e alocação de memória
2.3. Conversões padrão
2.4. Expressões
2.5. Instruções e declarações

UNIDADE: 03
CONTEÚDO: Classes

3.1. Abstração
3.2. Classes e instâncias
- **3.2.1.** Atributos
- **3.2.2.** Métodos

3.3. Encapsulamento
- **3.3.1.** Declaração e visibilidade de atributos
- **3.3.2.** Declaração e visibilidade de métodos

3.4. Relacionamento entre classes
3.5. Construtores
3.6. Sobrecarga
3.7. Atributos e métodos de classe
3.8. Auto-referência
3.9. Modularização (agrupamento de classes relacionadas)

Nº DA UNIDADE: 04
CONTEÚDO: Herança e interface

4.1. Hierarquia de classes
4.2. Classes abstratas
4.3. Polimorfismo
4.4. Conversão dinâmica de tipos

Nº DA UNIDADE: 05
CONTEÚDO: Tratamento de exceções

5.1. Geração de exceções
5.2. Criação de exceções
5.3. Asserções

Nº DA UNIDADE: 06
CONTEÚDO: Sistema básico de E/S

6.1. Fluxo de E/S
6.2. Manipulação de arquivos

Nº DA UNIDADE: 07
CONTEÚDO: Templates

7.1. Funções template
7.2. Classes template
7.3. Derivando classes template
7.4. STL - Standard Template Library

Nº DA UNIDADE: 08
CONTEÚDO: Programação de sistemas embarcados

8.1. Ferramentas e ambientes de desenvolvimento
8.2. Plataformas de prototipação microprocessadas
8.3. Modelos de programação
8.3. Programação de interfaces e periféricos

METODOLOGIA:

As aulas são realizadas em laboratório de ensino com disponibilidade de 15 computadores conectados a Internet. Durante as aulas são apresentados conceitos e técnicas de programação de sistemas embarcados em C++, com o auxílio de quadro branco e projetor multimídia. O slides e recursos de ensino são disponibilizados no site do professor em http://gse.ufsc.br/bezerra. Durante as aulas, são definidos exercícios práticos de projeto e implementação, e os alunos trabalham em duplas de forma a resolver os problemas teóricos e práticos, contando com o auxílio do professor.

AVALIAÇÃO:

Serão desenvolvidos trabalhos durante o semestre, que serão utilizados como base para a implementação do trabalho final. A média geral do semestre (MS) é obtida a partir da média geométrica MS = (A + T), onde A é a nota da avaliação, e T é a nota do trabalho final.

Condições para a aprovação: freqüência > 75% e MS >= 6,0

BIBLIOGRAFIA:

- **BÁSICA:**

1. Walter J. Savitch, "C++ absoluto", Addison Wesley, 2004 ISBN 85-88639-09-2. Disponível na biblioteca da UFSC em "Livros Eletrônicos". Título original: Absolute C++.

- **COMPLEMENTAR:**

1. H. M. Deitel e P. J. Deitel, C++: como programar, São Paulo: Pearson Prentice Hall, 2006. ISBN 85-7605-056-0 - Disponível na biblioteca da UFSC em "Livros Eletrônicos". Título original: C++ How to Program.

2. Victorine V. Mizrahi, Treinamento em Linguagem C++: módulo 1, Makron Books, 2007. ISBN 9788534602907 - Disponível na biblioteca da UFSC em "Livros Eletrônicos".

3. Victorine V. Mizrahi, Treinamento em Linguagem C++: módulo 2, Makron Books, 2007. ISBN 9788534603034 - Disponível na biblioteca da UFSC em "Livros Eletrônicos".

4. Herb Sutter, Programação avançada em C++, Pearson Makron Books, 2006. ISBN 85-346-1545-4 - Disponível na biblioteca da UFSC em "Livros Eletrônicos". Título original: Exceptional C++ Style.

5. Richard C. Lee; William M. Tepfenhart, UML e C++ - Guia Prático de Desenvolvimento Orientado a Objeto, Makron Books, ISBN 85-346-1364-8 - Disponível na biblioteca da UFSC em "Livros Eletrônicos".. Título original: UML and C++ - A Practical Guide to Object-Oriented Development.

