

Computação Científica I

Guia de Apoio ao Trabalho Prático

Ano lectivo 2023-2024

Turmas: EQM2_M1 e EQM2_M2

Guia #02 – Implementação das estruturas de dados para armazenamento das listas de tarefas e implementação das funções adicionar e listar tarefas

Objectivos

- Implementar estruturas de dados homogêneas para armazenamento de listas de tarefas
- Implementar a funcionalidade que permita adicionar tarefa numa determinada lista
- Implementar a funcionalidade que permita apresentar as tarefas de uma determinada lista

Sumário

Objectivos	1
Pré-requisitos.....	1
Tarefa #01 – Definição das estruturas de dados	3
Tarefa #02 – Implementação da função adicionar_tarefa	4
Tarefa #03 – Implementação da função listar_tarefas.....	7

Pré-requisitos

- Foram feitas as tarefas do guia anterior
- O grupo obteve as funções auxiliares desenvolvidas pelo docente da disciplina
- O grupo controlar as versões e cada membro tem sua cópia do trabalho. Antes de iniciar as actualizações do código, deve garantir que a versão funcional não será afectada. **Dica:** use o

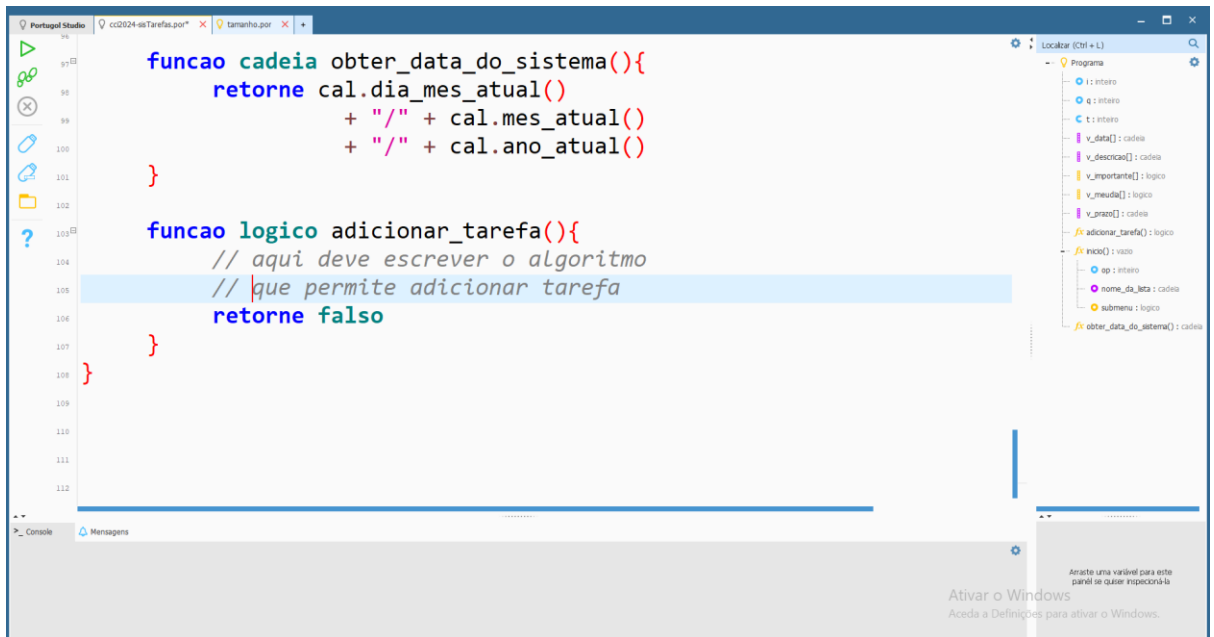
Git/GitHub ou copia para outros locais o código funcional afim de garantir backups que salvaguardem a versão funcional do trabalho.

