Laboratório de Programação Competitiva

Profa. Silvia Brandão

2024.2

Aula de hoje ...

- Apresentação do Plano de Ensino e do Sistema de Avaliação.
- Recapitulando conceitos fundamentais de programação em Python.
- Introdução às estruturas de dados em Python.

Plano de Ensino

Divisão da carga horária do componente:

30h/30h - 100% Presencial/ Prática
24h/24h - 100% Não Presencial/ Teórica

EMENTA

BIBLIOGRAFIA

SISTEMA DE AVALIAÇÃO

Plano de Ensino

EMENTA

- Estudo para o desenvolvimento de habilidades em programação competitiva, enfrentando desafios complexos e desenvolvendo soluções eficientes. Tópicos avançados em algoritmos, manipulação de Strings, Matrizes e Vetores, recursividade, essenciais nas competições de programação. Técnicas avançadas de algoritmos, incluindo algoritmos de busca e otimização, programação dinâmica, algoritmos gulosos. Técnicas de competições em maratonas de programação competitiva.

BIBLIOGRAFIA BÁSICA

- ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi De. Fundamentos da programação de computadores: algoritmos, pascal e C/C++. 2.ed. São Paulo: Pearson, 2007. Disponível em: https://plataforma.bvirtual.com.br/. Acesso em: 01 fev. 2024.
- FORBELLONE, André L. V. Lógica de programação: a construção de algoritmos e estruturas de dados com aplicações em Python 4. ed. 331 p. São Paulo. Pearson Education do Brasil, 2022. Disponível em: https://plataforma.bvirtual.com.br/Acervo/Publicacao/200078. Acesso em: 01 fev. 2024

Plano de Ensino

BIBLIOGRAFIA COMPLEMENTAR

- AHO, Alfred V. (et al). **Compiladores: princípios, técnicas e ferramentas**. 2. ed. São Paulo (SP): Pearson AddisonWesley, c2008. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 fev. 2024. ">
- ARAUJO, Sandro de. **Lógica de Programação e algoritmos**. Curitiba: Contentus, 2020. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 fev. 2024. ">
- GUEDES, S. **Lógica de Programação Algorítmica**. São Paulo: Pearson Education do Brasil, 2014. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 fev. 2024. ">
- GUILHON, André et al. (org.). **Jornada Python**: uma jornada imersiva na aplicabilidade de uma das mais poderosas linguagens de programação do mundo. Rio de Janeiro, RJ: Brasport, 2022. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 fev. 2024. ">
- LIMA, Janssen dos Reis. **Consumindo a API do Zabbix com Python**. 1. ed. Rio de Janeiro: Brasport, 2016. Ebook. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 fev. 2024. ">

Plano de Ensino

RECURSOS

- Plataformas de treinamento para competições de programação com foco em Python (Codeforces, AtCoder, Topcoder, Beecrowd, dentre outras).
- https://www.w3schools.com/python/default.asp
- https://docs.python.org/pt-br/3/tutorial/datastructures.html
- Pycharm

SITES DE INTERESSE

- uHunt :: UVa Hunting. http://uhunt.felix-halim.net/
- Brazilian ICPC Summer School, Universidade Estadual de Campinas. http://maratona.ic.unicamp.br
- Maratona de Programação da Sociedade Brasileira de Computação. http://maratona.ime.usp.br
- The ACM-ICPC International Collegiate Programming Contest. https://icpc.baylor.edu
- Olimpíada Brasileira de Informática https://olimpiada.ic.unicamp.br/
- UVA ONLINE JUDGE. (Juiz online) http://uva.onlinejudge.org
- CODEPIT (Site para organizar competições online) www.codepit.io

