Plano de Ensino

1 Código e nome da disciplina

ARA0066 PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO EM PYTHON

2 Carga horária semestral

80

3 Carga horária semanal

3 horas-aulas práticas presenciais + 1 hora-aula digital

4 Perfil docente

O docente deve ser graduado em Ciência da Computação, Informática, Engenharias, com forte aderência em programação de computadores.

O Professor deve possuir Pós-Graduação Lato Sensu, sendo desejável Pós-Graduação Stricto Sensu (Mestrado ou Doutorado) na área do curso ou afins.

É desejável que o professor possua experiência de na docência no nível superior da disciplina de pelo menos 3 anos e que também tenha experiência profissional no mercado de trabalho de programação de computadores. Deve possuir familiaridade com as ferramentas digitais que fazem parte do modelo de ensino da instituição (SGC, SIA, SAVA, BDQ) além de conhecer o Projeto Pedagógico dos Cursos dos quais a disciplina faz parte na Matriz Curricular.

É necessário que o docente conheça e aplique as metodologias ativas inerentes à educação por competências e ferramentas digitais que tornam a sala de aula mais interativa e interessante para o aluno. A articulação entre a teoria e prática deve ser o eixo direcionador das estratégias em sala de aula. Além disto, é indispensável que o docente estimule o autoconhecimento e autoaprendizagem entre seus alunos.

5 Ementa

PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO: MOTIVAÇÃO E PRELIMINARES. NOMES, VINCULAÇÕES E ESCOPO. TIPOS DE DADOS. EXPRESSÕES E SENTENÇAS DE ATRIBUIÇÃO. SUBPROGRAMAS. PARADIGMAS: ESTRUTURADO, ORIENTADO A OBJETOS, FUNCIONAL E LÓGICO.

6 Objetivos

- Distinguir as categorias de linguagens de programação, fazendo a reflexão sobre os projetos de

linguagens de programação, suas categorias e métodos de implementação, para decidir qual paradigma de linguagem de programação utilizar, conforme a classe de problema;

- Caracterizar a natureza dos nomes e palavras especiais nas linguagens de programação, baseando-se na linguagem Python, para empregar as regras de escopo;

- Especificar variáveis, empregando tipos de dados, de forma a contextualizar ao compilador/interpretador como o programador pretende utilizar os dados;

- Empregar formas fundamentais de instruções, baseando-se na sintaxe e semântica de expressões aritméticas, relacionais e booleanas e atribuições, para escrever instruções matemática e lógicas compreensíveis, corretas e executáveis por computadores;

- Escrever programas modularizados, baseando-se em fundamentos de subprogramas, para decompor problemas complexos em fragmentos mais simples, ou seja, mais facilmente tratáveis, cujos códigos sejam reutilizáveis e manuteníveis;

- Praticar a codificação de soluções, utilizando diferentes paradigmas de linguagem de programação, para resolver problemas aplicando o paradigma mais apropriado.

7 Procedimentos de ensino-aprendizagem

A disciplina adotará o modelo de sala de aula invertida e aprendizagem baseada em problemas, seguidas do detalhamento de cada tópico previsto neste plano. O professor será responsável pela contextualização do tema relacionando com as práticas do mercado de trabalho. Além disso poderá utilizar de exercícios e atividades que exemplifique e estimulem o aluno promover o conhecimento de forma orgânica, sempre evidenciando os objetivos de cada tema. O processo de ensino-aprendizagem será baseado em 3 etapas: a preleção, a partir da definição de uma situação problema (temática/problematização/pergunta geradora), utilização de metodologias ativas centradas no protagonismo do aluno e realização de uma atividade verificadora da aprendizagem ao final da aula.

O processo de ensino-aprendizagem priorizará o aluno, sendo este capaz de articular os temas discutidos nas aulas para responder à situação problema que abre a preleção. É Importante destacar o uso da Sala de Aula Virtual de Aprendizagem (SAVA), através do Webaula, onde o aluno terá acesso ao conteúdo digital da disciplina, poderá resolver questões propostas e explorar conteúdos complementares.

O modelo de aprendizagem prevê a realização da Atividade Autônoma Aura - AAA: duas questões elaboradas para avaliar se os objetivos estabelecidos, em cada plano de aula, foram alcançados pelos alunos. A Atividade Autônoma Aura - AAA tem natureza diagnóstica e formativa, suas questões são fundamentadas em uma situação-problema, estudada previamente, e cuja resolução permite aferir o aprendizado do(s) tema/tópicos discutidos na aula.

8 Temas de aprendizagem

1. PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO: MOTIVAÇÃO E PRELIMINARES
1.1 RAZÕES PARA ESTUDAR CONCEITOS DE LINGUAGENS DE PROGRAMAÇÃO
1.2 DOMÍNIOS DE PROGRAMAÇÃO
1.3 TRADE-OFFS NO PROJETO DE LINGUAGENS

2. NOMES, VINCULAÇÕES E ESCOPO
2.1 VARIÁVEIS
2.2 ESCOPO

3. TIPOS DE DADOS
3.1 PRIMITIVOS
3.2 AGLOMERADOS
3.3 PONTEIROS E REFERÊNCIAS
3.4 VERIFICAÇÃO DE TIPOS
3.5 TEORIA E TIPOS DE DADOS
3.6 DADOS ABSTRATOS E ENCAPSULAMENTO