Tarefa #01 – Definição das estruturas de dados

1. Comece por definir as seguintes estruturas de dados globais (definidas dentro do **programa** e antes da função **inicio**):
 - a. Constante **t** que representa quantidade de tarefas que o sistema pode armazenar (que deve ser inicializada a cem, porque o software que estamos a desenvolver só pode armazenar, no máximo, 100 tarefas);
 - b. Variável **q** que representa a quantidade de tarefas adicionadas pelo utilizador (que deve ser inicializada a zero, porque ainda não existe nenhuma tarefa adicionada);
 - c. Vector **v_descricao** de tamanho **t** armazenará a descrição de uma tarefa, por exemplo “Estudar CCI de noite.”. É preciso escolher o tipo de dado adequado para esta variável/vector.
 - d. Vector **v_meudia** de tamanho **t** indica se a tarefa é para ser realizada imediatamente (nesse caso armazena o valor lógico **verdadeiro**), e portanto deve aparecer na lista “O meu dia”, ou pode ser feita em outro momento (nesse caso armazena o valor lógico **falso**) e, portanto, não deve aparecer na lista “O meu dia”. É preciso escolher o tipo de dado adequado para esta variável/vector.
 - e. Vector **v_importante** de tamanho **t** armazenará a importância de uma tarefa, isto é, se é importante (armazena o valor lógico **verdadeiro**) ou não é importante (armazena o valor lógico **falso**). É preciso escolher o tipo de dado adequado para esta variável/vector.
 - f. Vector **v_prazo** de tamanho **t** armazenará a data correspondente ao prazo da tarefa, por exemplo “20/05/2024”. É preciso escolher o tipo de dado adequado para esta variável/vector.
 - g. Vector **v_data** de tamanho **t** armazenará a data da criação da tarefa, por exemplo “13/05/2024”. É preciso escolher o tipo de dado adequado para esta variável/vector. Para obter a data do sistema deve utilizar a função dada pelo docente da disciplina.
2. Agora defina as variáveis auxiliares globais (definidas dentro do **programa** e antes da função **inicio**):
 - a. Variáveis **i, j** serão usadas para iterações nas estruturas de repetição
 - b. Variável **li** para leitura de inteiro
 - c. Variável **buf** para leitura de cadeia de caracteres
 - d. Variável **id_lista** para identificar as tarefas a apresentar para cada lista seleccionada
3. Adicione os comentários as variáveis como boas práticas de codificação.

Tarefa #02 – Implementação da função adicionar_tarefa

1. Para melhorar estruturação do código-fonte do software no sentido de facilitar a manutenção e actualização para novas versões, vamos modularizar o programa em subrotinas (funções e procedimentos).
2. Comece por definir a função `adicionar_tarefa`:
Essa função vai permitir adicionar uma tarefa em uma determinada lista previamente seleccionada. A função não recebe parâmetro. Retorna `verdadeiro` se conseguir adicionar a tarefa ou retorna `falso`, caso contrário.



Repare que a função `obter_data_do_sistema`, já foi implementado pelo docente como pode ser visto na imagem acima. Essa função retorna a data, no formato `dd/mm/aaaa`, como uma cadeia de caracteres. Tivemos que incluir a biblioteca `Calendario` para usarmos as funções `dia_mes_atual`, `mes_atual` e `ano_atual`... `cal` é um pseudónimo criado para simplificar a escrita. Verifique também que concatenamos os valores e retornamos como uma cadeia.

3. Verifique se a quantidade de elementos é menor que a `t-1`, se for verdade deve adicionar a tarefa e retorna o valor lógico `verdadeiro`, senão for então deve apresentar uma mensagem a informar que a memória está cheia e retorna o valor lógico `falso`.

Algoritmo para adicionar a tarefa

- i. Pede a descrição da tarefa e armazena no vector `v_descricao` na posição `q`
- ii. Pergunta se a tarefa é ou não importante e armazena no vector `v_importante` na posição `q`

Sugestão de estratégia de implementação para leitura do valor lógico:

```

114
115 faca
116 {
117     escreva("\nÉ importante (1-Sim ou 2-Não): ")
118     leia(li)
119     escolha(li)
120     {
121         caso 1:
122             v_importante[q] = verdadeiro
123             pare
124         caso 2:
125             v_importante[q] = falso
126             pare
127         caso contrario:
128             li = 3
129             escreva("Opção inválida!!! Por favor, seleccione uma opção correcta.\n")
130     }
131 } enquanto (li == 3)

```

Note que o vector `v_importante` armazena valores lógicos e não o texto “importante” e “não importante”. É responsabilidade do programador manipular os dados e apresentar as mensagens mais perceptíveis para o utilizador, enquanto o código é implementado de forma adequada.

