PLANO DE ENSINO

ANO	PERÍODO	DISCIPLINA	CARGA HORÁRIA
2024	1.2	CB 23 - Programação 2	54 h

PROFESSOR(A)

Emílio Vital Brazil

LIVRO(S) DE REFERÊNCIA

1. RAMALHO, L. Python Fluente: Programação Clara, Concisa e Eficaz, São Paulo: Novatec, 2015

2. DOWNEY, A. Pense em Python. São Paulo: Novatec, 2016.

3. CORMEN, T.; LEISERSON, C.; RIVEST, R.; STEIN, C. Algoritimos, quarta edição. São Paulo: LTC, 2024.

4. BURDEN, R.; FAIRES, J.D.; BURDEN, A., Análise Numérica, tradução da 10a. edição americana. São Paulo: Cengage Learning, 2016

PRÉ-REQUISITOS

AVALIAÇÃO

Critérios gerais do curso

1. O discente com frequência inferior a 75% estará reprovado na disciplina, sendo-lhe atribuído a informação de Reprovação por Faltas.
2. Todas as notas são na escala de 0 a 10, com apenas um dígito decimal. Eventuais arredondamentos serão para o dígito decimal imediatamente superior.
3. A média final (MF) da disciplina é a média MF=(AV1+AV2)/2 dos resultados da AV1 e AV2.
   O discente fica: APROVADO se MF ≥ 6,0 e admitido a fazer a AVS se MF ≤ 5,9.
4. Para o discente que fizer a AVS, o resultado final (RF) da disciplina é RF=(MF+AVS)/2.
   O discente fica: APROVADO se RF ≥ 6,0 e REPROVADO se RF ≤ 5,9.
5. A participação nas avaliações regulares (AV1 e a AV2) é obrigatória: qualquer ausência terá que ser justificada pelo discente, sob pena de receber nota 0 (zero) na respectiva avaliação. A justificativa estará sujeita a validação pelo IMPA.
6. O discente que fizer as duas avaliações regulares e for aprovado por média final (MF ≥ 6,0) pode optar por fazer a AVS para melhoria de nota, mas nesse caso poderá igualmente ter a sua nota rebaixada (caso RF < MF).
7. O discente que deixar de fazer alguma das avaliações regulares e tiver sua(s) justificativa(s) aceita(s) pelo IMPA, estará automaticamente admitido a fazer a AVS e terá seu resultado final dado por:
   RF=(AV1+AVS)/2 se tiver feito apenas a AV1,
   RF=(AV2+AVS)/2 se tiver feito apenas a AV2, e
   RF = AVS se não tiver feito nenhuma das duas avaliações regulares,
ficando: APROVADO se RF ≥ 6,0 e REPROVADO se RF ≤ 5,9.

Critérios específicos da disciplina

AV1 Composição: 0.6 x avaliação escrita + 0.4 x lista de exercícios 1 a 4
AV2 Composição: 0.6 x avaliação escrita + 0.4 x lista de exercícios 5 a 8
AVS Composição: avaliação escrita

PLANO DE AULAS					
Aula	**Tipo**	**Data**	**Conteúdo**	**Referência**	**Entregas**
1	Teórica	segunda-feira, agosto 19	Apresentação da ementa: revisão dos conceitos básicos, apresentação da dinâmica do curso.		
	Prática				
2	Teórica	quarta-feira, agosto 21	Instalar e testar a biblioteca Numpy. Discutir o tamanho em memória de tipos de dados. Discutir a representação por array. Relembrar o conceito de listas.	https://numpy.org/	
	Prática				
3	Teórica	segunda-feira, agosto 26	Os conceitos fundamentais da Programação Orientada a Objetos (POO): como criar classes, objetos, atributos e métodos em Python. A importância da encapsulação, herança e polimorfismo na POO.	2. Cap: 16, 17 e 18	
	Prática				
4	Teórica	quarta-feira, agosto 28	Exercício sobre POO - Espaço vetorial.	2. Cap: 16, 17 e 18	
	Prática				
5	Teórica	segunda-feira, setembro 2	Conceitos fundamentais do modelo de dados do Python. Explorar as operações e métodos especiais que permitem a personalização de objetos em Python.	1. Cap. 1	
	Prática				
6	Teórica	quarta-feira, setembro 4	Aplicar os conhecimentos do modelo de dados em situações práticas.	1. Cap. 1	
	Prática				
7	Teórica	segunda-feira, setembro 9	Diferenças entre listas padrão do Python, arrays e numpy.arrays. Principais características e vantagens do numpy.array em relação às listas. A classe numpy.array e novos métodos personalizados.	1. Cap. 2	
	Prática				
8	Teórica	quarta-feira, setembro 11	Conceitos básicos da complexidade computacional. As notações comuns usadas para descrever complexidade.	Notas de Geometria computacional	
	Prática				
9	Teórica	segunda-feira, setembro 16	Complexidade da ordenação - limites Theta e Omega. Conceito de redução e seu papel na teoria da complexidade.	Notas de Geometria computacional	
	Prática				
10	Teórica	quarta-feira, setembro 18	Conceito de números em ponto flutuante (floating point) na computação. Conceitos de precisão e exatidão aplicados no sistema de ponto flutuante.	4. Cap. 1	Lista 1
	Prática				
11	Teórica	segunda-feira, setembro 23	Tipos de erros de ponto flutuante, incluindo erros de truncamento, arredondamento, representação e cálculos iterativos, fornecendo exemplos para cada um. Técnicas para mitigar erros numéricos e suas possíveis causas.	4. Cap. 1	