4. EXPRESSÕES E SENTENÇAS DE ATRIBUIÇÃO
4.1 INTRODUÇÃO À EXPRESSÕES E SENTENÇAS DE ATRIBUIÇÃO
4.2 SENTENÇAS DE ATRIBUIÇÃO

5. SUBPROGRAMAS
5.1 FUNDAMENTOS DOS SUBPROGRAMAS
5.2 QUESTÕES DE PROJETO PARA SUBPROGRAMAS

6. PARADIGMAS: ESTRUTURADO, ORIENTADO A OBJETOS, FUNCIONAL E LÓGICO (CRÉDITO DIGITAL)
6.1 PARADIGMA ORIENTADO A OBJETO
6.2 PARADIGMA FUNCIONAL
6.3 PARADIGMA LÓGICO

9 Procedimentos de avaliação

Os procedimentos de avaliação contemplarão competências desenvolvidas durante a disciplina nos âmbitos presencial e digital. Indicações para procedimentos e critérios de avaliação:

- As avaliações serão presenciais e digitais, alinhadas à carga-horária da disciplina, divididas da seguinte forma:

Avaliação 1 (AV1), Avaliação 2 (AV2), Avalição Digital (AVD) e Avaliação 3 (AV3):

* AV1 - Contemplará os temas abordados na disciplina até a sua realização e será assim composta:
- Prova individual com valor total de 7 (sete) pontos;
- Atividades acadêmicas avaliativas com valor total de 3 (três) pontos, assim distribuídos:
- Atividade da Aula 03 sobre o tempo de execução de código em diferentes linguagens, com valor de 1 ponto
- Atividade da Aula 04 envolvendo exercícios de desenvolvimento em Python, com valor de 1 ponto
- Atividade da Aula 05 envolvendo mais exercícios de desenvolvimento em Python, com valor de 1 ponto
A soma de todos os instrumentos que possam vir a compor o grau final da AV1 não poderá ultrapassar o grau máximo de 10 (dez) pontos.

* AV2 - Contemplará todos os temas abordados pela disciplina e será composta por uma prova teórica no formato PNI - Prova Nacional Integrada, no seguinte formato: PNI de 0 a 5,0. As demais atividades acadêmicas avaliativas devem somar 5 (cinco) pontos.
- Atividade da Aula 10 sobre subprogramas, com valor de 1 ponto
- Atividade da Aula 11 sobre bibliotecas para criação de uma calculadora, com valor de 1 ponto
- Atividade da Aula 12 sobre interface gráfica em python usando Tkinter com valor de 1 ponto
- Atividade da Aula 13 envolvendo a implementação de um Tamagoshi (animal virtual), com valor de 2 ponto

* AVD - Avaliação digital do(s) tema(s) / tópico(s) vinculado(s) ao crédito digital no valor total de 10

(dez) pontos ou AVDs - Avaliação digital do(s) tema(s) / tópico(s) vinculado(s) ao crédito digital no valor total de 10 (dez) pontos.

* AV3 - Contemplará todos os temas abordados pela disciplina. Será composta por uma prova no formato PNI - Prova Nacional Integrada, com total de 10 pontos, substituirá a AV1 ou AV2 e não poderá ser utilizada como prova substituta para a AVD.

Para aprovação na disciplina, o aluno deverá:
- atingir resultado igual ou superior a 6,0, calculado a partir da média aritmética entre os graus das avaliações presenciais e digitais, sendo consideradas a nota da AVD ou AVDs e apenas as duas maiores notas entre as três etapas de avaliação presencial (AV1, AV2 e AV3). A média aritmética obtida será o grau final do aluno na disciplina;
- obter grau igual ou superior a 4,0 em, pelo menos, duas das três avaliações presenciais;
- frequentar, no mínimo, 75% das aulas ministradas.

10 Bibliografia básica

PERKOVIC, Ljubomir. **Introdução à Computação Usando Python - Um Foco no Desenvolvimento de Aplicações**. 1ª Ed. Rio de Janeiro: LTC, 2016.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788521630937/

SEBESTA, Robert W. **Conceitos de Linguagens de Programação**. 11ª Ed. Porto Alegre: Bookkman, 2018.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788582604694/

TUCKER, Allen; NOONAN, Robert. **Linguagens de Programação: Princípios e Paradigmas**. 2ª Ed. Porto Alegre: AMGH, 2014.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788563308566/

11 Bibliografia complementar

AGUILAR, Luis Joyanes. **Programação em C++: Algoritmos, estruturas de dados e objetos**. 2ª Ed.. Porto Alegre: Mc Graw Hill, 2005.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788580550269/

ASCENCIO, Ana F. G.; DE CAMPOS, Edilene A. V. **Fundamentos da Programação de Computadores**. 3º Ed.. Rio de Janeiro: Pearson, 2012.
Disponível em: https://plataforma.bvirtual.com.br/Acervo/Publicacao/3272

MANZANO, José Augusto N. G.; COSTA JR., Roberto A. **Programação de Computadores com Java**. 1ª Ed.. São Paulo: Saraiva, 2014.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788536519494/

MENEZES, Alexandre Moreira. **Os Paradigmas de Aprendizagem de Algoritmo Computacional**. 1ª Ed.. São Paulo: Editora Blucher, 2015.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788580391039/

PINHEIRO, Francisco de A. C. **Elementos de Programação em C**. 1ª Ed.. Porto Alegre: Bookman, 2012.
Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788540702035/