- iii. Pede a data para o prazo da tarefa e armazena no vector `v_prazo` na posição `q`
- iv. Usa a função `obter_data_do_sistema()` para obter a data do sistema e armazena no vector `v_data` na posição `q`

Sugestão de estratégia para obter a data do sistema:

```

// armazena a data da criação da tarefa
v_data[q] = obter_data_do_sistema()

```

- v. Se a lista na qual está a ser adicionada a tarefa for a Lista “O meu dia”, então
 - i. Armazena no vector `v_meudia` na posição `q` o valor lógico `verdadeiro`
- Sugestão de estratégia para implementação:** inclua a biblioteca `Texto` e use a função `posicao_texto` para verificar se no nome do menu existe a frase “O meu dia”. Se existir, então retornará a posição correspondente ao início da ocorrência, que no nosso caso será superior a zero. Caso não encontre, então retorna o valor -1. Logo, se a condição for verdadeira, estaremos no contexto da lista “O meu dia”. Assim sendo, podemos marcar essa tarefa como sendo uma tarefa a ser realizada hoje.

```

// verifica se está no contexto da lista o meu dia
se(tx.posicao_texto("O meu dia", nome_da_lista, 0) > 0)
    v_meudia[q] = verdadeiro

```

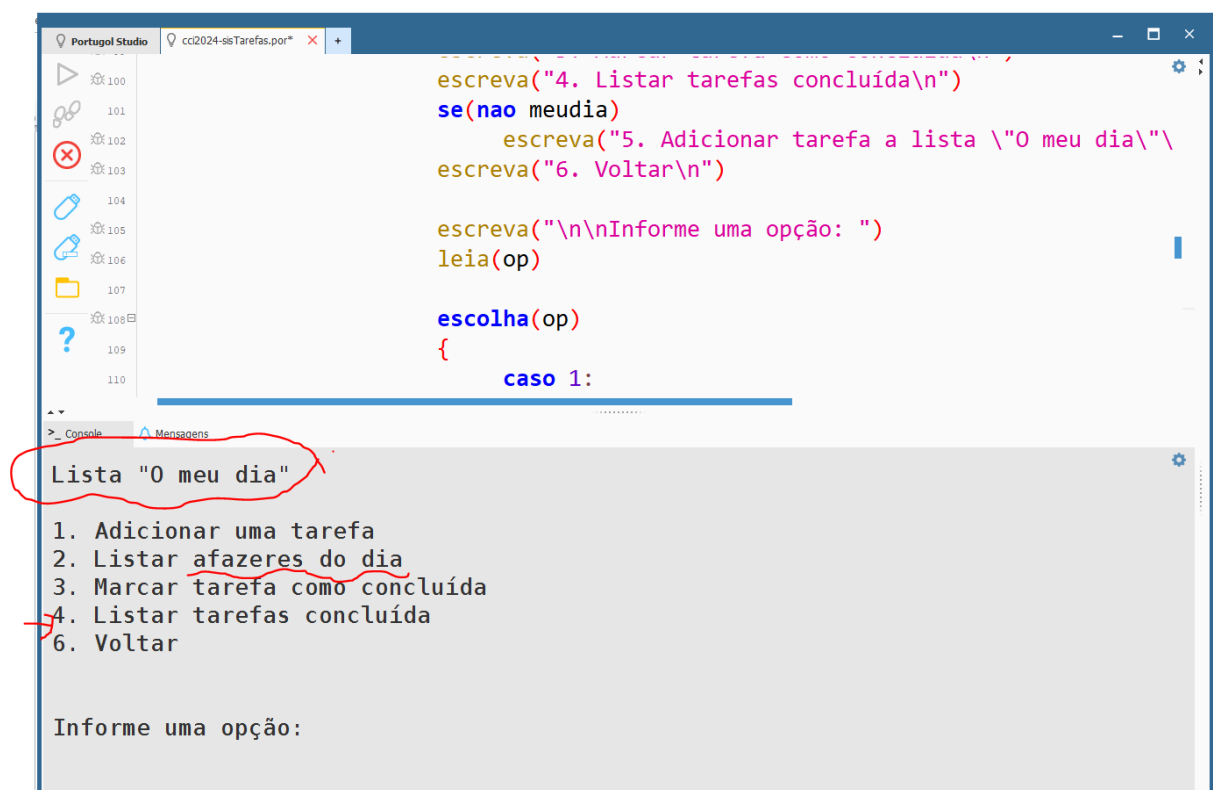
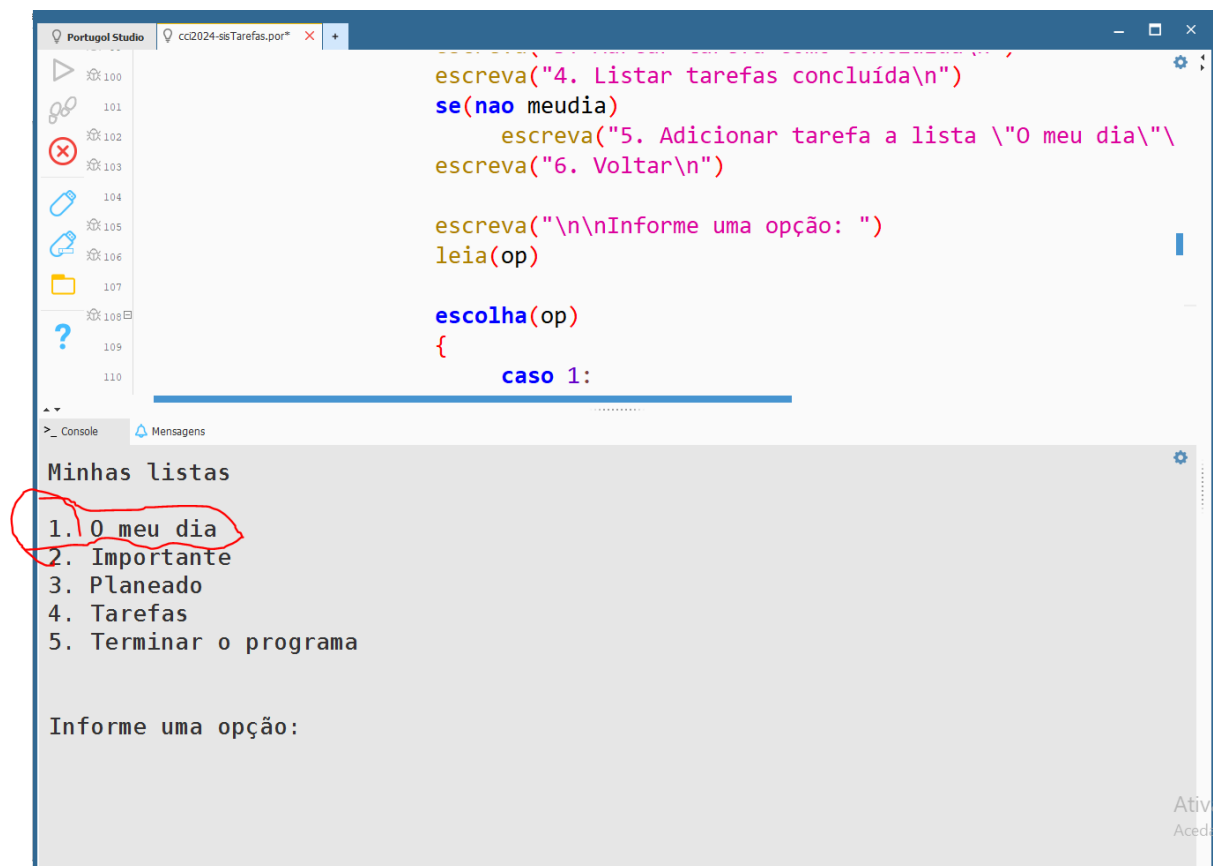
Verifica que precisamos do nome da lista para determinarmos o contexto, assim foi necessário alterar a definição da função para receber o nome da lista como parâmetro (veja a imagem abaixo):

```
110  /**
111  *
112  */
113  funcao cadeia obter_data_do_sistema(){
114      retorne cal.dia_mes_atual()
115              + "/" + cal.mes_atual()
116              + "/" + cal.ano_atual()
117  }
118
119  funcao logico adicionar_tarefa(cadeia nome_da_lista){
120      se(q < t - 1)
121      {
122          escreva ("\n\nAdicionar uma nova tarefa\n")
123
124          escreva("\nDescrição: ")
125          leia(v_descricao[q])
126
127          faca
```

- vi. Incrementar a variável q
 - vii. Retornar o valor verdadeiro
4. Trate a questão da indentação do código e adicione os comentários necessários para fácil manutenção.