Plano de Ensino

SISTEMA DE AVALIAÇÃO

- Momento N1: 25 pontos + (Uniube+)
  - **1ª PROVA**: **T17 e T11** – 25/09, **T12** – 26/09, 10pts;
  - Avaliação contínua/Trabalhos/Beecrowd: 15pts;
- Momento N2: 25 pontos + (Uniube+)
  - **2ª PROVA**: **T17 e T11** – 06/11, **T12** – 07/11, 10pts;
  - Avaliação contínua/Trabalhos/Beecrowd: 15pts;
  - 15ª Maratona de Programação: 23/11 – **pontuação extra**
- Momento N3: 25 pontos + (Uniube+ e Avaliação Institucional)
  - PROJETO PRÁTICO: **T17 e T11** – 04/12, **T12** – 05/12, 20pts;
  - Avaliação contínua/Trabalhos/Beecrowd: 5pts;
- **2ª OPORTUNIDADE dos Momentos N1 e N2:**
  - PROVA: **T17 e T11** – 11/12, **T12** – 12/12, CADA UMA 10pts; (CONTEÚDO DE TODO O SEMESTRE)
- **RECUPERAÇÃO DE NOTAS para atingir 60pts:**
  - VALOR: 20pts de prova (substituirá 1ª e 2ª avaliações); porém o aluno não poderá ter sua nota do Uniube+ zerada ou ter menos de 40pts no total do semestre.

Notas de trabalhos e projetos não são recuperáveis.

Recapitulando conceitos fundamentais de programação em Python

Faça o Quiz:

- https://www.w3schools.com/quiztest/quiztest.asp?qtest=PYTHON

Tirando dúvidas

2	1	-5
3	7	0
6	4	8

Uso de matrizes

```
#usando matrizes (lista de listas)
A = [[2, 1, -5], [3, 7, 0], [6, 4, 8]]

diag_A = 0
soma_A = 0

lin = len(A) # Qtde de elementos da lista
col = len(A[0]) # Qtde de elementos do 1º elemento da lista aninhada

for i in range(lin):
  for j in range(col):
  # Elemento da diagonal principal
    if i == j:
      diag_A += A[i][j]

    soma_A += A[i][j]

print(f"diag(A) = {diag_A}")
print(f"soma(A) = {soma_A}")
```

```
diag(A) = 17
soma(A) = 26
```

Tirando dúvidas

Assuntos bastante interessantes:

- o conceito de pilha
- a recursividade de funções.

```python
def fatorial(n):
  if n == 0 or n == 1:
    return 1
  else:
    return n * fatorial(n - 1)

for valor in [0,1,2,3,4,5,6]:
  print(fatorial(valor))
```

```
1
1
2
6
24
120
720
```

```python
def fatorial(n):
  if n == 0 or n == 1:
    return 1
  else:
    return n * fatorial(n - 1)

for valor in [0,1,2,3,4,5,6]:
  print(fatorial(valor))
```

```
1
1
2
6
24
120
720
```

Tabela 4 – Demonstração de uso da pilha com o armazenamento da chamada de função recursiva.

CÓDIGO	PILHA	COMENTÁRIO
fatorial(3)	fatorial(3)	A chamada da função fatorial com parâmetro 3 é colocada na pilha
return 3 * fatorial(2)	fatorial(2) fatorial(3)	O código de fatorial(3) está na pilha, porém, deve esperar a resposta de fatorial(2)
return 2 * fatorial(1)	fatorial(1) fatorial(2) fatorial(3)	Agora, estão na pilha: fatorial(3) e fatorial(2). Só que fatorial(2) exige a solução de fatorial(1) que também é colocada na pilha

return 1	fatorial(1) #Retorna 1
	fatorial(2)
	fatorial(3)
		No caso de fatorial 1, n é 1, então ele retornará 1. Assim, fatorial(1) deve sair da pilha.

return 2 * 1	fatorial(2) #Retorna 2
	fatorial(3)
		Como fatorial(1) foi resolvido, agora fatorial(2) pode ser resolvida também, utilizando o retorno de fatorial(1). Nesse caso, a resposta de fatorial(2) é 2.

return 3 * 2	fatorial(3) #Retorna 6	Por fim, fatorial(3) que aguardava na pilha, recebe a resposta de fatorial(2) e consegue calcular a resposta final, que no caso é 6.