6. B. Stroustrup, The C++ Programming Language. Reading: Addison-Wesley, 1997.

7. W. Wolf, Computer as Components - Principles of embedded computing system design, Morgan Kaufmann Publishers, 2000.

8. L. Lavagno, UML for real design of embedded real-time systems, Kluwer Academic, 2003.

9. Christopher Hallinan, Embedded Linux Primer, Prentice Hall, 2007.

DEPARTAMENTO: Engenharia Elétrica e Eletrônica
CURSO: Engenharia Eletrônica
DISCIPLINA: Programação C++ para Sistemas Embarcados
CÓDIGO: EEL 7323
CRÉDITOS: 04
CARGA HORÁRIA: 72 horas-aula
REQUISITOS:
Pré-requisito: INE5201 - Introdução à Ciência da Computação
OFERTA: Engenharia Eletrônica
Engenharia Elétrica
Engenharia de Controle e Automação
VALIDADE: a partir de 2019/2
AUTOR: Prof. Eduardo Augusto Bezerra <Eduardo.Bezerra@ufsc.br>

Esse cronograma previsto para as aulas está disponível em: http://gse.ufsc.br/bezerra/?page_id=2308

Conteúdo das aulas

- **Apresentação da disciplina.** Slides aula 1.
- Artigo: "Embedded Software", Edward A. Lee, Advances in Computers, Vol. 56, Academic Press, London, 2002. [cópia local]
- Artigo: "Incorporating C++ into Mars Rover Flight Software" [cópia local] (slides: 13; 16-19; 34-40; 52)
- Slides "C em sistemas embarcados".
- **Ambiente de desenvolvimento I [20]**: Tutorial [6]; One page Linux; Comandos básicos; Exercícios Linux.
- **Ambiente de desenvolvimento II**: Ferramentas para desenvolvimento na

web (github, terminal linux, editor de código fonte, compilador C/C++, ...): http://www.compileonline.com/

- **Revisão de C, Ponteiros [3]**: Notas de aula; Exemplos; Exercícios; Soluções.

- **Revisão de C, introdução a C++:** Estrutura programa (Hello World!) [7]; Variáveis [8]; Operadores [9]; Controle de fluxo [10]; Funções [11].
- **Introdução a C++:** Class x struct [12]; Exercícios (fazer o exercício 6 da lista na aula) [13].
- **Paradigma de orientação a objetos**: Objetos [14]; Classes [15]; Abstração [16]; Encapsulamento [17]; Métodos, atributos, construtores [18]; Classe Resistor. **Exercícios** [19].
- **Relacionamento entre classes**: Agregação; Exercícios.
- **Exercício para fixação do conteúdo a ser desenvolvido pela turma durante a aula**: Cadastro de alunos (Exercício 6 da lista [13]).
- Correção do exercício em aula (Ver na pasta "Agregacao" uma possível solução para o exercício).

- **Hierarquia de classes**: Herança; Exemplo; Herança múltipla; Construtores e destrutores; Notas de aula [1].
- **Exercício para fixação do conteúdo a ser desenvolvido pela turma durante a aula**: Implementação de calendário com data e hora, usando como base os exemplos de herança múltipla discutidos na aula. O programa deverá ficar em laço infinito, atualizando a data e hora.

- **Plataforma Atlys, Leon3 + FPGA (System-on-a-Chip – SoC)**
- Tutorial Leon3 na plataforma Atlys – apresentação do fluxo de

desenvolvimento utilizando um cross-compiler, e ferramenta de download.

- Utilização do display OLED na plataforma Atlys [material OLED na Atlys] e [projeto 2015/1].
- **Exercício**: Protipação do clock/calendar descrito nas Notas de aula [1] na plataforma Atlys com Leon3 – solução para a implementação do clock/calendar, considerando os recursos reduzidos de um kit de desenvolvimento baseado em um microprocessador, sem sistema operacional (bare-metal). Dicas para acesso aos recursos de I/O (chaves/botões) estão disponíveis em: grlib_README
- **Plataforma *32F429 Discovery*** *(disponível no LABSDG). Para utilizar Linux nessa plataforma, os seguintes tutoriais podem ser seguidos:*
  - eLinux.org: https://elinux.org/STM32
  - Ubuntu: http://vedder.se/2012/07/get-started-with-stm32f4-on-ubuntu-linux/
- Outra placa que pode ser utilizada (adquirida pelos alunos que desejarem): ARM Cortex M4F

- **Exercício**: herança.