Tarefa #03 – Implementação da função listar_tarefas

1. Comece por actualizar as opções do submenu, conforme o enunciado actualizado:



```

100 escreva("4. Listar tarefas concluída\n")
101 se(nao meudia)
102     escreva("5. Adicionar tarefa a lista \"0 meu dia\"\n")
103     escreva("6. Voltar\n")
104
105 escreva("\n\nInforme uma opção: ")
106 leia(op)
107
108 escolha(op)
109 {
110     caso 1:

```

Lista "Planeado"

1. Adicionar uma tarefa
2. Listar afazeres "Planeado"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "0 meu dia"
6. Voltar

Informe uma opção:

2. Agora chame a função adicionar_tarefa no caso correspondente do submenu.

```

100 escreva("4. Listar tarefas concluída\n")
101 se(nao meudia)
102     escreva("5. Adicionar tarefa a lista \"0 meu dia\"\n")
103     escreva("6. Voltar\n")
104
105 escreva("\n\nInforme uma opção: ")
106 leia(op)
107
108 escolha(op)
109 {
110     caso 1:
111         adicionar_tarefa(nome_da_lista)
112         pare
113     caso 2:
114         para(i=0; i < q; i++)
115             se(v_meudia[i])
116                 escreva(i+1, ". ", v_descricao[i], "\n")
117             senao
118                 escreva(i+1, ". ", v_descricao[i], "\n")
119             leia(buf)
120 }
121 quanto (op != 6)

```

3. Agora vamos criar a função que permite imprimir as tarefas de uma determinada lista:


```

funcao vaziao listar_tarefas(inteiro tipo) {

    escolha(tipo)
    {
        caso 1: // meu dia
        caso 2: // importante
        caso 3: // planeado
        caso 4: // tarefas

    }
}

```

Repare que, para evitar a criação de várias funções de impressão, criamos a função `listar_tarefas` e recebe um parâmetro (`tipo`) que determina a lista que queremos imprimir. Assim simplifica o código da seguinte forma:

