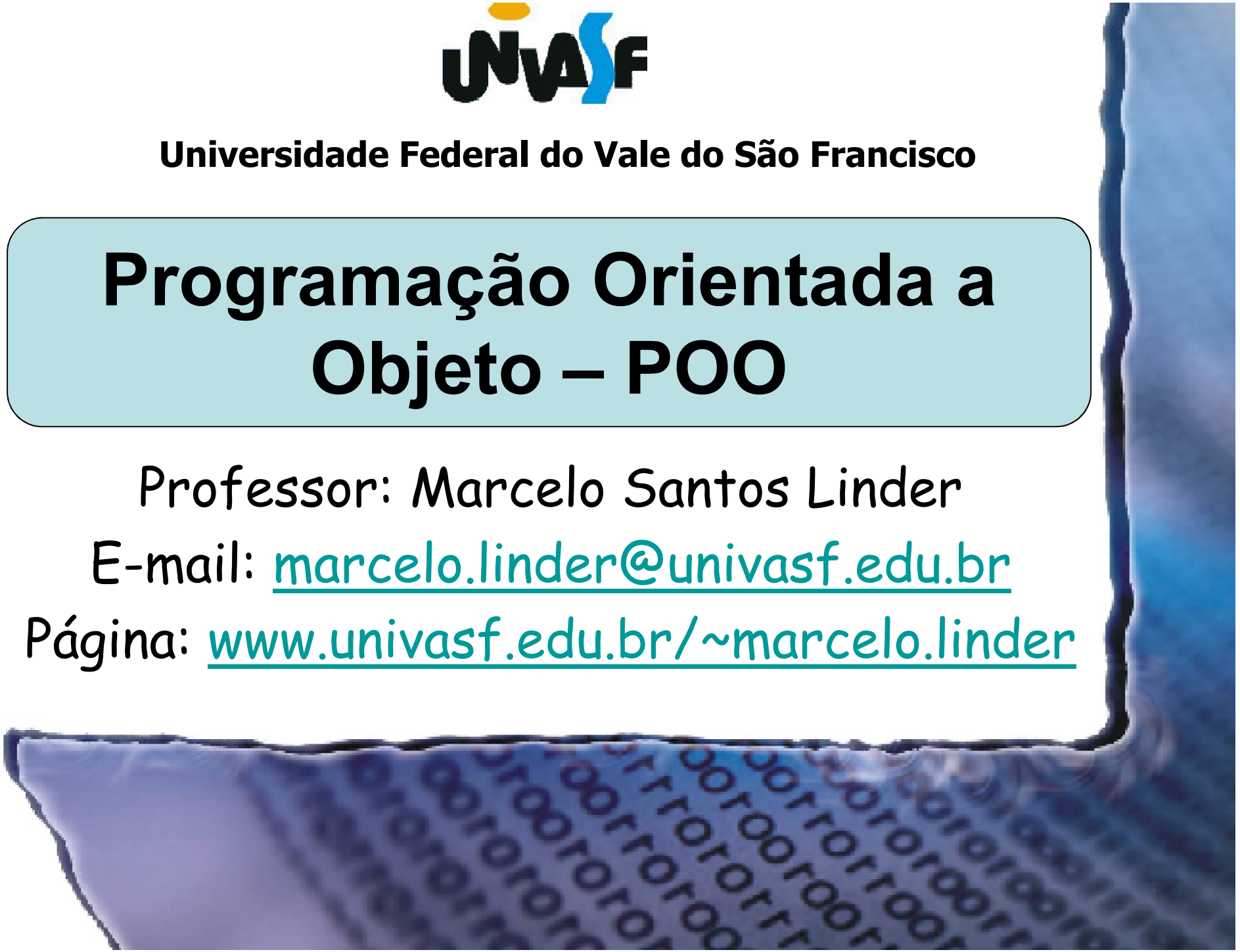


Universidade Federal do Vale do São Francisco

Programação Orientada a Objeto – POO

Professor: Marcelo Santos Linder

E-mail: marcelo.linder@univasf.edu.br

Página: www.univasf.edu.br/~marcelo.linder

PD e PUD

- **Programa da Disciplina (PD)**
  - Ementa
  - Objetivos
  - Metodologia
  - Recursos
  - Forma de Avaliação
  - Conteúdo Programático
  - Bibliografia

- **Plano de Unidade Didática (PUD)**
  - Datas (aulas, avaliações)
  - Tópicos das aulas

Programa da Disciplina

- Ementa
- Objetivos
- Metodologia
  - Recursos
- Forma de Avaliação
- Conteúdo Didáticos
  - Datas (aulas, avaliações)
  - Tópicos das aulas
- Referências Bibliográficas

Forma de Avaliação

- A avaliação será realizada mediante duas provas e um trabalho. A média do discente resultará da média aritmética das notas obtidas.

