

Sobre a Disciplina

Programação Imperativa

Prof. Alberto Costa Neto

Prof. Giovanny Fernando Lucero Palma

Prof. Kalil Araujo Bispo

Prof. Rafael Oliveira Vasconcelos

DComp/UFS

Sobre a Disciplina PI

- **Disciplina:** Programação Imperativa (COMP0334)
- **Equivalentes:**
 - Introdução à Ciência da Computação
 - Programação Imperativa (COMP0197)
- **Carga horária:** 60 horas
- **Créditos:** 4

Ementa

Noções fundamentais sobre algoritmos e sobre a execução de programas. Análise e síntese de problemas. Identificadores, tipos, constantes, variáveis, tipos. Operadores e expressões. Comandos condicionais e de repetição. Variáveis compostas homogêneas e heterogêneas. Procedimentos, funções e passagem de parâmetros. Noções sobre o uso de arquivos em programação. Algoritmos básicos de ordenação. Recursividade. Uma linguagem imperativa. Convenções de código. Boas práticas de programação.

Objetivos

Geral

- Apresentar os conceitos básicos e principais técnicas de desenvolvimento de programas de computador, tornando-o apto a compreendê-los e aplicá-los.

Específicos

- Tornar o aluno capaz de implementar programas básicos usando uma linguagem de programação imperativa.
- Habilitar o aluno a criar programas para executar computação científica na sua área de conhecimento.
- Colocar em prática os conhecimentos aprendidos no curso, desenvolvendo aplicações de pequeno porte em Python.

Conteúdo Programático

1º Unidade

- Motivação para Programar
- Hardware, software e princípios
- Visão Geral da Linguagem Python
- Preparação do Ambiente de Desenvolvimento
- Instruções primitivas: atribuição, entrada e saída
- Expressões
- Tipos
- Comandos Condicionais (if)
- Tratamento de exceções (try / except)
- Funções
- Laço While
- Strings
- Laços For

2º Unidade

- Listas
- Recursividade
- Dicionários
- Tuplas
- Arquivos

Afinal, por que o nome PI?

- Vem da denominação do Paradigma que vamos estudar:
Paradigma Imperativo
- Você escreve explicitamente as ordens e o computador obedece
- Mais próximo do funcionamento real do computador
- Existem outros paradigmas, como por exemplo:
 - Funcional
 - Orientado a Objetos

Método de Ensino

Inovação na Disciplina de PI

- Queremos oferecer um **curso melhor**
- Usar **ferramentas modernas** de apoio pedagógico
- Aproveitar a característica da **nova geração estar sempre conectada à Internet**
- E sobretudo com um *Smartphone* sempre à mão

Metodologia - Presencial

- Conteúdo teórico estará disponível pela Internet
- Sistema que permite programar e tem autoavaliação
- Tempo de aula será focado em exercícios e tirar dúvidas

Metodologia - EaD

- A principal diferença é que **não haverá um horário fixo e obrigatório para realizar os exercícios e tirar dúvidas**
- Atividades podem ser feitas em casa ou laboratório.
- O aluno terá que cumprir as **mesmas atividades exigidas nas turmas presenciais**.
 - Caso tenha dúvidas, **deverá procurar professores e monitores (se houver) nos horários de atendimento**

Aulas Presenciais e Horários de atendimento

- As aulas presenciais e os horários de atendimento servirão para **tirar dúvidas e resolver exercícios**
- Sempre que possível, serão alocadas em **laboratório**
- Os professores irão comparecer às aulas (nas turmas presenciais) para:
 - Tirar **dúvidas referentes ao assunto visto nas videoaulas**
 - Tirar **dúvidas e resolver problemas do The Huxley em sala**
 - Levar soluções incompletas (pelo próprio The Huxley, via Pen Drive ou outro meio), para, com o auxílio do professor, fazer correções e submeter ao The Huxley

Recursos didáticos e AVA's

Recursos Didáticos

As aulas serão ministradas por meio da Internet, utilizando um software de videoconferência, com horários agendados previamente. As ferramentas utilizadas serão:

- **Youtube**, para exposição das videoaulas.
- **Google Jamboard**, para apresentação dos objetos de ensino.
- **Ferramentas de Videoconferência**: Google Meet
- **Editores de programas**: Repl.it, Notepad++ ou Sublime Text.
- **Interpretador da linguagem Python**, que permite a verificação de erros de sintaxe e execução de programas em Python.
- **Apps** que permitem elaborar, executar e testar programas em smartphones e tablets.
- **Ambientes Virtuais de Aprendizagem (AVA)** SIGAA e Google Classroom
- **Questionários e Atividades** via SIGAA
- **Questionários com Problemas de Programação** no site <http://thehuxley.com>

Correção de Questões

- Imagine se seu professor terá como corrigir 100 questões de cada um dos 50 alunos...
Façamos as contas:
 - São 5.000 questões!
 - Supondo que o professor gaste 6 min por questão, seriam necessários 30.000 minutos, ou seja, 500 horas!
- Seria interessante ter uma ferramenta que ajudasse o professor, concordam?