- i. Definiremos uma variável `id_lista`, que assumirá valores iguais a cada caso de acordo a lista selecionada.

```

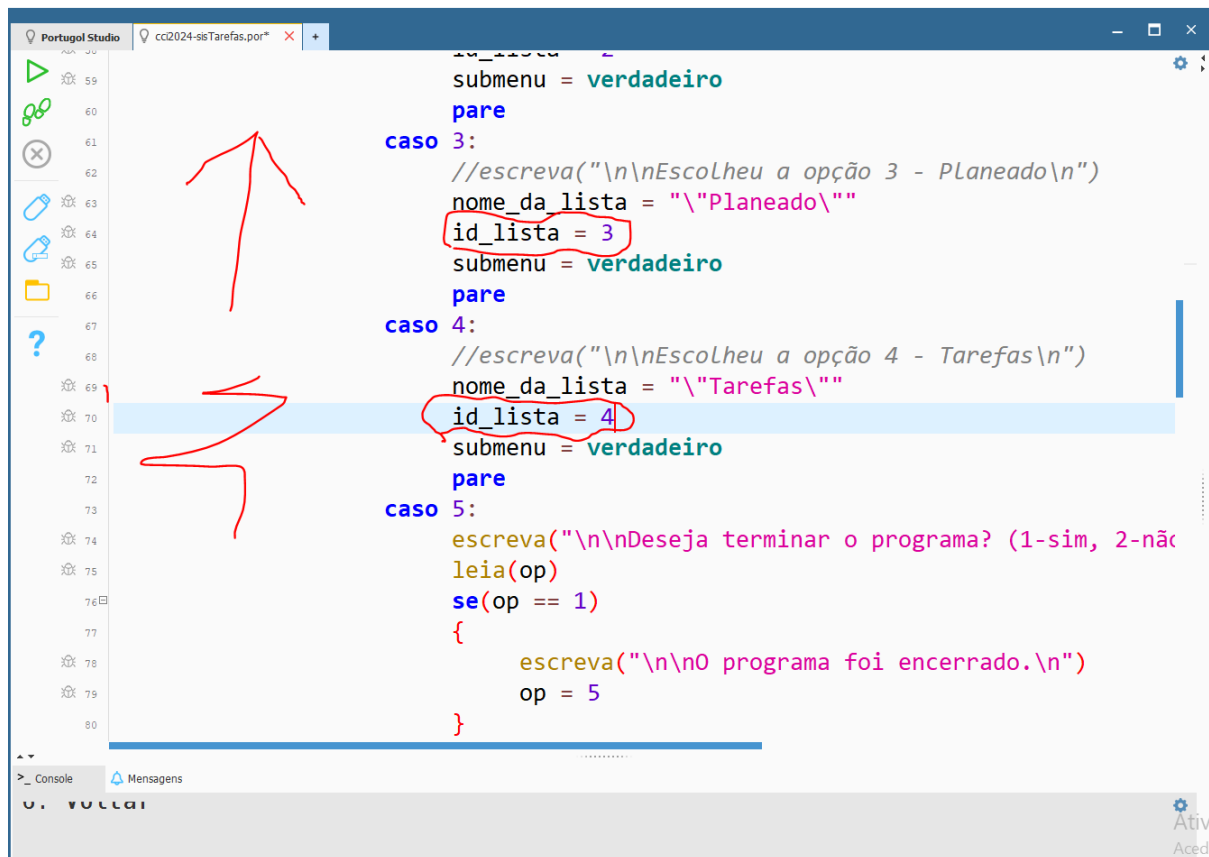
// ...
cadeia v_prazo[t] // prazo de realização da tarefa
cadeia v_data[t] // data de criação da tarefa

// variaveis auxiliares globais
inteiro i, li
cadeia buf

funcao inicio()
{
    inteiro op
    cadeia nome_da_lista = ""
    inteiro id_lista
    logico submenu = falso
    // menu principal
    faca
    {
        limpa()
        escreva("Minhas listas\n\n")
        escreva("1. O meu dia\n")
        escreva("2. Importante\n")
        escreva("3. Planeado\n")
        escreva("4. Tarefas\n")
    }
}

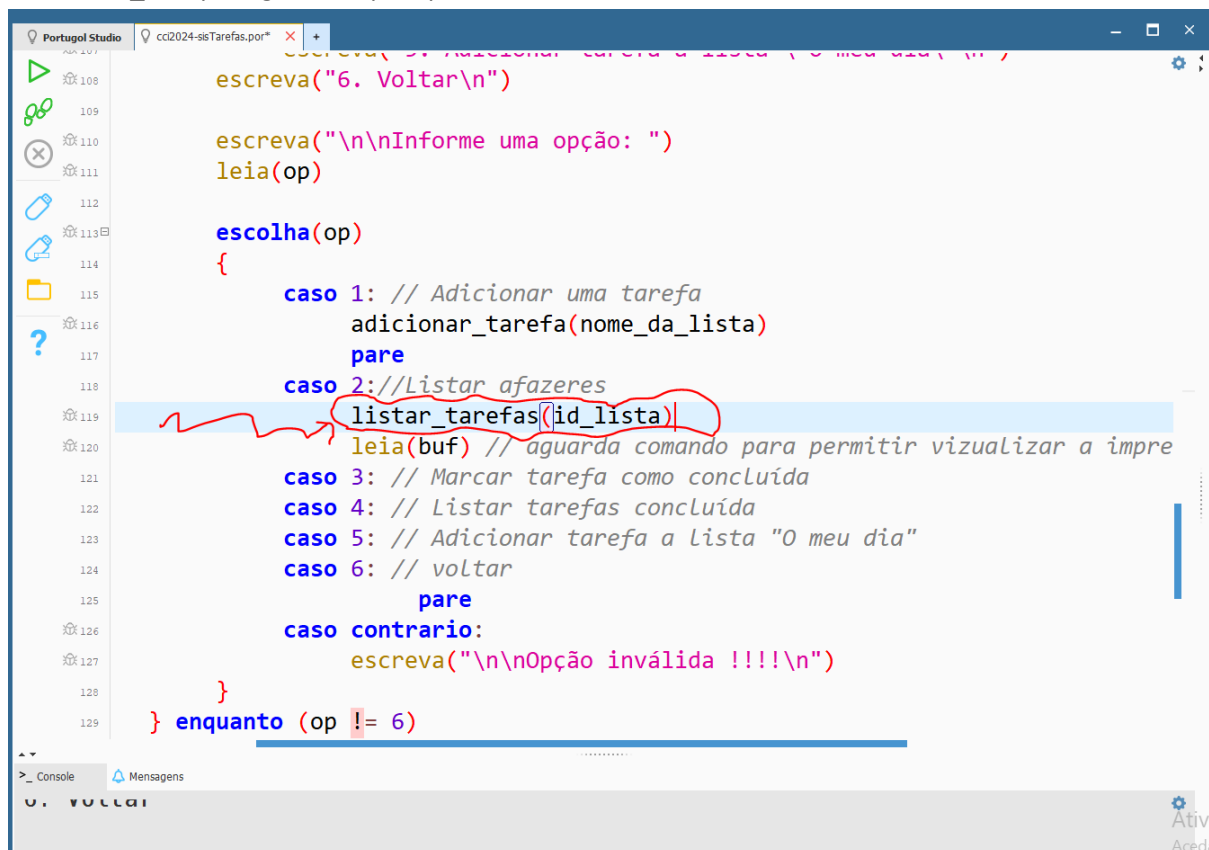
```

