

INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

## RELATÓRIO TÉCNICO

### ATTENDANCE LINE

*Arthur Cadore Matuella Barcella*

*Gabriel Luiz Espindola Pedro*

*Matheus Pires Salazar*

### RESUMO

Para a realização desta tarefa, o projeto foi orquestrado por partes, primeiramente recebendo o arquivo de configuração e separando seus componentes.

Após concluir essa etapa inicial, partimos para o desenvolvimento da interface inicial para o usuário final, que mostraria o menu onde poderíamos ver o tipo de usuário que estava acessando o sistema. Assim que finalizamos esse item, e também com os dados já separados, partimos para o desenvolvimento de exibição de classes ao usuário final segundo as diretrizes do arquivo, e ao mesmo tempo, receber e armazenar os dados inseridos por parte do cliente.

Após finalizarmos essa etapa, passamos para o desenvolvimento da função atendente, utilizamos funções particionadas em vários arquivos, para reaproveitarmos as funções já implementadas no início do sistema, e então, ordenamos os clientes novamente segundo as diretrizes do arquivo para então decidir qual dos clientes era o próximo a ser chamado pelo atendente.

14/05/2022



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

## INTRODUÇÃO

É apresentado o cenário de um ambiente de atendimento onde é necessário segmentar a entrada dos clientes em classes de atendimento, onde cada classe tem uma prioridade. Em seguida, é necessário que os clientes sejam chamados para o atendimento de acordo com sua prioridade e também em relação ao tempo que esperaram na fila para serem atendidos.

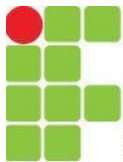
## DESENVOLVIMENTO

Na implementação do projeto, as seguintes estruturas de dados foram utilizadas:

- **Estruturas:**

As estruturas foram utilizadas em nosso programa para separar cada informação através de parâmetros e ao mesmo tempo, mantê-las unidas em um mesmo grupo que poderia ser facilmente acessado ao longo da aplicação. Utilizamos duas estruturas no total, a primeira sendo a “classe” que recebe os valores do arquivo de configuração, a segunda sendo a “senha”.

```
struct senha {  
    string senha_usuario;  
    time_t expiracao;  
    int prioridade;  
};  
  
struct classe {  
    string nome;  
    int prioridade;  
    int max_espera;  
    string descricao;  
    int contador_senhas;  
};
```



- **Vetor (de estruturas):**

Em nossa implementação utilizamos um vetor de estrutura com a estrutura “classe” definida. Inicialmente, importamos uma linha completa do arquivo CSV informado no argumento da linha de comando, em seguida essa linha é dividida em várias strings a partir de uma função separadora (com delimitador sendo o caractere vírgula “,”), em seguida, as strings separadas são importadas dentro de cada parâmetro da estrutura, e por fim, esta é armazenada, no vetor de estruturas. Posteriormente, o vetor de estruturas é consultado na função clientes para informar os parâmetros a serem exibidos na CLI para o usuário final. [OBJ]

```
vector<string> string_to_vector(const string& input,
                                const string& str_separator) {
    int space = 0;

    vector<string> line;

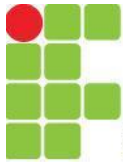
    string aux;

    while (space != string::npos) {
        int word = input.find_first_not_of(str_separator, space);

        if (word == string::npos) break;

        space = input.find(str_separator, word);
        if (space == string::npos) {
            aux = input.substr(word);
        } else {
            aux = input.substr(word, space - word);
        }
        line.push_back(aux);
    }
}
```

Abaixo podemos verificar a função utilizada para importar os valores recebidos pelo vetor para cada parâmetro da estrutura de “classes”, que posteriormente é enviada para a lista de classes:



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

```
classe cria_classe(const string& linha_csv) {  
    auto string_separada = string_to_vector(linha_csv, ",");  
  
    classe nova_classe;  
  
    nova_classe.nome = string_separada[0];  
    nova_classe.prioridade = stoi(string_separada[1]);  
    nova_classe.max_espera = stoi(string_separada[2]);  
    nova_classe.descricao = string_separada[3];  
    nova_classe.contador_senhas = 0;  
  
    return nova_classe;  
}
```