Observação: O aluno para obter aprovação deve ter no mínimo 75% de presença.

Bibliografia

Bibliografia Básica:

- [1] KEOGH, J.; GIANNINI, M. **OOP Desmistificado – Programação a Objetos.** Alta Books, 2005.
- [2] BARNES, K. **Programação orientada a objetos com Java: Uma introdução Prática Usando o BlueJ**. 4ª ed. Pearson Education, 2004.
- [3] DEITEL, M.D.; DEITEL, P.J. **C++ como programar**. 3ª ed. Bookman, 2001.

Bibliografia Complementar:

- [4] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML Guia do Usuário**. 2ª ed. Elsevier, 2005.
- [5] SAVITCH, W. **C++ Absoluto**. Pretice-Hall, 2003.
- [6] KOFFMAN, E.B.; WOLFGANG, P.A.T. **Objetos, Abstração, Estruturas de Dados e Projeto Usando C++**. LTD, 2008.
- [7] DEITEL, H.M.; DEITEL, P.J. **Java: Como Programar**. 6ª ed. Pearson Education, 2005.

Referências Bibliográficas

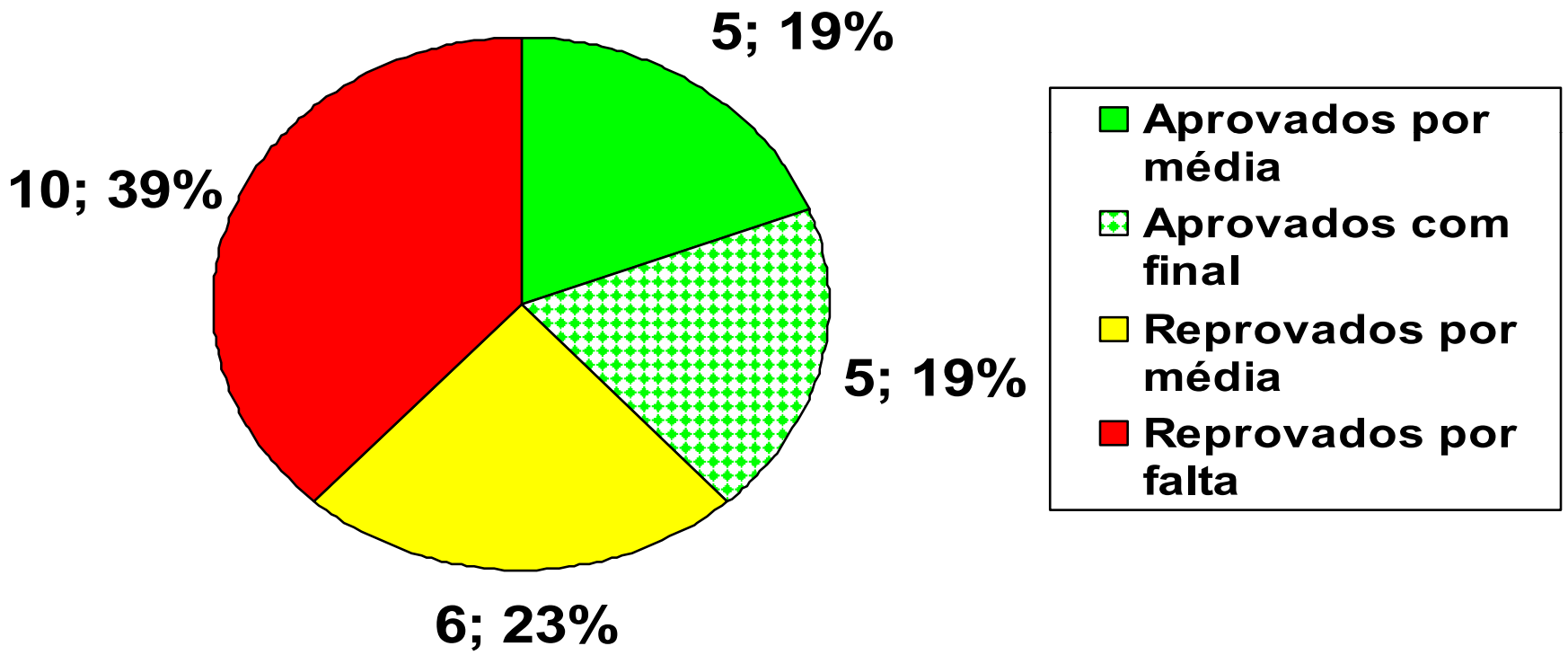
- [8] Page-Jones, Meilir. **O que todo programador deveria saber sobre projeto orientado a objetos.** Editora Makron Books, 1997.

