Universidade Federal do Espírito Santo
Departamento de Informática

Trabalho Prático

TickTrack - Gerenciador de Tickets

Programação II - 2024/2
18 de fevereiro de 2024

Update 01/03/2025: novos casos de testes (e uma pequena correção no Caso 3) foram divulgados no repositório do template do trabalho

Objetivo

O objetivo deste trabalho é colocar em prática as habilidades de programação na linguagem C adquiridas ao longo do curso. O trabalho foi elaborado para você exercitar diversos conceitos, principalmente a construção de TADs opacos e genéricos, alocação e manipulação de memória de maneira dinâmica. Como consta no plano de ensino da disciplina, o trabalho será realizado em duas etapas (a primeira realizada em casa e a segunda em sala de aula), sendo esta a primeira etapa. O somatório de ambas as etapas **equivale a 50% da nota final do curso**.

Introdução

Dentro de grandes organizações, como empresas e universidades, é comum existir um sistema para gerenciamento de tickets (ou chamados) para registrar e acompanhar solicitações de suporte técnico. Esses sistemas permitem que usuários relatem problemas relacionados à infraestrutura em geral, como manutenção predial, falhas em computadores, dificuldades com softwares corporativos, dentre muitos outros. Além disso, esses sistemas ajudam na organização das demandas, garantindo que chamados sejam atendidos de forma organizada. Ao centralizar as solicitações, o sistema também facilita o acompanhamento do tempo de resolução e a distribuição das tarefas entre os técnicos, melhorando a eficiência no atendimento e a satisfação dos usuários.

Sendo assim, neste trabalho, sua missão **é implementar um protótipo de gerenciamento de tickets** (na qual chamaremos de **TickTrack**) cujo objetivo é realizar algumas das tarefas comuns em sistemas do tipo.

Descrição geral do sistema

A intenção principal do TickTrack é gerenciar digitalmente os tickets solicitados em uma organização fictícia. Para isso, deve ser considerado as seguintes premissas:

- O sistema possui dois atores principais: usuário e técnico
- O sistema deve ser capaz de cadastrar ambos os atores
- O usuário abre tickets para solicitar algum serviço
- O técnico recebe os tickets para solucionar o problema solicitado de acordo com a disponibilidade de tempo dos mesmos
- Existem três tipos de tickets disponíveis no sistema: `SOFTWARE`, `MANUTENCAO` e `OUTROS`
- Cada ticket tem uma série de particularidades e para cada um deles é estabelecido um tempo de solução
- Os tickets devem ser distribuídos entre os técnicos respeitando uma fila
- O sistema deve ser capaz de notificar todos os tickets do sistemas
- O sistema deve ser capaz de realizar relatórios em relação aos atores e tickets

Dessa forma, o TickTrack deve ser capaz de fornecer as seguintes funcionalidades:

1. Cadastro de Usuários e Técnicos
2. Abertura de Ticket com inserção em uma fila
3. Distribuição dos tickets para os técnicos
4. Notificação de tickets
5. Relatórios solicitados

Todas as interações com o programa são realizadas utilizando entrada e saída padrão. Em outras palavras, lendo e escrevendo na tela do terminal.

Cadastro de usuários e técnicos

A primeira funcionalidade do TickTrack é o cadastro de atores do sistema. Os atores possuem dados em comum e dados específicos. Em comum, todos eles possuem:

- Nome completo (string com no máximo 100 caracteres)
- CPF (no formato: 000.000.000-00)
- Data de nascimento (seguindo padrão dd/mm/aaaa)
- Telefone (formato: (00)00000-0000)
- Gênero (`MASCULINO`, `FEMININO`, `OUTROS`)

O CPF é o identificador único de um ator. O sistema não deve permitir cadastro de atores do mesmo tipo com um mesmo CPF. Por exemplo, não é permitido o cadastro de dois usuários com o mesmo CPF. Caso seja realizada uma tentativa de cadastro com o mesmo CPF o sistema deve ignorar e não realizar o cadastro.

Além das informações a seguir, um técnico também possui as seguintes informações:

- Área de atuação, que pode assumir os valores `GERAL` e `TI`
- Salário
- Disponibilidade de tempo (valor inteiro de horas que o técnico tem para trabalhar)

No caso de um usuário, a única informação a mais que ele possui é:

- Setor de trabalho, que pode assumir os valores `RH`, `FINANCEIRO`, `P&D`, `VENDAS` ou `MARKETING`.