- **Lista (de estruturas):**

Por fim, utilizamos uma fila para armazenar as estruturas passadas geradas para cada cliente, a estrutura de “senha” recebe no total 3 (três) parâmetros, sendo eles:

- **“senha\_usuario”**: Classe da senha retirada da estrutura “classe” através do valor informado pelo cliente, somada ao número da senha, também retirado da estrutura “classe” porém incrementado em uma unidade;
- **“expiracao”**: Tempo exato da geração da senha somada ao tempo limite de espera do cliente, para posterior verificação do tempo limite de espera (o valor é multiplicado por 60, visto que o tempo informado pela função time é em segundos e o descrito no CSV é em minutos);
- **“prioridade”**: Valor de prioridade da classe do cliente, para ordenamento do mesmo na lista que será devolvida para a função atendente já ordenada em ordem de chegada e prioridade;



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

```
classe& classe_escolhida = classes[client_input - 1];
classe_escolhida.contador_senhas++;

senha nova_senha;

nova_senha.senha_usuario =
    classe_escolhida.nome + to_string(classe_escolhida.contador_senhas);
nova_senha.expiracao = time(NULL) + (classe_escolhida.max_espera * 60);
nova_senha.prioridade = classe_escolhida.prioridade;

cout << endl << nova_senha.senha_usuario << endl;

senhas.push_back(nova_senha);

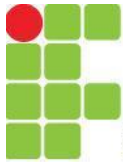
senhas.sort(compara_senhas);
```

## RESULTADOS

Para comprovar o funcionamento do programa, utilizamos um arquivo de teste para configurar o sistema, e então adicionamos vários clientes fictícios à lista de clientes (com tempo de espera baixo para que pudéssemos realizar os testes).

A seguir segue uma tabela de exemplo com os dados utilizados no arquivo .csv para teste:

TABELA DE DADOS DE EXEMPLO UTILIZADA NO PROGRAMA PARA TESTES			
CLASSE	PRIORIDADE	TEMPO MÁXIMO	DESCRIÇÃO
A	1	1	Criação de contrato
B	4	1	Abertura de conta
C	5	5	Atendimento normal
D	2	10	Atendimento prioritário
E	3	12	Saque ou reclamação



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

Ao executar o programa inicialmente, o arquivo de configuração é enviado para o sistema e então os parâmetros contidos no arquivo são armazenados individualmente em uma fila de estruturas, e também é adicionado um contador para cada conjunto de parâmetros informado, abaixo segue a função utilizada na segmentação:

```
#include "cria_classe.h"

classe cria_classe(const string& linha_csv) {
    auto string_separada = string_to_vector(linha_csv, ",");

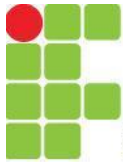
    classe nova_classe;

    nova_classe.nome = string_separada[0];
    nova_classe.prioridade = stoi(string_separada[1]);
    nova_classe.max_espera = stoi(string_separada[2]);
    nova_classe.descricao = string_separada[3];
    nova_classe.contador_senhas = 0;

    return nova_classe;
}
```

Em seguida, o programa entra em loop permanente, onde informará ao usuário as opções de operação, sendo estas “cliente” e “atendente”.

Ao escolher a opção “cliente” os dados armazenados na fila de estrutura são exibidos para o usuário no terminal, e então o usuário pode escolher a opção que deseja, conforme o exemplo abaixo:



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

## Menu

```
-> Digite '1' para acesso à interface de cliente  
-> Digite '2' para acesso à interface de atendente  
1
```

Voce escolheu a opção cliente!