Informações Gerais

- **Material de apoio**
  - Os slides utilizados em aula, PD, PUD, datas de avaliações e demais informações referentes à disciplina são encontradas na página www.univasf.edu.br/~marcelo.linder

Dados sobre a oferta anterior (2014.1)

- **Total de discentes:** 26
- **Percentual geral de aprovação:** 38,46%



Introdução

Introdução

O que vamos estudar?

Estudaremos os conceitos que nos possibilitaram responder a estas perguntas:

- O que é o paradigma orientado a objeto?
- Por que programar orientado a objeto?
- Necessito de uma linguagem de programação para estudar/compreender o paradigma orientado a objeto?
- Como representar meus sistemas desenvolvidos com o paradigma orientado a objeto?
- Quais linguagens utilizar para implementar meu sistema?
- Como utilizar estas linguagens?

UNIVASF

Introdução

O que é um paradigma?

Um modelo ou um padrão.

Em outras palavras: É uma forma de abordar um problema, segundo um conjunto de procedimentos, valores ou conceitos que direcionam o pensamento.

Introdução

Vocês já estudaram algum paradigma de programação?

Sim.

Quantos?

Um. Ou seriam dois?

Qual (is)?

O paradigma de programação estruturada.

O paradigma de programação procedural ou imperativa.

Obs.1: Na realidade a programação estruturada não é considera um paradigma de programação.

Obs.2: É possível intuir que nos primórdios da programação se programava sem uma metodologia, ou seja, a atividade de programar era muito peculiar a cada programador.

Gerando?

Uma insegurança na qualidade dos softwares produzidos.

Introdução

Por volta da década de 1970, **com o advento da programação estruturada**, ocorreu uma grande melhora na qualidade dos softwares produzidos e, por fim, ocorreu o surgimento do paradigma de programação imperativa.

Em que baseia-se a programação estruturada?

O princípio básico de programação estruturada é que um programa é composto por blocos elementares de código que se interligam através de três mecanismos básicos, que são **sequência**, **seleção** e **iteração**.

Introdução

Em que baseia-se o paradigma de programação imperativa?

O paradigma de programação imperativa baseia-se na resolução de um problema através da definição de procedimentos responsáveis pela resolução de subproblemas resultantes da decomposição do problema inicial em sub-problemas menores.

Considerando o problema da gestão acadêmica, como resolvê-lo com o paradigma de programação imperativa?

No caso de nossa instituição temos o SIG@...

Introdução

Como implementá-lo?

Começar por onde?

Definir suas funcionalidades:

- Lançar disciplinas;
- Especificar público para as vagas nas disciplinas;
- Efetuar matrícula;
- Lançar notas;
- Consultar média;
- Lançar faltas;

...

Introdução

Para determinarmos que é necessário “lançar disciplinas” e “especificar público para as vagas nas disciplinas” pensamos nas funções do

Para determinarmos que é necessário “efetuar matrícula” e “consultar média” pensamos nas atividades executadas pelo

Por fim, para determinarmos que é necessário “lançar notas” e “lançar faltas” pensamos nas funções do

Por que isso acontece?

Porque vivemos em um mundo composto por objetos.

Introdução

Em uma universidade temos pessoas que por sua vez são funcionários ou alunos, os funcionários são professores ou técnicos.

É fácil visualizar que estes objetos, ou melhor, componentes do sistema, possuem funções, por exemplo, os professores têm a função de registrar as notas, registrar as faltas e etc..

Devido a observação deste fato foi criado o paradigma orientado a objeto.

Se no nosso cotidiano estamos envoltos por objetos, por que não é ensinado apenas o paradigma da programação orientada a objeto ou por que este não é ensinado antes do paradigma de programação imperativa?

UNIVASF

Introdução

Porque o paradigma orientado a objeto requer um grau de abstração maior e, além disto, percebemos que mesmo no contexto da orientação a objeto temos funções desempenhadas pelos objetos.

Se a orientação a objeto pressupõe mais capacidade de abstração e sendo assim de análise, por que utilizá-la?

Visando uma maior facilidade na:

- compreensão do problema/solução;
- correção**/validação** do sistema desenvolvido**/proposto**;
- manutenção do sistema desenvolvido;
- reutilização do sistema desenvolvido;
- e extensão do sistema desenvolvido.

UNIVASF

Introdução

Por que?

É mais fácil compreender o que condiz com nosso cotidiano; É mais fácil corrigir o que se compreende melhor; A grande questão é que ao escrevermos o código-fonte de um programa utilizando a programação imperativa, ao existir a necessidade de modificarmos as características de um determinado objeto, encontraremos uma grande dificuldade em localizar quais as ações executadas por/com este objeto e quais destas devemos adequar às novas características; Além, do fato de reutilizar objetos ser muito mais plausível do que reutilizar funções; Estas características conduzem a uma maior facilidade na extensão.

UNIVASF