O cadastro de ambos os atores é realizado via entrada padrão e são representados pelos caracteres `T` e `U`, cadastro do **T**écnico e **U**suário, respectivamente. Um exemplo de cadastro para ambos os atores é ilustrado a seguir:

```
T (caractere indicando cadastro de um técnico)
HEIMERDINGER GARCIA (nome)
456.987.123-96 (CPF)
20/10/1945 (data de nascimento)
(90)99876-1234 (telefone)
MASCULINO (gênero)
GERAL (área de atuação)
20 (disponibilidade de trabalho)
3500 (salário)
U (caractere indicando cadastro de um usuário)
TAHM KENCH (nome)
123.932.250-46 (CPF)
01/09/1955 (data de nascimento)
(34)96373-7004 (telefone)
OUTROS (gênero)
RH (setor)
```

Abertura de tickets

Um ticket pode ser aberto por um usuário, que por sua vez deve informar algumas características de acordo com o ticket. O TickTrack comporta três tipos de tickets: `MANUTENCAO`, `SOFTWARE` e `OUTROS`. As informações necessárias para abrir cada um deles é descrita na sequência.

- **`MANUTENCAO`:** ticket que será solicitado para realizar manutenção de algum equipamento em geral. É necessário informar:
    - Nome do item a ser executado a manutenção (string com no máximo 100 caracteres)
    - O estado do item (`RUIM`, `REGULAR` ou `BOM`)
    - O local exato onde se encontra o item (string com no máximo 100 caracteres)
- **`SOFTWARE`:** ticket que será solicitado para trabalhar em alguma coisa relacionado a Software que a organização utiliza. É necessário informar:
    - Nome do software (string com no máximo 100 caracteres)
    - Categoria da solicitação (`BUG`, `DUVIDA` ou `OUTROS`)
    - O impacto do problema no trabalho do usuário (inteiro com valor 3, 2 ou 1, representando impacto alto, médio e baixo)
    - Motivo da solicitação (string com no máximo 200 caracteres)
- **OUTROS**: ticket que será solicitado para situações em que as demais categorias não cobrem ou não se encaixa perfeitamente. É necessário informar:
    - Descrição (string com no máximo 500 caracteres)
    - Local (string com no máximo 100 caracteres)
    - Nível de dificuldade que o usuário acredita que será necessário para solucionar o problema (valor inteiro iniciando de 1)

Um ticket só pode ser aberto por um usuário cadastrado. E ele deve ser realizado utilizado o caractere A, informando na sequência o CPF do usuário e o tipo do ticket. Caso o CPF não exista dentro do banco de usuários cadastrado, o ticket deve ser ignorado. Um exemplo de abertura de ticket é exibido a seguir:

```
A (caractere indicando a abertura do ticket)
255.942.213-22 (CPF do usuário solicitante)
OUTROS (tipo do ticket)
RECOLHER COBRA QUE APARECEU NA ENTRADA DO PREDIO (descrição)
PASSARELA DE ENTRADA (local)
5 (dificuldade estimada pelo usuário)
A (caractere indicando a abertura do ticket)
123.932.250-46 (CPF do usuário solicitante)
MANUTENCAO (tipo do ticket)
PORTA DE ENTRADA (nome do item)
RUIM (estado de conservação)
SALA 22 BLOCO B (localização)
A (caractere indicando a abertura do ticket)
524.456.852-98 (CPF do usuário solicitante)
SOFTWARE (tipo do ticket)
```

```
EXCEL (nome do software)
DUVIDA (categoria da solicitação)
2 (impacto no trabalho)
NAO CONSIGO ORDENAR COLUNA DE DADOS (motivo da solicitação)
```

Sempre que um ticket é aberto, ele deve ser inserido em uma fila de tickets. Essa fila será utilizada para distribuição entre os técnicos. Além disso, sempre que o ticket é aberto ele assume o status Aberto, que só muda quando ele for Finalizado. Além disso, o sistema deve atribuir a cada ticket um tempo estimado em que um técnico deve trabalhar para resolver o problema. Esse tempo é calculado de acordo com o tipo do ticket e as informações fornecidas pelo usuário. Considerando cada um dos tickets, o tempo é calculado da seguinte forma