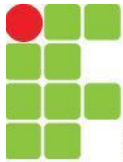
Por favor, escolha qual seu atendimento!

```
Pressione 1 para CRIAÇÃO DE CONTRATO  
Pressione 2 para ABERTURA DE CONTA  
Pressione 3 para ATENDIMENTO NORMAL  
Pressione 4 para ATENDIMENTO PRIORITARIO  
Pressione 5 para SAQUE OU RECLAMAÇÃO  
1
```

A1

Após a inserção do valor por parte do usuário, o programa irá verificar qual opção o cliente escolheu a partir da correlação entre o valor inserido e os valores exibidos, e então irá adicionar os parâmetros necessários para registro dentro da fila de estruturas correspondente.

Abaixo é possível verificar o método utilizado para inserir os dados na fila, e também as modificações realizadas nos parâmetros antes da inserção. Por fim, após inserir o usuário na fila e imprimir a senha para o mesmo, é executada a função “sort” para ordenar os clientes a nível de prioridade.



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

```
classe& classe_escolhida = classes[client_input - 1];
classe_escolhida.contador_senhas++;

senha nova_senha;

nova_senha.senha_usuario =
    classe_escolhida.nome + to_string(classe_escolhida.contador_senhas);
nova_senha.expiracao = time(NULL) + (classe_escolhida.max_espera * 60);
nova_senha.prioridade = classe_escolhida.prioridade;

cout << endl << nova_senha.senha_usuario << endl;

senhas.push_back(nova_senha);

senhas.sort(compara_senhas);
```

Após finalizar a função, o sistema retorna para o menu principal para receber o próximo usuário que poderá ser outro cliente a se registrar, ou então, um atendente.

Após selecionar a opção de atendente, o programa retorna para o mesmo a senha gerada com o maior nível de prioridade e/ou com tempo de espera excedido (o tempo de espera excedido é verificado antes da prioridade).

Menu

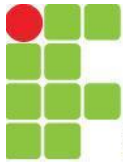
```
-> Digite '1' para acesso à interface de cliente
-> Digite '2' para acesso à interface de atendente
```

2

Voce escolheu a opção atendente!

A1





INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

Abaixo está o método utilizado para verificar se o cliente excedeu o tempo limite de espera e neste caso, o mesmo é encaminhado para outra fila (que será priorizada em relação a fila padrão), desta maneira, o ordenamento por prioridade é mantido, mas se os clientes estiverem com tempo de espera expirado, receberão mais prioridade em relação aos clientes que ainda não tiveram seu tempo excedido.

```
for (auto const& senha : senhas) {
    if (senha.expiracao < tempo_atual) {
        senhas_expiradas.push_back(senha);
    }
}

if (!senhas_expiradas.empty()) {
    senhas_expiradas.sort(compara_senhas);

    cout << senhas_expiradas.front().senha_usuario << endl;

    for (auto it = senhas.begin(); it != senhas.end(); it++) {
        if (it->senha_usuario == senhas_expiradas.front().senha_usuario) {
            senhas.erase(it);
            break;
        }
    }
}
```

## MANUAL

Para utilizar o sistema de atendimento é necessário realizar alguns comandos iniciais, e apresentar um arquivo de configurações de segmentação que o programa irá utilizar como referência, abaixo segue um descritivo de cada item:

### 1. Arquivo de configuração (CSV):



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

Para execução do projeto é necessário que haja um arquivo de configuração prévio com os seguintes itens separados pelo caractere vírgula (","): Para utilizar o sistema de atendimento é necessário realizar alguns comandos iniciais, e apresentar um arquivo de configurações de segmentação que o programa irá utilizar como referência, abaixo segue um descritivo de cada item:

- Classe (carácter),
- Prioridade (número inteiro),
- Tempo máximo de espera (número inteiro, em minutos),
- Descrição (texto).

Esses quatro parâmetros devem ser informados para cada classe que for desejada a segmentação.

## 2. Comandos iniciais:

1º Comando: Gera o Makefile para compilação de todos os arquivos do projeto.

**Sintaxe: cmake CMakeLists.txt**

2º Comando: Compila os arquivos informados do projeto gerando os arquivos para execução.

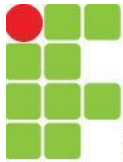
**Sintaxe: make**

3º Comando: Executar o código do projeto:

**Sintaxe: ./Projeto1 {arquivo de configuração}**

Caso não seja encaminhado um valor no primeiro argumento da linha de comando, o sistema retornará erro na execução, dessa maneira, é necessário encaminhar em todas as execuções o nome do arquivo de configuração.

