
Universidade Federal
de São João del-Rei

# TRABALHO I- DISCIPLINA DE AEDS 2

Davi Medeiros Gonçalves - 212050085
Rafael Carvalho Avidado Geraldo - 212050057

# Introdução:

O trabalho proposto consiste em implementar um simulador do funcionamento de um servidor de Emails, com algumas funções e operações comuns a esse tipo de programa, sendo elas:

*Cadastrar um novo usuário*
*Remover um usuário*
*Entregar Email*
*Consultar Email*

Para implementar com sucesso, era esperado que usássemos o conteúdo já visto na matéria, tais como fila, pilha, lista além é claro de um gerenciamento de memória bem feito e coeso ao programa.

Veremos tudo isso mais a fundo a seguir, nos demais tópicos.

# Implementação:

**Estrutura de dados:**

Para a implementação do trabalho, foram criados Tipos Abstratos de Dados seguindo o padrão de lista, onde um deles era referente ao e-mail contendo: Prioridade, conteúdo, número de e-mails, ponteiro para Email_t que aponta para o próximo e-mail na lista.

Além disso, foi utilizada a "struct user" contendo o id, tamanho do inbox e um ponteiro para a estrutura Email, já que a estrutura não armazena a última mensagem enviada pelo usuário.

## Funções e procedimentos:

**void cadastraUser (int ID):**

A função recebe um id como parâmetro e tem o objetivo de cadastra um novo usuário no sistema. Para isso ela percorre todos os usuários já cadastrados e verifica se algum usuário já possui o mesmo ID passado como parâmetro. Caso exista, ela retorna uma mensagem de erro, caso contrário ela verifica se o número está dentro do máximo de usuários cadastrados e se houver espaço ela adiciona o ID passando como parâmetro NULL para a caixa de entrada do usuário e o tamanho da caixa de entrada igual a 0.

Por fim, retorna a mensagem de OK: USUARIO CADASTRADO.

**void removeUser (int ID):**

A função percorre todos os usuários já cadastrados para buscar o ID informado como parâmetro. Se encontrar o usuário, a função libera a memória utilizada para todas as mensagens contidas na caixa de entrada e, em seguida, libera a memória alocada para a estrutura user e reorganiza a estrutura users. Caso o usuário não exista, a função imprime uma mensagem de erro.

**void criaEmails(int ID, int prioridade, char *conteudo):**

Inicialmente, a função percorre a lista de usuários cadastrados para encontrar aquele com o ID igual ao informado. Se o ID for encontrado, a função armazena a referência para ele na variável "destino". Caso contrário, a função imprime uma mensagem de erro informando que a conta é inexistente. Em seguida, a função cria uma nova estrutura "Email", que recebe a prioridade, o conteúdo de Email e o número de Emails na caixa do usuário. Para o conteúdo, foi utilizado a função "strncpy" para copiar os primeiros 1018 caracteres (1024-6) para que a mensagem termine com a palavra "FIM".

Por fim, a função atribui a referência para Email criada ao ponteiro "prox" do novo Email, adiciona a nova estrutura "Email" à lista de Email do usuário com o ID informado como parâmetro e envia a mensagem de confirmação do envio da mensagem.

**void consultaEmail(int ID):**

A função recebe um ID e busca o usuário correspondente na estrutura users, percorrendo todos os usuários cadastrados. Se o usuário não for encontrado, a função imprime uma mensagem de erro e retorna. Caso contrário, a função verifica se a caixa de entrada do usuário está vazia. Se estiver vazia, a função imprime uma mensagem informando que a caixa de entrada está vazia e retorna. Caso contrário, a função cria uma lista auxiliar "emailOrdem", que armazena os emails do usuário de forma ordenada por prioridade.

A lista é inicializada com NULL para todas as prioridades de 0 até MAX_PRI. Em seguida, a função percorre a lista de emails na caixa de entrada do usuário e remove cada email da lista original, atualizando o ponteiro "prox" do email anterior para apontar para o próximo email da lista. O email é adicionado na lista auxiliar de acordo com sua prioridade.

Depois disso, a função percorre a lista emailOrdem de trás para frente, imprimindo todo o conteúdo de cada email na tela, juntamente com sua prioridade e o ID do usuário consultado. Por fim, a função limpa a caixa de entrada do usuário, definindo o ponteiro "inbox" como NULL e o tamanho da caixa de entrada como 0.

## Resultados e discussões:

A implementação do programa ocorreu como o esperado, assim como os resultados obtidos, e os testes feitos utilizando o programa, vale destacar a importância das estruturas de dados como fundamentos da construção do mesmo.

## Conclusão:

O trabalho como ferramenta de ensino foi excepcional, nos trazendo uma situação “real” e de algo que já estamos habituados, o Email, isso foi importante do ponto de vista que era mais fácil notar como algumas implementações deveriam funcionar. Outro ponto de destaque é que o trabalho abrange a matéria como um todo.

Quanto a dificuldade, nossa maior foi com a implementação de consulta por ela envolver ordenação ao criar uma lista auxiliar, em partes por termos feito o trabalho num momento anterior a explicação da estrutura de lista.