- **MANUTENCAO**: o tempo estimado depende do estado de conservação do item e o do setor do usuário que abriu o ticket.
    - Cada estado atribui uma quantidade de tempo: RUIM é 3h, REGULAR é 2h e BOM é 1h
    - O setor do usuário impacta no cálculo. Cada setor tem um fator de multiplicação. RH = 2, FINANCEIRO = 3, P&D = 1, VENDAS = 1, MARKETING = 1
    - O cálculo é realizado multiplicando o tempo do estado pelo fator do setor. Por exemplo, se o estado é RUIM e o setor é o RH, o tempo estimado será atribuído como 3 * 2 = 6h
- **SOFTWARE**: o tempo estimado depende da categoria do problema e o impacto no trabalho do usuário.
    - Cada categoria atribui uma quantidade de tempo: BUG é 3h, OUTROS é 2h e DUVIDA é 1h
    - O tempo é calculado somando o tempo da categoria e o impacto (que é um inteiro de 1 a 3)
    - Por exemplo, se a categoria é BUG e o impacto é 1, o tempo estimado é dado por 3 + 1 = 4h
- **OUTROS**: o tempo estimado é igual ao nível de dificuldade que o usuário acredita ser necessário para solucionar o problema
    - Por exemplo, se o nível de dificuldade é 4, logo o tempo estimado será de 4h.

Por fim, sempre que um ticket for registrado, ele ganha um ID único que segue o padrão `Tick-n`, sendo `n` a posição do ticket na fila. Por exemplo: `Tick-3`.

Distribuição dos tickets

Os tickets abertos que estão na fila, deverão ser distribuídos para os técnicos cadastrados no sistema. Essa distribuição respeita algumas regras, são elas:

- Um ticket só será distribuído se ele estiver com o status Aberto
- Tickets do tipo SOFTWARE só podem ser atribuídos para técnicos de TI
    - Técnicos de TI só recebem este tipo de ticket
- Os demais tipos de ticket podem ser distribuídos para técnicos do tipo GERAL
- Para atribuir um ticket a um técnico, é necessário que o técnico tenha disponibilidade de tempo igual ou maior ao tempo estimado do ticket
    - Se isso acontecer, o ticket é atribuído para o técnico e o tempo estimado é descontado da disponibilidade de tempo do técnico.
    - O ticket passa para o status Finalizado.
- A ordem de distribuição respeita a ordem de chegada na fila de tickets. Porém, a distribuição é realizada seguindo o método *Round Robin* (RR), ou seja, uma fila circular entre os técnicos.
- Imagine uma lista de Tickets LK = {K1, K2, K3, ..., Kn} e uma lista de Técnicos LT = {T1, T2, T3, ..., Tn}
    - O ticket Ki é oferecido para Tj. Se for possível ele assumir este ticket, ou seja, ele satisfaz os requisitos de área e tempo, ele pega esse ticket e o próximo Ticket Ki+1 será oferecido para Tj+1
    - Se não for possível que Tj assuma o ticket Ki, este ticket é oferecido para Tj+1. Isso vai acontecer até encontrar algum técnico capaz de solucionar o problema ou acabar a lista de técnicos
    - Se nenhum técnico for capaz de assumir, o ticket ficará em aberto e será iniciado uma nova rodada para Ki+1
- O algoritmo de distribuição respeita a fila de tickets e a ordem de cadastro dos técnicos

Um exemplo da solicitação de distribuição dos tickets é mostrado a seguir:

```
E (caractere indicando que alguma ação será executada)
DISTRIBUI (ação solicitada)
```

Quando a ação é solicitada, devem ser considerados apenas os técnicos cadastrados e tickets abertos naquele momento.

Outras ações disponíveis no sistema

O sistema deve permitir a execução de outras ações baseadas nos dados cadastrados e nos tickets solicitados pelos usuários. Essas ações são descritas a seguir.