- ii. Agora só precisamos passar os valores apropriados de acordo ao tipo de lista:



```
59 submenu = verdadeiro
60 pare
61 caso 3:
62     //escreva("\n\nEscolheu a opção 3 - Planeado\n")
63     nome_da_lista = "\"Planeado\""
64     id_lista = 3
65     submenu = verdadeiro
66     pare
67 caso 4:
68     //escreva("\n\nEscolheu a opção 4 - Tarefas\n")
69     nome_da_lista = "\"Tarefas\""
70     id_lista = 4
71     submenu = verdadeiro
72     pare
73 caso 5:
74     escreva("\n\nDeseja terminar o programa? (1-sim, 2-não)
75     leia(op)
76     se(op == 1)
77     {
78         escreva("\n\nO programa foi encerrado.\n")
79         op = 5
80     }
```

- iii. Agora precisamos chamar a função `listar_tarefas` passando como parâmetro a variável `id_lista` para garantir que apresentemos as tarefas da lista dada:



```
108 escreva("6. Voltar\n")
109 escreva("\n\nInforme uma opção: ")
110 leia(op)
111 escolha(op)
112 {
113     caso 1: // Adicionar uma tarefa
114         adicionar_tarefa(nome_da_lista)
115     pare
116     caso 2: // Listar afazeres
117         listar_tarefas(id_lista)
118     leia(buf) // aguarda comando para permitir visualizar a impre
119     caso 3: // Marcar tarefa como concluída
120     caso 4: // Listar tarefas concluída
121     caso 5: // Adicionar tarefa a lista "O meu dia"
122     caso 6: // voltar
123         pare
124     caso contrario:
125         escreva("\n\nOpção inválida !!!!\n")
126 }
127 } enquanto (op != 6)
```

- iv. Lembre-se de inicializar com a variável `id_lista`.

- v. Adicione as funções auxiliares para facilitar a apresentação:

```
funcao cadeia obter_estado_importancia(logico estado)
{
    se (estado) retorne "importante"
    retorne "não é importante"
}

funcao cadeia obter_prazo(cadeia prazo)
{
    se(tx.numero_caracteres(prazo) == 0) retorne ""
    retorne "Prazo: " + prazo
}

funcao vazio listar_tarefas(inteiro tipo) {
    escolha(tipo)
    {
        caso 1: // meu dia
        {
            para(i=0; i < q; i++)
            {
                se(v_meudia[i]) {
                    escreva(i+1, ". ",
                        v_descricao[i],
                        ", [" ,obter_estado_importancia(v_importante[i]),"]",
                        ", ", obter_prazo(v_prazo[i]),
                        ", criada em: ", v_data[i], "\n")
                }
            }
            pare
        }
        caso 2: // importante
    }
```

4. Adicione os comentários necessários, faça indentação do código e teste o programa para verificar a correcção e em caso de erros realizar o debugging do mesmo.
5. É esperado o seguinte outputs:

```
>_ Console Mensagens

Minhas listas
1. 0 meu dia
2. Importante
3. Planeado
4. Tarefas
5. Terminar o programa

Informe uma opção:
```

```
>_ Console Mensagens
Lista "0 meu dia"
1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção:
```

```
>_ Console Mensagens
Lista "0 meu dia"
1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção: 1

Adicionar uma nova tarefa

Descrição: Estudar CCI hoje antes de dormir.

É importante (1-Sim ou 2-Não): 1

N.B.: Informe uma data no formato dd/mm/aaaa;
ou não insere a data, basta pressionar a tecla ENTER)
Prazo:
```

Lista "0 meu dia"

1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção: 1

Adicionar uma nova tarefa

Descrição: Estudar CCI hoje antes de dormir.

É importante (1-Sim ou 2-Não): 1

N.B.: Informe uma data no formato dd/mm/aaaa;
ou não insere a data, basta pressionar a tecla ENTER)
Prazo: 12/05/2024

Lista "0 meu dia"

1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção: 2

1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024

```
>_ Console Mensagens

Lista "0 meu dia"

1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção: 6
```

```
>_ Console Mensagens

Minhas listas

1. 0 meu dia
2. Importante
3. Planeado
4. Tarefas
5. Terminar o programa

Informe uma opção: 2
```

```
>_ Console Mensagens

Lista "Importante"

1. Adicionar uma tarefa
2. Listar afazeres "Importante"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "0 meu dia"
6. Voltar

Informe uma opção: 2
```

```
>_ Console Mensagens
Lista "Importante"
1. Adicionar uma tarefa
2. Listar afazeres "Importante"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 2
1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024

>_ Console Mensagens
Lista "Importante"
1. Adicionar uma tarefa
2. Listar afazeres "Importante"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 6
```

Repare que a tarefa adicionada por ser criada na lista "O meu dia" foi marcada como tarefa a ser realizada hoje, e portanto deve aparecer em referida lista, por ser marcada como importante, deve aparecer na lista "Importante" e por possuir um prazo, deve aparecer na lista "Planeado" e, obviamente, por ser uma tarefa, deve aparecer na lista "Tarefas".

