UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Faculdade de Computação
Av. João Naves de Ávila, nº 2121, Bloco 1A - Bairro Santa Mônica, Uberlândia-MG, CEP 38400-902
Telefone: (34) 3239-4144 - http://www.portal.facom.ufu.br/ facom@ufu.br

PLANO DE ENSINO

1. IDENTIFICAÇÃO

Componente Curricular:	PROGRAMAÇÃO DE COMPUTADORES						
Unidade Ofertante:	Faculdade de Computação - FACOM						
Código:	FACOM39201	Período/Série:	2º	Turma:			
Carga Horária:					Natureza:		
Teórica:	30	Prática:	30	Total: 60	Obrigatória: (x)	Optativa: ( )	
Professor(A):	Shigueo Nomura				Ano/Semestre:	2023/2	
Observações:							

2. EMENTA

Tipos de dados. Variáveis e constantes. Expressões e operadores.

Estruturas de controle: estruturas básicas, estruturas condicionantes e estruturas de repetição.

Estruturas básicas de dados: vetores, matrizes. Funções. Arquivos.

3. JUSTIFICATIVA

O Python é uma linguagem de programação de alto nível e por isso, os seus programas se tornam fáceis de ler e entender.

A linguagem Python tem a vantagem de melhorar a produtividade dos programadores, já que facilita a programação. Em razão de sua simplicidade, é importante ressaltar que os desenvolvedores conseguem focar na solução do problema sem necessidade de se preocupar com a sintaxe da linguagem.

Assim, em comparação com outras linguagens, o ato de programar em Python é mais fácil, possibilitando a solução de problemas de forma mais rápida e eficiente.

4. OBJETIVO

Geral:

Preparar o estudante para que seja capaz de desenvolver programas em linguagem Python, empregando adequadamente os recursos oferecidos por esta linguagem.

Objetivos Específicos:

- Revisar comandos básicos da linguagem Python
- Dominar comandos para estruturas de seleção e de repetição
- Dominar definição e chamada de funções
- Ter noções de recursividade
- Dominar comandos para manipulação de strings
- Dominar comandos para manipulação de listas e tuplas
- Dominar importação de módulos
- Ter noções de manipulação de arquivos
- Dominar manipulação de dicionários

5. PROGRAMA

1. Revisão de linguagem Python: variáveis, tipos básicos, operações aritméticas, entrada/saída (input/print)

2. Estruturas de seleção

2.1. Operadores lógicos: *and*, *or*, *not.*

2.2. Operadores de comparação.

2.3 Condicionais: *if/else, if/elif/else.*

3. Estruturas de repetição

3.1. Cláusula *while.*

3.2. Cláusula *for*: uso em conjunto com a função range.

3.3. Cláusulas *break* e *continue.*

4. Funções

4.1. Definição de funções.

4.2. Escopo de função.

4.3. Variáveis locais e globais.

5. Strings

5.1. Definição de strings.

5.2. Operações com strings.

5.3. Codificação ASCII e funções *ord* e *chr.*

5.4. Métodos básicos.

6. Listas e tuplas (vetores)

6.1. Definição de listas (*list*).

6.2. Operações com listas e métodos básicos.

6.3. Compreensão de listas (*list comprehension*).

6.4. Listas aninhadas (matrizes): introdução e operações básicas com matrizes numéricas: soma, traço,multiplicação por escalar.

6.5. Tuplas (*tuple*).

7. Introdução às funções recursivas

8. Módulos

8.1. Importação de módulos próprios.

8.2. Importação de módulos pré-definidos, módulos *math, random.*

9. Introdução a arquivos

9.1. Leitura e escrita de arquivos texto.

9.2. Serialização/desserialização de objetos em arquivos com módulo *pickle*: funções *pickle.dump* e *pickle.load.*

10. Dicionários

10.1. Definição de dicionários (*dict*).

10.2. Operações e métodos básicos com dicionários

6. METODOLOGIA

Essencialmente, a metodologia do curso será do tipo ativa em que o estudante deverá atuar como agente participativo (não passivo) do processo de aprendizagem de programação em Python. O interesse em aprender deverá partir do estudante e este será estimulado a estudar no seu ritmo e a procurar o professor para orientações ou esclarecimento de dúvidas conforme necessário. O professor terá o importante papel de orientador e facilitador desse processo de aprendizagem.

A. Atividades presenciais

De acordo com a ficha de disciplina, está prevista uma carga horária semanal de quatro horas-aula para as atividades presenciais.

A tabela a seguir apresenta a carga horária em horas-aula, considerando a distribuição prevista entre as 16 semanas de aulas do calendário acadêmico, totalizando 72 horas-aula.