Fonte:
http://2.bp.blogspot.com/_Q4jxiezF5Hk/TNbebADQ2FI/AAAAAAAAABM/gnjeS8-S2I0/s1600/estres-laboral-y-enfermedad-periodontal.jpg

- Uma ferramenta Web que oferece um **banco de problemas de programação** (juiz *on-line*).
- Os **alunos podem enviar soluções** (programas em várias linguagens de programação).
- O **The Huxley executa a solução** com entradas presentes em casos de teste e compara com o resultado esperado.
- Com esta ferramenta o aluno tem um **feedback imediato**

The Huxley



Avaliação

Critério de Avaliação

Através de testes presenciais, obedecendo à fórmula:

$$\text{Nota Final} = (\text{NT1} + \text{NT2}) / 2$$

Onde:

- **NT1** = Nota do 1º Teste
- **NT2** = Nota do 2º Teste

Observação: Haverá um teste de reposição no final do semestre apenas para os alunos com falta justificada em algum teste, desde que a justificativa esteja prevista nas normas acadêmicas.

Desafios de Programação

- Durante o semestre haverá competições no estilo maratona de programação (desafios),
- Poderá obter até 1,5 (um ponto e meio) extras, ou seja, será uma nota adicionada à sua média geral, conforme sua classificação final na competição.

Pontuação dos Desafios

- No [ranking Top Coder](#) do site [The Huxley](#), o discente deverá ter, no dia da prova presencial da respectiva unidade, a pontuação mínima de:
 - 60 pontos na primeira unidade; e
 - 100 pontos na segunda unidade.

Pontuação dos Desafios

- Ordenados de forma decrescente em relação a esta pontuação e classificados em 4 grupos de tamanhos iguais, conforme a pontuação obtida. Em caso de divisão inexata, o resto da divisão ficará no grupo 4.
 - Grupo 1: 1,5 pontos extras iniciais.
 - Grupo 2: 1,0 ponto extra inicial.
 - Grupo 3: 0,5 ponto extra inicial.
 - Grupo 4: sem pontos extras.

Pontuação dos Desafios

- Além disso, o aluno terá que participar dos desafios de programação, cujas notas serão usadas para calcular a pontuação extra final do aluno, conforme fórmulas abaixo:
 - Pont extra unid 1 = Pont extra inicial x (nota desafio 1 + nota desafio 2)/20
 - Pont extra unid 2 = Pont extra inicial x (nota desafio 3 + nota desafio 4)/20

Calendário de Provas

As provas serão realizadas **presencialmente**:

- No horário da aula
- Segundo **calendário** e **orientações** divulgados nos AVA's

Controle de Frequência

Controle de Frequência (Turmas Presenciais)

- O aluno não é obrigado a estar presencialmente nas aulas, desde que cumpra com as atividades on-line.
- Assim, a frequência dos alunos será computada através de:
 - Lista de presença nos dias das aulas presenciais; OU
 - Através da realização das atividades on-line.
- No final do semestre, as faltas de quem realizou as atividades on-line serão abonadas de acordo com a planilha.

Controle de Frequência (Turma EaD)

- Será obrigatório participar das aulas presenciais.
- A carga horária restante será calculada em função do cumprimento das atividades on-line.
 - Atividades no The Huxley

Bibliografia

Referências Bibliográficas (Básica)

- **Fundamentos da Programação de Computadores**. Ana Fernanda Gomes Ascencio / Edilene Aparecida Veneruchi De Campos. 3º edição; 2012, Pearson; ISBN 978-8564574168
- **Algoritmos e Lógica de Programação**. Marco A. Furlan de Souza, Marcelo M. Gomes, Marcio V. Soares, Ricardo Concilio. Editora Cengage Learning, 2ª edição, 2011.
- **Algoritmos: Lógica para Desenvolvimento de Programação de Computadores**. José Augusto N. G. Manzano, Jayr Figueiredo de Oliveira. Editora Érica, 17ª edição, 2005.
- **Python for Everybody: Exploring Data Using Python 3**. Charles R. Severance. CreateSpace Independent Publishing Platform; 1st. ed., 2016; ISBN: 978-1530051120

Referências Bibliográficas (Complementares)

- **Python for Everybody: Exploring Data Using Python** Charles R. Severance. CreateSpace Independent Publishing Platform; 1st. ed., 2016
- **Python for Informatics: Exploring Information**. Charles R. Severance. CreateSpace Independent Publishing Platform; 1st. ed., 2013
- **Como pensar como um Cientista da Computação usando Python (traduzido)**. Allen Downey, Jeffrey Elkner, and Chris Meyers. 2002.
- **Introdução à Programação com Python**. Nilo Ney Coutinho, 2º edição, 2014, ISBN: 978-85-7522-408-3.
- **Python para Desenvolvedores**. Luiz Eduardo Borges. Rio de Janeiro; 2010
- **Learning to Program Using Python**. Cody Jackson. CreateSpace Independent Publishing Platform

Contatos dos Professores

- Alberto Costa Neto (albertocn@dcomp.ufs.br) – T10, T17 e T18
- Giovanni Fernando Lucero Palma (giovanny@dcomp.ufs.br) - T12 e T16
- Kalil Araujo Bispo (kalil@dcomp.ufs.br) – T02, T04 e T05
- Rafael Oliveira Vasconcelos (rafael@dcomp.ufs.br) – T05

Como proceder em caso de dificuldade?

- Sempre que identificar alguma dificuldade, dúvida sobre conceitos das videoaulas ou problemas, **entre em contato com o(s) professor(es)** responsáveis pela sua turma.
- Caso não consiga **acessar os AVAs ou sites**, também entre em contato com o(s) professor(es).

Não deixe de tirar suas dúvidas!

E sejam bem-vindos ao curso de PI!!!