Agora vamos adicionar uma tarefa na lista "Tarefas", que não é importante e não tem prazo de conclusão, que portanto só deve aparecer na lista "Tarefas" e não nas listas "Importante" e "Planeado".

```
>_ Console Mensagens

Minhas listas

1. 0 meu dia
2. Importante
3. Planeado
4. Tarefas
5. Terminar o programa

Informe uma opção: 4

Lista "Tarefas"

1. Adicionar uma tarefa
2. Listar afazeres "Tarefas"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "0 meu dia"
6. Voltar

Informe uma opção:
```



```
>_ Console  Mensagens

1. Adicionar uma tarefa
2. Listar afazeres "Tarefas"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 1

Adicionar uma nova tarefa

Descrição: Jogar bola com os amigos
É importante (1-Sim ou 2-Não): 2

N.B.: Informe uma data no formato dd/mm/aaaa;
ou não insere a data, basta pressionar a tecla ENTER)
Prazo:
```

```
>_ Console  Mensagens

Lista "Tarefas"

1. Adicionar uma tarefa
2. Listar afazeres "Tarefas"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 2
1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024
2. Jogar bola com os amigos, [não é importante], , criada em: 12/5/2024
```

Repare que aparece, tanto a tarefa adicionada na lista "O meu dia", quanto a tarefa adicionada na lista "Tarefas". No entanto a segunda tarefa, não é importante e não tem prazo, logo não aparece nas listas "Importante" e "Planeado", respectivamente.

```
>_ Console  Mensagens

Lista "Importante"

1. Adicionar uma tarefa
2. Listar afazeres "Importante"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 2
1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024
```

Vamos adicionar uma tarefa importante, mas sem prazo na lista "Tarefas" para mais uma vez verificarmos o comportamento esperado:

```
>_ Console Mensagens

1. Adicionar uma tarefa
2. Listar afazeres "Tarefas"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 1

Adicionar uma nova tarefa

Descrição: Preparar o exame final de CCI
É importante (1-Sim ou 2-Não): 1

N.B.: Informe uma data no formato dd/mm/aaaa;
ou não insere a data, basta pressionar a tecla ENTER)
Prazo:
```

Agora vamos listar para vermos as tarefas da lista "Tarefas":

```
>_ Console Mensagens

Lista "Tarefas"

1. Adicionar uma tarefa
2. Listar afazeres "Tarefas"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 2
1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024
2. Jogar bola com os amigos, [não é importante], , criada em: 12/5/2024
3. Preparar o exame final de CCI, [importante], , criada em: 12/5/2024
```

Agora vamos listar as tarefas da lista "Importante":

```
>_ Console Mensagens

Lista "Importante"

1. Adicionar uma tarefa
2. Listar afazeres "Importante"
3. Marcar tarefa como concluída
4. Listar tarefas concluída
5. Adicionar tarefa a lista "O meu dia"
6. Voltar

Informe uma opção: 2
1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024
3. Preparar o exame final de CCI, [importante], , criada em: 12/5/2024
```

Como pode ver, só existem duas tarefas importantes, nomeadamente a **tarefa 1** e a **tarefa 3**.

Se listarmos as tarefas do dia, teremos:

Lista "O meu dia"

1. Adicionar uma tarefa
2. Listar afazeres do dia
3. Marcar tarefa como concluída
4. Listar tarefas concluída
6. Voltar

Informe uma opção 2

1. Estudar CCI hoje antes de dormir., [importante], Prazo: 12/05/2024, criada em: 12/5/2024

Só temos a primeira tarefa.

Se concluiu todas as tarefas deste guia, meus parabéns!!!! Tem agora 80% do trabalho concretizado!!!!

Precisa agora realizar a última etapa de construção:

1. Marcar as tarefas como concluída
2. Listar as tarefas concluídas
3. Adicionar tarefa a lista "O meu dia"

Bom trabalho!!!!!!