Notificação de tickets

Todos os tickets da lista podem ser notificados. A notificação nada mais é do que imprimir a fila dos tickets na tela. Um exemplo de solicitação de notificação é mostrado a seguir:

```
E (caractere indicando que alguma ação será executada)
NOTIFICA (ação solicitada)
```

Uma vez solicitada, todos os tickets da fila devem ser notificados na mesma ordem de inserção. Um exemplo de notificação é ilustrado a seguir:

```
---------TICKET-----------
- ID: Tick-1
- Usuario solicitante: 255.942.213-22
- Tipo: Outros
- Descricao: RECOLHER COBRA QUE APARECEU NA ENTRADA DO PREDIO
- Local: PASSARELA DA INFORMATICA
- Nivel de Dificuldade: 5
- Tempo Estimado: 5h
- Status: Finalizado
-------------------------
```

Como pode ser observado, a notificação nada mais é do que apresentar todas as informações do ticket.

Banco de usuários

O sistema deve permitir que todos os usuários cadastrados sejam exibidos na tela seguindo a ordem de cadastro. Essa ação é solicitada da seguinte forma:

```
E (caractere indicando que alguma ação será executada)
USUARIOS (ação solicitada)
```

Uma vez solicitada, os usuários cadastrados até aquele momento devem ser exibidos na tela. Um exemplo para um único usuário é mostrado a seguir:

```
- Nome: TAHM KENCH
- CPF: 123.932.250-46
- Data de Nascimento: 1/9/1955
```

```
- Telefone: (34)96373-7004
- Genero: OUTROS
- Setor: RH
- Tickets solicitados: 2
```

Observe que além das informações do usuário também é exibido quantos tickets o usuário já solicitou no sistema.

Banco de técnicos

O sistema deve permitir que todos os técnicos cadastrados sejam exibidos na tela seguindo a ordem de cadastro. Essa ação é solicitada da seguinte forma:

```
E (caractere indicando que alguma ação será executada)
TECNICOS (ação solicitada)
```

Uma vez solicitada, os técnicos cadastrados até aquele momento devem ser exibidos na tela. Um exemplo para um único técnico é mostrado a seguir:

```
- Nome: JAYCE BARBOSA
- CPF: 854.632.894-20
- Data de Nascimento: 22/2/1990
- Telefone: (37)9938-3382
- Genero: MASCULINO
- Area de Atuacao: GERAL
- Salario: 2000.00
- Disponibilidade: 2h
- Tempo Trabalhado: 5h
```

Observe que também é exibida a disponibilidade atual do técnico e o tempo em que ele já trabalhou.

Rankings

Existe uma variação possível dos bancos de usuários e técnicos. A solicitação pode vir com a palavra RANKING à frente, por exemplo:

```
E (caractere indicando que alguma ação será executada)
RANKING TECNICOS (ação solicitada)
```

ou

```
E (caractere indicando que alguma ação será executada)
RANKING USUARIOS (ação solicitada)
```

Neste caso a exibição **não** segue a ordem de cadastro. No caso dos técnicos, eles devem ser exibidos de acordo com a quantidade de tempo trabalhado. Já para os usuários, eles devem ser exibidos de acordo com a quantidade de tickets abertos. Em ambos os casos o ranking segue ordem decrescente, ou seja, para os técnicos quem trabalhou mais tempo deve aparecer no topo. Naturalmente, para os usuários, o topo pertence a quem abriu mais tickets. Para ambos os casos, em caso de empate, os atores devem aparecer em ordem alfabética.

Relatório geral

Por fim, o sistema deve ser capaz de gerar um relatório geral do sistema. Essa solicitação é realizada da seguinte forma:

```
E (caractere indicando que alguma ação será executada)
RELATORIO (ação solicitada)
```

O relatório deve exibir as seguintes informações:

- Quantidade de tickets na fila do sistema
- Quantidade de tickets com status Aberto
- Quantidade de tickets com status Finalizado
- Quantidade de usuários cadastrados
- Média de idade dos usuários
- Quantidade de técnicos cadastrados
- Média de idade dos técnicos
- Média de tempo trabalhado dos técnicos

