ATIVIDADE 6: FILA COM ARRANJOS

Pedro Henrique Santos

Proposta:

Atividade 6: Fila com arranjos

Crie um programa que permita ao usuário manter uma fila de pedidos, simulando os pedidos de um restaurante, por exemplo. O programa deve:
- armazenar, no mínimo, o nome do cliente, o número da mesa e a descrição de cada pedido;
- possibilitar a inserção de novos pedidos, a visualização do próximo pedido a ser atendido sem removê-lo, a visualização completa da fila de pedidos e a remoção de pedidos;
- realizar a importação e exportação da fila para arquivo de texto quando o usuário solicitar.

Entregue um arquivo PDF com uma breve descrição do que o programa faz e com prints do código para ilustrar seu trabalho.

Primeiramente foi criada uma *struct* do pedido e a fila na qual os pedidos vão ser armazenados.

```
10  struct Pedido
11  {
12      string nome;
13      int mesa;
14      string descricao;
15  };
16
17  struct FilaDePedidos
18  {
19      int quantidade = -1;
20      Pedido pedido[MAX_PEDIDOS];
21  };
22
23  FilaDePedidos pedidos;
```

```
int main()
{
    UINT CPAGE_UTF8 = 65001;
    UINT CPAGE_DEFAULT = GetConsoleOutputCP();
    SetConsoleOutputCP(CPAGE_UTF8);

    int opcao;
    do
    {
        system("cls");

        if (pedidos.quantidade > -1)
        {
            cout << "\tPedido Atual" << endl
                 << endl;
            cout << "Nome: " << pedidos.pedido[0].nome << endl;
            cout << "Mesa: " << pedidos.pedido[0].mesa << endl;
            cout << "Descrição: " << pedidos.pedido[0].descricao << endl
                 << endl;
        }

        cout << "\tOpções" << endl
             << endl;
        cout << "1- Adicionar pedido" << endl;
        cout << "2- Remover pedido" << endl;
        cout << "3- Ver proximo pedido" << endl;
        cout << "4- Ver todos os pedidos" << endl;
        cout << "5- Importar pedidos" << endl;
        cout << "6- Exportar pedidos" << endl;
        cout << "0- Fechar programa" << endl
             << endl;

        cout << "Digite: ";
        cin >> opcao;
```

Na minha função *main()* eu irei exibir o primeiro pedido a ser feito caso tenha e logo abaixo o menu que tem base em *switch case* que leva a função de cada tarefa.

```
Pedido leDoTeclado()
{
    Pedido pedido;
    cout << "Nome: ";
    cin.ignore();
    getline(cin, pedido.nome);
    cout << "Mesa: ";
    cin >> pedido.mesa;
    cout << "Descrição: ";
    cin.ignore();
    getline(cin, pedido.descricao);
    return pedido;
}
```

No case 1 será chamada a função *leDoTeclado()* que serve para preencher a fila de pedidos

No case 2 será chamada a função de remoção de pedido que irá remover o pedido atual da fila.

```
void removerPedido()
{
    if (pedidos.quantidade > -1)
    {
        for (int i = 0; i <= pedidos.quantidade; i++)
        {
            pedidos.pedido[i] = pedidos.pedido[i + 1];
        }
        pedidos.quantidade--;
    }
    else
    {
        cout << "Não possui pedidos para remover." << endl;
    }
}
```

```
void proximoPedido()
{
    if (pedidos.quantidade > 0)
    {
        cout << "--------- Pedido 2 ----------" << endl;
        cout << "Nome: " << pedidos.pedido[1].nome << endl;
        cout << "Mesa: " << pedidos.pedido[1].mesa << endl;
        cout << "Descrição: " << pedidos.pedido[1].descricao << endl
             << endl;
    }
    else
    {
        cout << "Não possui próximos pedidos." << endl;
    }
}
```

No case 3 irá mostrar o próximo pedido a ser atendido (se existir), que no caso será o segundo da lista.

No case 4 irá mostrar todos os pedidos da fila

```
void mostrarPedidos()
{
    for (int i = 0; i <= pedidos.quantidade; i++)
    {
        cout << "--------- Pedido " << i + 1 << " ---------" << endl;
        cout << "Nome: " << pedidos.pedido[i].nome << endl;
        cout << "Mesa: " << pedidos.pedido[i].mesa << endl;
        cout << "Descrição: " << pedidos.pedido[i].descricao << endl;
    }
    cout << endl;
}
```

O case 5 e 6 irão usar o comando *fstream* para exportar e importar dedos em um arquivo .txt

```
void salvarPedidos()
{
    if (pedidos.quantidade > -1)
    {
        ofstream file("pedidos.txt");
        if (file.is_open())
        {
            for (int i = 0; i <= pedidos.quantidade; i++)
            {
                if (i == pedidos.quantidade)
                {
                    file << pedidos.pedido[i].nome << endl;
                    file << pedidos.pedido[i].mesa << endl;
                    file << pedidos.pedido[i].descricao;
                }
                else
                {
                    file << pedidos.pedido[i].nome << endl;
                    file << pedidos.pedido[i].mesa << endl;
                    file << pedidos.pedido[i].descricao << endl;
                }
            }
            file.close();
        }
        else
        {
            cout << "Não foi possível abrir o arquivo para salvar os pedidos." << endl;
        }
    }
    else
    {
        cout << "Não possui pedidos para salvar." << endl;
    }
```

```
void lerPedidos()
{
    ifstream file("pedidos.txt");
    if (file.is_open())
    {
        while (!file.eof())
        {
            Pedido pedido;
            getline(file, pedido.nome);
            file >> pedido.mesa;
            file.ignore(numeric_limits<streamsize>::max(), '\n');
            getline(file, pedido.descricao);
            inserirPedido(pedido);
        }
        file.close();
    }
    else
    {
        cout << "Não foi possível abrir o arquivo para ler os pedidos." << endl;
    }
}
```