	Prática				
12	Teórica	quarta-feira, setembro 25	Conceito de filas e pilhas como estruturas de dados. Aplicar o conceito de fila de prioridade entendendo a sua complexidade. Exemplificar aplicações de pilhas e filas usando Python.	3. Cap. 10	Lista 2
	Prática				
13	Teórica	segunda-feira, setembro 30	Conceito de árvores e árvores binárias. Como implementar árvores binárias em Python. Operações básicas em árvores binárias, como inserção, remoção e travessia.	3. Cap. 12	
	Prática				
14	Teórica	quarta-feira, outubro 2	Exercício de implementação de árvores binárias.	3. Cap. 12	Lista 3
	Prática				
15	Teórica	segunda-feira, outubro 7	O que é o módulo SciPy e por que é importante para a programação científica. Como importar o SciPy e seus submódulos. Explorar algumas das principais funcionalidades do SciPy, incluindo estatística e interpolação.	https://scipy.org/	
	Prática				
16	Teórica	quarta-feira, outubro 9	Revisão para a AV1.		Lista 4
	Prática				
	P1	sexta-feira, outubro 18			
17	Teórica	segunda-feira, outubro 21	Correção da AV1.		
	Prática				
18	Teórica	quarta-feira, outubro 23	Conceito de grafos como uma estrutura de dados abstrata. Os componentes fundamentais de um grafo.	3. Cap. 20	
	Prática				
19	Teórica	segunda-feira, outubro 28	Representação de grafos por meio de lista de arestas, matrizes de adjacência e listas de adjacência. Explorar algumas aplicações práticas de grafos.	3. Cap. 20	
	Prática				
20	Teórica	quarta-feira, outubro 30	Os conceitos fundamentais da programação funcional. Aprender a usar funções de ordem superior em Python. Ferramentas como map(), filter() e reduce() para operações funcionais.	1. Cap. 7	Lista 5
	Prática				
21	Teórica	segunda-feira, novembro 4	O conceito de funções lambda em Python. A compreensão de listas simplifica a criação de listas em Python. A compreensão de listas como opção para map() e filter(). Praticar a criação de funções lambda e a utilização de	1. Cap. 7	
	Prática				
22	Teórica	quarta-feira, novembro 6	O conceito de raiz de uma função real. Aplicar algoritmos como o Método da Bissecção e o Método de Newton-Raphson, e código Python para resolver problemas de raízes de funções.	4. Cap. 2	Lista 6
	Prática				
23	Teórica	segunda-feira, novembro 11	Conceito de interpolação e extrapolação, métodos de interpolação polinomial. Criticar a interpolação polinomial.	4. Cap. 3	

	Prática				
24	Teórica	quarta-feira, novembro 13	Principais tipos de modelos de ajuste de curva e algoritmos. Exercício de ajuste de curvas.	4. Cap. 3	Lista 7
	Prática				
	Teórica	segunda-feira, novembro 18	Feriado		
	Prática				
	Teórica	quarta-feira, novembro 20	Feriado		Lista 8
	Prática				
	P2	segunda-feira, novembro 25			
25	Teórica	segunda-feira, dezembro 2	Correção da AV2.		
	Prática				
26	Teórica	quarta-feira, dezembro 4	Como se preparar para uma entrevista técnica.		
	Prática				
	AVS	segunda-feira, dezembro 9 quinta-feira, dezembro 12	(segunda chamada)		