Essas informações devem ser exibidas na tela da seguinte forma

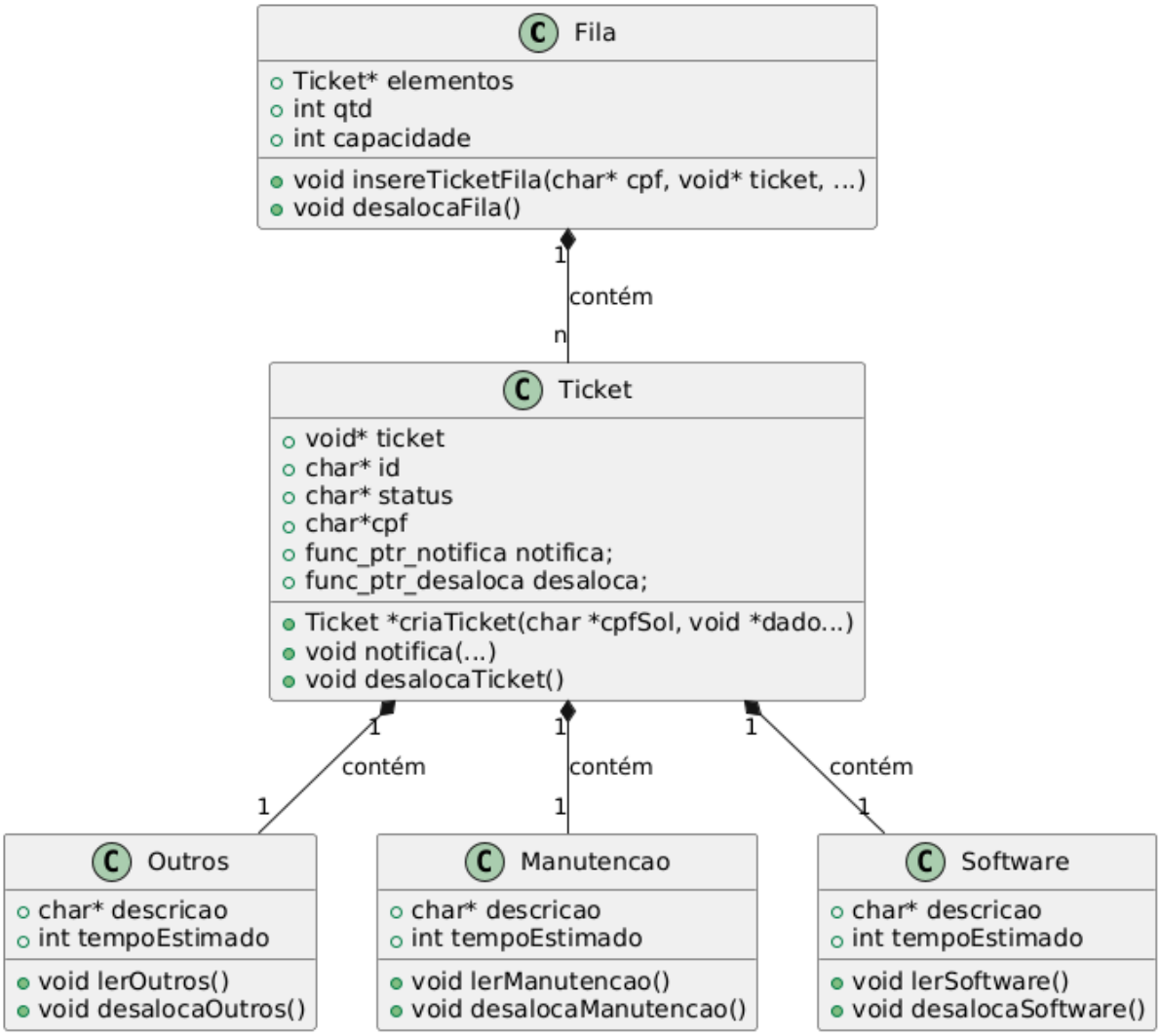
```
---------RELATORIO-----------
- Qtd tickets: 10
- Qtd tickets (A): 4
- Qtd tickets (F): 6
- Qtd usuarios: 12
- Md idade usuarios: 29
- Qtd tecnicos: 5
```

```
- Md idade tecnicos: 33
- Md trabalho tecnicos: 14
-----------------------------
```

Uso de generalização para reaproveitamento de código

Considerando que um dos objetivos do trabalho é aprender a codificar permitindo maior reusabilidade do código, você deverá usar tipos genéricos que permitam estender o tipo de Ticket facilmente.

