

Exercício para Nota

Prazo até 11/05/2024



Entregar esta atividade no Teams, **somente o código-fonte (.cpp)**

Lista de tarefas

Uma lista de tarefas é um registro organizado de atividades ou afazeres que uma pessoa precisa realizar em um determinado período de tempo. Ela é uma ferramenta simples, mas eficaz, que ajuda a planejar, priorizar e acompanhar o progresso de suas responsabilidades diárias, semanais ou de longo prazo.

Você recebeu um sistema que simula uma lista de tarefas. Este sistema liga a tarefa com uso de uma estrutura de dados duplamente ligada.

Os protótipos das funções estão definidas.

As estruturas de dados estão definidas como o modelo apresentado:

```
typedef struct Item {  
    int categoria;  
    int prioridade;  
    string tarefa;  
} TAREFA;
```

```
struct Node {  
    TAREFA tarefa;  
    Node *anterior;  
    Node *proximo;  
};
```

```
struct Lista {  
    Node *inicio;  
    Node *final;  
    int tamanho;  
};
```

O programa nesta versão possui leaks de memória. Ao terminar os exercícios, estes leaks devem ser sanados.

1) Desenvolver a função Retirar Tarefa por Posição 2. (**void retirar_tarefa_por_posicao2(Lista *, int);**). Esta função recebe o ponteiro da estrutura Lista e a posição da lista, e deve remover a posição da lista na posição passada como argumento. Caso o índice argumento seja maior que o número de elementos na lista, apresentar a mensagem “Tamanho da lista menor que a posição informada” . **(2 pontos)**

2) Desenvolver a função Buscar tarefa por categoria. (**void buscar_tarefas_por_categoria(Lista *, int);**). Esta função recebe como argumento o ponteiro da estrutura Fila e um inteiro, e deve apresentar as tarefas que estejam na categoria informada como argumento. **(2 pontos)**

3) Desenvolver a função Buscar tarefa por prioridades. (**void buscar_tarefas_por_prioridades(Lista *, int);**). Esta função recebe como argumento o ponteiro da estrutura Fila e um inteiro , e deve apresentar as tarefas que possuírem a prioridade informada como argumento. **(2 pontos)**

4) **[Corrigir um BUG]** Ao descomentar a linha 60, compilar e rodar, ocorre uma falha de segmentação. Ajuste a função retirar_tarefa_por_posicao (**void retirar_tarefa_por_posicao(Lista *, int);**) para que o erro não ocorra mais. **(2 pontos)**

5)Desenvolver a função Encerrar. (**void encerrar(Lista *)**;). Esta função recebe como argumento o ponteiro da estrutura Lista, e deve desalocar todas as memórias em uso antes do término da aplicação. **(2 pontos)**

Saída esperada ao descomentar as linhas da função main

Tamanho da lista: 0

[]

Tamanho da lista menor que a posição informada

Tamanho da lista: 25

[Tarefa 1][Tarefa 2][Tarefa 3][Tarefa 4][Tarefa 5][Tarefa 6][Tarefa 7][Tarefa 8][Tarefa 9][Tarefa 10][Tarefa 11][Tarefa 12][Tarefa 13][Tarefa 14][Tarefa 15][Tarefa 16][Tarefa 17][Tarefa 18][Tarefa 19][Tarefa 20][Tarefa 21][Tarefa 22][Tarefa 23][Tarefa 24][Tarefa 25]

Tarefa: Tarefa 5

Categoria: 1

Prioridade: 1

Tarefa: Tarefa 1

Categoria: 1

Prioridade: 1

Tarefa: Tarefa 10

Categoria: 1

Prioridade: 1

Tarefa: Tarefa 11

Categoria: 1

Prioridade: 2

Tamanho da lista: 21

[Tarefa 2][Tarefa 3][Tarefa 4][Tarefa 6][Tarefa 7][Tarefa 8][Tarefa 9][Tarefa 12][Tarefa 13][Tarefa 14][Tarefa 15][Tarefa 16][Tarefa 17][Tarefa 18][Tarefa 19][Tarefa 20][Tarefa 21][Tarefa 22][Tarefa 23][Tarefa 24][Tarefa 25]

Removido: Tarefa 3

Tamanho da lista: 20

[Tarefa 2][Tarefa 4][Tarefa 6][Tarefa 7][Tarefa 8][Tarefa 9][Tarefa 12][Tarefa 13][Tarefa 14][Tarefa 15][Tarefa 16][Tarefa 17][Tarefa 18][Tarefa 19][Tarefa 20][Tarefa 21][Tarefa 22][Tarefa 23][Tarefa 24][Tarefa 25]

Tarefas na categoria 2

[Tarefa 16][Tarefa 17][Tarefa 18][Tarefa 19][Tarefa 20][Tarefa 21][Tarefa 22][Tarefa 23][Tarefa 24][Tarefa 25]

Tarefas com prioridade 2

[Tarefa 12][Tarefa 13][Tarefa 14][Tarefa 15][Tarefa 16][Tarefa 17][Tarefa 18][Tarefa 19][Tarefa 20][Tarefa 21][Tarefa 22][Tarefa 23][Tarefa 24][Tarefa 25]