UNIVERSIDADE DE FORTALEZA

CURSO DE ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

DISCIPLINA: N704-PROGRAMAÇÃO FUNCIONAL

RELATÓRIO FINAL

INTEGRANTES DA EQUIPE:

VICTOR HUGO FONSECA CAMPOS - 2317221

FORTALEZA

MARÇO/2025

1. Introdução

Este relatório apresenta a implementação de um gerenciador de tarefas utilizando conceitos de programação funcional.

A proposta tem como foco a aplicação prática de closures, funções de alta ordem, funções lambda e list comprehensions.

O sistema permite o gerenciamento básico de tarefas em memória, simulando funcionalidades de cadastro, listagem, filtragem e ordenação.

2. Requisitos Funcionais

Os seguintes requisitos funcionais foram definidos e implementados:

- Para adicionar tarefas com descrição e prioridade está implementado na função: add_task()
- Para listar todas as tarefas existentes, na função: list_tasks()
- Para marcar tarefas como concluídas, função: complete_task()
- Permitir remover tarefas existentes, está na função: remove_task()
- Permitir ordenar as tarefas por prioridade, implementado na função: sort_tasks()
- Para filtrar tarefas por status (pendente ou concluída), está na função: filter_tasks()

3. Requisitos Não Funcionais

Os seguintes requisitos não funcionais foram observados:

- Para executar no terminal via Python 3.x, está contemplado em: execução direta via terminal com if __name__ == "__main__"
- O código deve utilizar conceitos de programação funcional: aplicados conforme descrito na seção 4 abaixo.
- O sistema deve ter testes automatizados utilizando unittest: conforme arquivo de testes separado com uso da biblioteca unittest.
- O projeto deve ser versionado e armazenado em repositório público (GitHub): o link repositório está no item 7 abaixo.

4. Uso de Programação Funcional

- A estrutura de *closure* está na função `task_manager()`, que encapsula o estado da lista de tarefas e retorna as funções de manipulação.

- O uso de função lambda ocorre dentro da função `filter_tasks()`, para filtrar tarefas por status.
- O uso de list comprehension ocorre dentro da função `list_tasks()`, para gerar a lista formatada de tarefas.
- O uso de função de alta ordem ocorre na função `sort_tasks()`, que utiliza a função `sorted()` com lambda para ordenar tarefas.

5. Conclusão

A atividade permitiu explorar conceitos fundamentais da programação funcional de maneira prática.

O uso de *closures* e funções puras possibilitou a criação de um sistema limpo, organizado e com lógica de estado isolada. A separação clara das funções também facilitou os testes unitários.

6. Uso de Inteligência Artificial

Durante o desenvolvimento, foi utilizado apoio pontual com ferramentas de IA (ChatGPT e DeepSeek) para estruturação do código e ajustes de sintaxe. Todo o conteúdo foi compreendido, adaptado e validado manualmente.

7. Repositório

Link para repositório no GitHub: https://github.com/Victorhfcampos/gerenciador-tarefas-funcional