Considere o tipo Fila e ticket, disponibilizado no Fila.h e representado abaixo, ele recebe chamados (tickets). No entanto, cada chamado tem suas peculiaridades, em especial nas funções de leitura e notificações. Neste sentido, um ticket contém um elemento genérico especializado pelo tipo de um chamado. Desta forma, a fila contém tickets que são elementos genéricos e o tratamento de cada elemento é definido por meio das implementações das respectivas funções de callback nos próprios tipos de Ticket. Por exemplo, a forma de ler um Ticket do formato Sofware é definida juntamente com o struct Software. Este tipo é adicionado a um ticket, que por sua vez é adicionado à fila de tickets.



Informações gerais

Alguns arquivos de entrada e respectivos arquivos de saída serão fornecidos para o aluno. O aluno deverá utilizar tais arquivos para testes durante a implementação do trabalho. É de responsabilidade do aluno criar novos arquivos para testar outras possibilidades do programa e garantir seu correto funcionamento. O trabalho será corrigido usando, além dos arquivos dados, outros arquivos (específicos para a correção e não disponibilizados para os alunos) seguindo a formatação descrita neste documento. Em caso de dúvida, entre em contato com os professores e/ou monitores. O uso de arquivos com formatação diferente poderá acarretar em incompatibilidade durante a correção do trabalho e consequentemente na impossibilidade de correção do mesmo (sendo atribuído a nota zero). Portanto, siga estritamente o formato estabelecido.

Implementação e verificação de resultados

A implementação deverá seguir o *template* de código disponibilizado no repositório do trabalho dentro da organização da disciplina no Github:

- Link para o repositório: https://github.com/prog-II-ufes/2024-2-template-TP1-etapa-1

A utilização do template segue os mesmos princípios dos exercícios disponibilizados ao longo da disciplina. Você deve utilizar os arquivos .h disponibilizados e, obrigatoriamente, implementar todas as funcionalidades requisitadas para a construção destes TADs. Todavia, vocês são livres para implementar outros TADs e/ou bibliotecas que julgarem necessárias. Além disso, você já possui acesso ao script de correção automática de exercícios e trabalhos. Ele será utilizado para corrigir o seu trabalho. Desta forma, é extremamente importante que você avalie seu trabalho utilizando o script e os casos de teste disponibilizados.

Para fins de depuração, você também pode utilizar softwares de comparação de arquivos, como diff e o Meld (alternativa gráfica do diff). Diferenças na formatação poderão impossibilitar a comparação e consequentemente impossibilitar a correção do trabalho. O programa será considerado correto se gerar a saída esperada idêntica à fornecida com os casos de teste.

Pontuação e Processo de Correção

A primeira etapa do trabalho será pontuada de acordo com sua implementação e seguindo os pontos apresentados na tabela a seguir.

Item	Descrição	Peso
fila.h	TAD relacionado a fila	10%
ticket.h	TAD relacionado a um ticket	10%
software.h	TAD relacionado ao tipo de ticket SOFTWARE	5%
main.c + restante	Função principal do programa + restante do código necessário	75%

A pontuação da tabela será utilizada para calcular a nota para cada caso de teste. A pontuação final será a média de pontos considerando todos os casos. Haverá casos

de teste visíveis e ocultos (que serão divulgados durante a segunda etapa do trabalho em sala).

A pontuação obtida de acordo com a tabela anterior será utilizada para calcular a nota final do seu trabalho. O gerenciamento de memória e modularização são itens obrigatórios. Sendo assim, não basta apenas acertar a saída, é necessário seguir os templates de código disponibilizados e gerenciar corretamente o uso da memória. Logo, considere que seu trabalho seja capaz de gerar todos os arquivos corretamente e obtenha pontuação igual a 10. A partir desta pontuação a sua nota final será calculada da seguinte maneira:

(1) primeiramente será verificado se você seguiu corretamente o template de código fornecido.

Se você implementou todos os templates corretamente, não haverá nenhum desconto na sua pontuação nesta fase. Porém, se você deixar de implementar algum dos templates, sua pontuação será descontada de acordo com o peso do template.

(2) após a verificação dos templates, será verificado o gerenciamento de memória através do Valgrind. Caso o seu programa apresente vazamento de memória e/ou erro de memória será **descontado 50%** da pontuação que chegar até essa fase.

Sendo assim, é de **suma importância que seu trabalho siga a risca os templates e gerencie corretamente a memória** (lembre-se, esses são os principais objetivos desta disciplina). Todo esse processo de pontuação será realizado por meio do script de correção que vocês já possuem acesso. Observe que a compilação da pasta completo (dentro de resultados) no script não é utilizado para computar a pontuação. Porém, pode ser utilizado para depuração de erros no seu programa.