Semana	Data	Conteúdo	Atividade presencial	TDE
1	08/01/24	Apresentação do curso	2	
1	09/01/24	Revisão de linguagem Python	2	
2	15/01/24	Revisão de linguagem Python	2	
2	16/01/24	Revisão de linguagem Python	2	
		Revisão de linguagem Python		1
3	22/01/24	Estruturas de seleção	2	
3	23/01/24	Estruturas de seleção	2	
4	29/01/24	Estruturas de seleção	2	
		Estruturas de seleção		1
4	30/01/24	Estruturas de repetição	2	
5	05/02/24	Estruturas de repetição	2	
		Estruturas de repetição		1
5	06/02/24	Funções	2	
6	12/02/24	Feriado	0	
6	13/02/24	Feriado	0	
7	19/02/24	Atividade avaliativa 1	2	
7	20/02/24	Funções	2	
8	26/02/24	Funções	2	
		Funções		1
8	27/02/24	Strings	2	
9	04/03/24	Strings	2	

		Strings		1
9	05/03/24	Listas e tuplas (vetores)	2	
10	11/03/24	Listas e tuplas (vetores)	2	
10	12/03/24	Listas e tuplas (vetores)	2	
		Listas e tuplas (vetores)		1
11	18/03/24	Introdução às funções recursivas	2	
11	19/03/24	Atividade avaliativa 2	2	
12	25/03/24	Introdução às funções recursivas	2	
		Introdução às funções recursivas		1
12	26/03/24	Módulos	2	
13	01/04/24	Módulos	2	
		Módulos		1
13	02/04/24	Introdução a arquivos	2	
14	08/04/24	Introdução a arquivos	2	
		Introdução a arquivos		1
14	09/04/24	Dicionários	2	
15	15/04/24	Dicionários	2	
		Dicionários		1
		Desenvolvimento do projeto prático		2
15	16/04/24	Entrega do projeto prático	2	
16	22/04/24	Atividade avaliativa 3	2	
16	23/04/24	Recuperação	2	
		Total de horas-aula	60	12

A distribuição da carga horária prevista para cada um dos 10 itens do conteúdo programático poderá sofrer ajustes conforme o ritmo de aprendizagem ou as necessidades dos estudantes ou ainda, as circunstâncias do curso. O trabalho discente efetivo (TDE) se refere à carga horária para resolver exercícios de fixação do item de conteúdo programático indicado e para o desenvolvimento do projeto prático. A carga horária do TDE poderá ser distribuída pelo estudante durante a semana conforme as suas disponibilidades.

B. Trabalho discente efetivo (TDE)

Conforme se verifica no cronograma de atividades, a carga horária prevista para a realização das atividades presenciais é de 60 horas-aula. Assim, para completar as 72 horas-aula da disciplina, estão previstas no mínimo 12 horas-aula a serem dedicadas pelo estudante para exercer o TDE referente a cada um dos itens do conteúdo programático, ao longo do semestre.

As atividades do TDE deverão ser realizadas para a consolidação dos itens do conteúdo programático, utilizando os materiais de apoio fornecidos pelo professor e informações complementares pesquisadas na Internet. Além disso, a carga horária deverá ser utilizada para execução de exercícios e trabalhos e principalmente para aquisição das habilidades de programação, partindo-se da premissa de que tais habilidades só se adquirem com a prática.

C. Controle de frequência

O controle de frequência será realizado através da chamada ou assinatura da lista de presença em sala durante as aulas. Também, o controle de frequência será realizado pela entrega dos exercícios de fixação e trabalhos nos prazos estipulados, no caso do TDE. Não serão abonadas faltas por motivos alheios aos casos previstos nas normas do CONGRAD.

D. Técnicas de ensino e aprendizagem

Em suma, serão adotadas as seguintes técnicas de ensino e aprendizagem:

· Aulas expositivas (com uso de quadro e giz ou de recursos audiovisuais) abordando os conceitos e fundamentos teóricos do programa da disciplina;

· Execução ou resolução de estudos dirigidos, de atividades avaliativas periódicas pelos estudantes (de forma individual ou em grupo) obedecendo aos prazos de entrega estabelecidos (considerando o ritmo de aprendizagem da turma);

· Aplicação de técnicas da metodologia ativa para estimular a discussão em grupo e promover a troca de experiências entre os estudantes, auxiliando na consolidação dos fundamentos teóricos e aquisição de habilidades práticas;

· Codificação de programas pelos estudantes utilizando os comandos relacionados a cada item do conteúdo programático sob a orientação e supervisão do professor. Os estudantes deverão de forma intensiva, utilizar e explorar o software (Python) como ferramenta de programação abrangendo todos os itens do conteúdo programático;

· Atendimento individual ou coletivo pelo professor de forma remota ou presencial para esclarecimento de dúvidas mediante agendamento através do e-mail shigueonomura@ufu.br em horário conveniente para ambos os lados. Conforme a necessidade e conveniência, poderá ser fixado um horário semanal para atendimento dos estudantes.