## 3. Execução do código para o cliente:



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

Para acessar o menu do cliente, o mesmo deve digitar na linha de comando o numeral 1 (um), e então acessar as opções de senha que poderá escolher, abaixo um exemplo da execução inicial do código:

```
Menu
```

```
-> Digite '1' para acesso à interface de cliente  
-> Digite '2' para acesso à interface de atendente
```

```
1
```

```
Voce escolheu a opção cliente!
```

O terminal aceitará apenas valores inteiros correspondentes a lista, caso o usuário tente digitar um caractere alfanumérico, string, ou um número fora da lista, o programa retornará o erro de inserção e solicitará que o usuário insira novamente os dados.

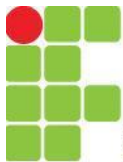
Uma vez nas opções de senha, o cliente pode escolher entre todas as opções cadastradas no arquivo .csv, as opções serão listadas em ordem sequencial, para o cliente selecionar a opção que deseja, bastará digitar o número correspondente a linha desejada, conforme o exemplo abaixo:

```
Por favor, escolha qual seu atendimento!
```

```
Pressione 1 para CRIAÇÃO DE CONTRATO  
Pressione 2 para ABERTURA DE CONTA  
Pressione 3 para ATENDIMENTO NORMAL  
Pressione 4 para ATENDIMENTO PRIORITARIO  
Pressione 5 para SAQUE OU RECLAMAÇÃO
```

```
2
```

```
B1
```



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

O terminal aceitará apenas valores inteiros correspondentes a lista, caso o usuário tente digitar um caractere alfanumérico, string, ou um número fora da lista, o programa retornará o erro de inserção e solicitará que o usuário insira novamente os dados.

#### 4. Execução do código para o atendente:

Para acessar o menu do atendente, o mesmo deve digitar na linha de comando o numeral 2 (dois), e então o programa irá retornar a próxima senha de atendimento, conforme o exemplo abaixo:

Menu

```
-> Digite '1' para acesso à interface de cliente  
-> Digite '2' para acesso à interface de atendente  
2
```

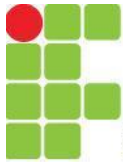
Voce escolheu a opção atendente!

A1

O terminal aceitará apenas valores inteiros correspondentes a lista, caso o usuário tente digitar um caractere alfanumérico, string, ou um número fora da lista, o programa retornará o erro de inserção e solicitará que o usuário insira novamente os dados.

## CONCLUSÃO

O programa conseguiu atingir todos os objetivos propostos para este trabalho, onde foi criado a configuração de classes, a livre escolha de qual atendimento o cliente deseja e o pleno funcionamento da fila de atendimento seguindo os requisitos de tempo máximo de espera para cada classe e a ordem de prioridade dentre todas as classes.



INSTITUTO FEDERAL  
SANTA CATARINA

MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

CURSO DE ENGENHARIA DE TELECOMUNICAÇÕES - CÂMPUS SÃO JOSÉ

## BIBLIOGRAFIA

- Ordenamento de estruturas, utilizado no início do código para receber os parâmetros de configuração descritos no arquivo:

<https://stackoverflow.com/questions/21233597/sorting-a-list-of-structs>

- Lista dinâmica encadeada, utilizada para ordenar os clientes:

<https://pt.stackoverflow.com/questions/61065/como-criar-uma-lista-din%C3%A2mica-encadeada-dentro-de-outra-lista-din%C3%A2mica-em-c>

- Vetor de listas, utilizado para criar a quantidade de filas e senhas de acordo com a quantidade de classes passadas:

<https://pt.stackoverflow.com/questions/223533/inserir-elementos-em-vetor-de-listas-c>

- Função utilizada para corrigir loop de impressão, quando passado um char ou string:

<https://www.clubedohardware.com.br/forums/topic/1220075-como-verificar-se-string-e-numerica/>