Processo de correção do trabalho: como consta no plano de ensino da disciplina, o trabalho prático é dividido em duas etapas. A primeira etapa é o desenvolvimento do projeto seguindo as regras e descrições apresentadas neste documento. A segunda etapa consiste na correção, atualização e/ou incremento de funcionalidades implementadas na etapa anterior. Esta última etapa é realizada em sala de aula de acordo com cronograma da disciplina. A nota final do trabalho é a média entre as duas etapas, ou seja, cada etapa vale 50% da nota.
É importante ressaltar que ao iniciar a etapa 2 vocês terão acesso à nota e aos casos de teste ocultos da etapa 1. Portanto, vocês poderão corrigir possíveis erros cometidos durante o desenvolvimento da etapa 1.

Prazo e submissão

Todas as informações referentes a prazo de entrega e forma de submissão estão disponíveis na atividade **Trabalho Prático - Etapa 1** no classroom/ava da disciplina.

Regras gerais

Para todas as etapas do trabalho, considere as seguintes regras:

- Trabalhos entregues após o prazo **não** serão corrigidos (recebendo a nota zero)
- O trabalho deverá ser feito **individualmente** e pelo próprio aluno, isto é, o aluno deverá necessariamente conhecer e dominar **todos** os trechos de código criados. Cada aluno deverá trabalhar independente dos outros, não sendo permitido a cópia ou compartilhamento de código.
    - Haverá verificação automatizada de plágio. Trabalhos identificados como iguais, em termos de programação (por exemplo, mudar nomes de variáveis e funções, entre outros, não faz dois trabalhos serem diferentes), **serão penalizados com a nota zero e poderão ser submetidos para penalidades adicionais em instâncias superiores.** Isso também inclui a pessoa que forneceu o trabalho, sendo, portanto, de sua obrigação a proteção de seu trabalho contra cópias ilícitas. Proteja seu trabalho e não esqueça cópias do seu código nas máquinas de uso comum.
- Nenhuma entrada dos dados nos casos de teste vai possuir acentos ou caracteres especiais. Além disso, estarão sempre em caixa alta (ex: JOAO).
- Todos os arquivos de entrada vão seguir sempre o mesmo padrão.
- Todas as entradas de dados serão fornecidas de acordo com a descrição. Por exemplo, se o padrão do CPF é 000.000.000-00, todos CPFs fornecidos vão estar neste padrão.
- Para dados que resulte em flutuante, use sempre a parte inteira. Por exemplo, 3.14 deve ser exibido como 3.
- **Para calcular a idade dos atores considere a data de referência como sendo 18/02/2025 (dia em que o trabalho foi divulgado)**
- Para dados multidimensionais no qual o tamanho é desconhecido, é obrigatório o uso de alocação dinâmica de memória. Liberar a memória alocada é de sua responsabilidade.
    - Caso você tente burlar essa regra, você será penalizado
- O programa deve ser encerrado quando o caractere F (de **F**inalizar) for lido pela entrada padrão.

- Modularização e organização são fundamentais na construção do código. Você é obrigado a implementar os TADs solicitados via o template dos .h. Todavia, isso não impede a criação de outros.
- Todos os trabalhos serão corrigidos utilizando o sistema operacional Linux. **Garanta que seu código funcione nos computadores do Labgrad**. Qualquer discrepância de resultado na correção por conta de Sistema Operacional, os computadores do Labgrad serão utilizados para solucionar o problema.
- **Todos os TADs implementados no trabalho devem ser opacos**
- Possíveis problemas na descrição do trabalho serão solucionados e comunicados via Discord/Classroom. Don't Panic!

Considerações finais

Esse documento descreve de maneira geral as regras de implementação do trabalho. É de responsabilidade do aluno garantir que o programa funcione de maneira correta e amigável com o usuário. Qualquer alteração nas regras do trabalho será comunicada em sala e no portal do aluno. É responsabilidade do aluno frequentar as aulas e se manter atualizado em relação às especificações do trabalho. Caso seja notada qualquer tipo de inconsistência nos arquivos de testes disponibilizados, comunique imediatamente ao professor para que ela seja corrigida e reenviada para os alunos.