E. Acesso às plataformas e ao material bibliográfico e de apoio

Todo o material bibliográfico ou de apoio (apostilas, resumos, vídeos, estudos dirigidos, slides, exercícios, capítulos de livros digitais) necessário ao acompanhamento pleno do curso será postado na plataforma Moodle ou equivalente, devidamente organizado conforme os módulos do conteúdo programático. As senhas ou chaves de acesso às plataformas serão fornecidas pelo professor nas primeiras aulas da disciplina.

Para a realização das atividades, o estudante deverá providenciar meios (equipamentos computadorizados e conexões com a Internet) apropriados. Tais meios deverão permitir acesso adequado às plataformas e oferecer condições apropriadas de trabalho. Conforme a necessidade, o professor fornecerá o instalador (software livre com baixa carga computacional) para a codificação e execução dos programas.

7. AVALIAÇÃO

· A média das atividades avaliativas será calculada pela seguinte equação:

$MA=(A1+A2\cdots A3)/3$ para $i=1,\ 2,\ 3$ onde $Ai\leqslant 100\Rightarrow$ notas de atividades avaliativas periódicas que serão aplicadas respeitando-se o ritmo de aprendizagem dos estudantes.

As atividades avaliativas serão individuais ou em grupos, dissertativas ou objetivas, dependendo do conteúdo programático a ser avaliado. Os detalhes do tipo de avaliação serão devidamente informados e explicados aos estudantes com antecedência suficiente para se prepararem para a atividade avaliativa.

De acordo com as normas do CONGRAD, está prevista uma avaliação de recuperação de aprendizagem que substituirá a nota MA. A prova substitutiva abrangerá todo o conteúdo visto no semestre. Ainda, de acordo com as referidas normas, somente fará jus ao direito de realizar a avaliação de recuperação

substitutiva, o estudante que não obtiver o rendimento mínimo de aprovação (60 pontos) e que possuir no mínimo 75% de frequência na disciplina. Após a avaliação de recuperação, caso o estudante atinja pontuação suficiente, a média final será de 60 pontos.

As datas previstas para as atividades avaliativas obedecem ao seguinte cronograma:

Atividade avaliativa 1	19/2/2024
Atividade avaliativa 2	19/3/2024
Atividade avaliativa 3	22/4/2024
Recuperação	23/4/2024

· A média dos exercícios de fixação referentes às atividades extracurriculares será calculada pela seguinte equação:

$ME=(E1+E2\cdots En)/n$ para $j=1, 2,\cdots n$ onde $n\geqslant 3$ e $Ej\leqslant 100\Rightarrow$ notas de exercícios de fixação (individuais ou em grupos) para apoio ao aprendizado de cada item do conteúdo programático e ainda para a complementação da carga horária do curso. A data de entrega de cada exercício será agendada respeitando-se a execução do cronograma e o ritmo de aprendizagem dos estudantes. Haverá um desconto de 10% (sobre a nota original do exercício corrigido) por dia de atraso na entrega.

· Nota referente à avaliação de projeto prático desenvolvido em grupo:

$$0\leq PP\leq 100$$

A data prevista para a entrega do projeto prático é 16/4/2024.

· A média final será calculada pela seguinte equação:

$$MF=(0,6\times MA+0,4\times ME+0,5\times PP)/1,5$$

O estudante será considerado aprovado ao atingir MF maior ou igual a 60 pontos e assiduidade maior ou igual a 75% da carga horária total para o curso.

8. BIBLIOGRAFIA

Básica

LUTZ, M. **Learning Python**. O'Reilly, 2013.

MATHES, E. **Curso Intensivo de Python**. Novatec, 2016.

MENEZES, N. C. **Introdução à Programação com Python**. Novatec, 2010.

Complementar

BAKA, B. **Python Data Structures and Algorithms**: Improve application performance with graphs, stacks, and queues. Packt Publishing, 2017.

GOODRICH, M. T; TAMASSIA, R.; GOLDWASSER, M.H. **Data structures and algorithms in Python**. John Wiley & Sons Ltd, 2013.

HETLAND, M. L. **Beginning Python: from novice to professional**. New York, Apress, 2008.

MORAES, C. R. **Estruturas de dados e algoritmos: uma abordagem didática**. 2. ed. São Paulo: Futura,2003.

SWIGART, A. **Automatize Tarefas Maçantes com Python**. Novatec, 2015.

9. APROVAÇÃO

Aprovado em reunião do Colegiado realizada em: ____/____/_____

Coordenação do Curso de Graduação: _____

Documento assinado eletronicamente por **Shigueo Nomura**, **Professor(a) do Magistério Superior**, em 17/03/2024, às 00:12, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **5278750** e o código CRC **4B9112A0**.

Referência: Processo nº 23117.003020/2024-14 SEI nº 5278750