- **Classes bases virtuais; Funções virtuais; *Friends*; Sobrecarga de operadores**: Notas de aula [1]; Exemplo friends; Exercícios.
- **Exercício friends/sobrecarga operadores**: Implementação do “calendário com data/hora” (herança múltipla) na Atlys, utilizando o conceito de sobrecarga de operadores e friends [ver exercício 2].
- **Classes abstratas; Polimorfismo**: Notas de aula [1].

- **Exercício sobre polimorfismo**: implementar as classes necessárias para que o programa testCShape3d funcione corretamente.

- **Avaliação (data a ser definida)**

- **Funções Template e Classes Template**: **Notas de aula** e **exemplos**.
- **Exercício**: Adaptar o programa da CShape3d para utilização de templates, possibilitando o cálculo de volumes com float e inteiro.
- **C++ Standard Template Library – STL**: tutorial e exemplos.
- **Tratamento de exceções**: exemplos e std::exception.
- **Manipulação de arquivos**: exemplos.
- **Sistema básico de I/O**: exemplos.
- **Exercício**: projeto e desenvolvimento de software para controle de uma máquina de venda de refrigerantes (vending machine). Será exercitado o ciclo completo de projeto, incluindo o levantamento de requisitos, descrição do projeto com máquina de estados finita (FSM), codificação, implementação e testes.
    - Especificação completa, e sugestão de solução
    - Dicas:
        - Tutorial para uso da Atlys/Leon3;
        - Instruções para uso dos LEDs e botões da Atlys, com o Leon3 (ver linhas 56 a 58);
        - Template para uso do OLED;
        - Template para uso dos LEDs da Atlys;
        - Template para data/hora.

- **Estruturas de dados em C++**: Vetor, Matriz, Listas, Filas, Pilhas, Grafo, Arvore.
  - Exemplo de lista encadeada: [ListaInteiros].
  - Exemplo de árvore: [Arvore].
- Demonstração durante a aula de detalhes de operações em estruturas de dados: inclusão, remoção, consulta, listagem, construtor, destrutor.
- Uso de templates nas estruturas de dados (ex. nodo em lista encadeada)
- ~~**Exercício**: Implementar uma lista para armazenamento de valores inteiros ordenada, ou seja, uma lista que a cada inclusão deverá permanecer com os valores armazenados nos nodos em ordem crescente. A lista não pode aceitar valores duplicados.~~
- ~~**Exercício**: Implementar uma lista duplamente encadeada, com as mesmas funcionalidades apresentadas na lista simplesmente encadeada, utilizando as facilidades dos nodos com ponteiros para o próximo nodo e para o nodo anterior e, também, utilizando um ponteiro fixo no último nodo da lista.~~
- **Exercício**: Alterar o exercício 6 desenvolvido no início do semestre [13] (cadastro de aluno e notas), visando utilizar uma lista duplamente encadeada como estrutura de dados, no lugar do vetor utilizado originalmente. Os nodos da lista, além dos dados do aluno (matrícula e notas), deverá conter também a data e hora atual (usar o clock/calendar desenvolvido anteriormente). O valor da data/hora pode ser salva como uma string. Após testar o programa em um computador, transferir para o kit de desenvolvimento e realizar os devidos testes de funcionamento.

- Questionário respondido por engenheiros e gerentes da área tecnológica: “Greatest technology challenges”

    - Respostas fornecidas por engenheiros e gerentes de projeto
- Palestra do Bjarne Stroustrup, criador do C++: "Writing good C++14". Essa palestra e' relacionada a uma das frases famosas do Stroustrup: "*C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do it blows your whole leg off*". O link fornecido inicia no final do slide 21, onde e' apresentado um problema discutido nas aulas da disciplina. Que problema e' esse? Quais as implicações desse problema? Quais as solucoes apresentadas na palestra?
- Webinar: "Migrating from Embedded C to C++". Tópicos cobertos:
    - Encapsulation by classes and namespaces
    - Automatic initialization with class constructors
    - Function overloading
    - Improved reuse with class inheritance and virtual functions
    - Safe flexibility with class templates
    - Stronger checking by compiler
    - Standard library of containers and algorithms
    - Integration with existing C code
- System-on-a-Chip (SoC): Plataforma DE1-SoC Cyclone V da Altera, ARM (dual core Cortex A9) + FPGA
- System-on-a-Chip (SoC): Plataforma Zynq da Xilinx, ARM (dual core Cortex A9) + FPGA

- Aulas reservadas para o desenvolvimento do projeto final (atividade extra-classe).

- Entrega/apresentação do projeto final.