Relatório Trabalho Algoritmos e Complexidade:

- O código a seguir desenvolvido tem como objetivo armazenar e manipular os tempos de corrida de 75 atletas, nome, idade e tempos dos atletas utilizando diferentes estruturas de dados em python.

```python
import random

tempos_treinos = [[random.uniform(10, 15) for _ in range(3)] for _ in range(75)]

print("Lista de tempos dos atletas:")
for i, tempos in enumerate(tempos_treinos, 1):
    print(f"Atleta {i}: {tempos}")

dados_atletas = []
nomes = [f"Atleta_{i+1}" for i in range(75)]
idades = [random.randint(18, 35) for _ in range(75)]

for i in range(75):
    atleta = {
        "nome": nomes[i],
        "idade": idades[i],
        "tempos": tempos_treinos[i],
        "melhor_tempo": min(tempos_treinos[i]),
        "media_tempo": sum(tempos_treinos[i]) / len(tempos_treinos[i]),
    }
    dados_atletas.append(atleta)

print("\nDetalhes dos 75 atletas:")
for atleta in dados_atletas:
    print(f"Nome: {atleta['nome']}, Idade: {atleta['idade']}")
    print(f"Tempos: {atleta['tempos']}")
    print(f"Melhor Tempo: {atleta['melhor_tempo']:.2f} segundos")
    print(f"Média de Tempo: {atleta['media_tempo']:.2f} segundos\n")

```

import Random

- Importa o módulo random para gerar valores aleatórios.

tempos_treinos = [[random.uniform(10, 15) for _ in range(3)] for _ in range(75)] # 3 tempos por atleta

- Cria uma lista bidimensional chamada tempos_treinos, onde cada atleta tem 3 tempos de corrida gerados aleatoriamente entre 10 e 15 segundos.
- A lista tem 75 elementos (um para cada atleta), e cada elemento é uma lista de 3 tempos.

print("Lista de tempos dos atletas:")

for i, tempos in enumerate(tempos_treinos, 1):

print(f"Atleta {i}: {tempos}")

- Exibe os tempos de cada atleta no console.
- Usa enumerate(tempos_treinos, 1) para numerar os atletas a partir de 1.
- Cada tempo é impresso na tela dentro de uma f-string.

dados_atletas = []

nomes = [f"Atleta_{i+1}" for i in range(75)]

idades = [random.randint(18, 35) for _ in range(75)]

- Inicializa dados_atletas, uma lista vazia para armazenar os dados individuais de cada atleta.
- Cria uma lista nomes com os nomes dos atletas no formato Atleta_1, Atleta_2, etc.
- Gera uma lista idades com idades aleatórias entre 18 e 35 anos.

for i in range(75):

atleta = {

"nome": nomes[i],

"idade": idades[i],

"tempos": tempos_treinos[i],

"melhor_tempo": min(tempos_treinos[i]),

"media_tempo": sum(tempos_treinos[i]) / len(tempos_treinos[i]),

}

dados_atletas.append(atleta)

- Cria um dicionário atleta contendo:
- Nome do atleta.
- Idade do atleta.
- Lista de tempos.
- Melhor tempo (é calculado usando min() na lista de tempos).
- Média dos tempos (é calculada somando os tempos e dividindo pelo número de tempos).

print("\nDetalhes dos 75 atletas:")

for atleta in dados_atletas: # Exibe todos os 75 atletas

print(f"Nome: {atleta['nome']}, Idade: {atleta['idade']}")

print(f"Tempos: {atleta['tempos']}")

print(f"Melhor Tempo: {atleta['melhor_tempo']:.2f} segundos")

print(f"Média de Tempo: {atleta['media_tempo']:.2f} segundos\n")

- Exibe os detalhes de cada atleta.
- Usa f-strings para formatar a saída.
- Exibe a lista de tempos, o melhor tempo (formatado para 2 casas decimais) e a média dos tempos.

Comparação de vetores e Listas/Tuplas

Vetores

- **Vantagens:**
  - Estrutura Simples e fácil de manipular.
  - Boa escolha para armazenar conjuntos homogêneos de dados (como os tempos da corrida).

- **Desvantagens:**
  - Não permite armazenar diferentes tipos de dados de forma organizada (por exemplo, nome, idade e tempos do atleta).
  - Dificuldade na recuperação de dados.

Dicionários/Tuplas

- **Vantagens:**
  - Estrutura flexível para armazenar vários tipos de dados associados a um único elemento (como nome, idade e tempos de um atleta).
  - Melhor organização quando há múltiplos atributos associados a cada item.

- **Desvantagens:**
  - Consome mais memória que uma lista simples.
  - Pode ser um pouco mais complexo para iniciantes em comparação aos vetores.

1- **Qual estrutura foi mais fácil de manipular para armazenar e recuperar dados?**
   - O vetor foi mais fácil de usar para armazenar apenas os tempos dos atletas. No entanto, a estrutura de Tuplas facilitou a recuperação de dados associados a cada atleta, como nome, idade e média dos tempos.

2- **Quando seria mais vantajoso usar listas simples em vez de Tuplas?**
   - Quando os dados são homogêneos, como apenas os tempos de corrida dos atletas não há necessidade de rótulos ou acesso baseado em chave.

3- **Como essas estruturas podem ser utilizadas no mundo real?**
   - **Vetores**: Podem ser usadas para armazenar séries de dados numéricos, como preços de ações, temperaturas diárias ou registros de sensores.
   - **Dicionários/Tuplas:** São úteis para representar informações estruturadas, como perfis de usuários em um sistema, registros médicos ou inventários de produtos e vantajosas para armazenar conjuntos de valores imutáveis, como regiões geográficas ou configurações fixas.