**Aula XVII - Desenvolvimento Web III**

**Estudo Dirigido - Valor 2,0 pontos**

**ArrayList**

**Laços de Repetição**

**Estruturas Condicionais**

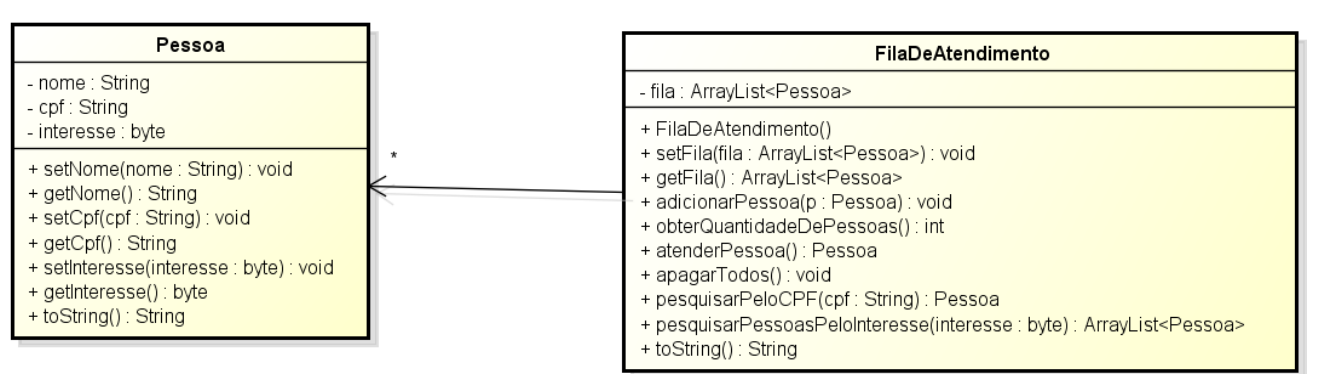
**Manipulação de Classes no Java**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Caso:**

Vamos pensar em uma fila de atendimento bancário, nesta fila cada pessoa que chega específica seu **nome, CPF** e seu **interesse** **(Os interesses podem ser: 1-pagamento de conta, 2-Recebimento de salário, 3-Outros)**.

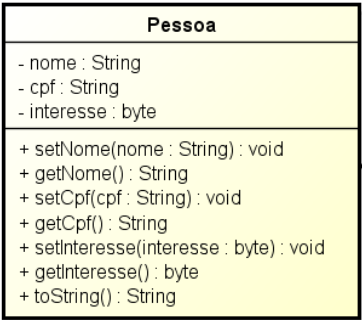
**Vamos desenvolver um sistema para esta fila de atendimento, seu sistema deve permitir adicionar pessoas na fila de atendimento, atender uma pessoa (chame pelo nome e exclua a mesma automaticamente da fila), listar todas as pessoas que estão na fila, ver a quantidade de pessoas que estão na fila, pesquisar uma pessoa pelo CPF, listar as pessoas de acordo com interesse informado no sistema (exemplo: pessoas que desejam efetuar pagamento, recebimento ou outro interesse). No final do expediente tenha uma forma de excluir todas as pessoas da fila. Temos, assim, um objeto Pessoa e um objeto FilaDeAtendimento, de modo que uma fila de atendimento contém uma coleção de objetos da classe Pessoa.**

****

**Abra a sua IDE e crie a estrutura:**

|  |  |
| --- | --- |
|  | * Classe Sistema é a classe Main * Classe Pessoa - classe Java * Classe FilaDeAtendimento - classe Java |

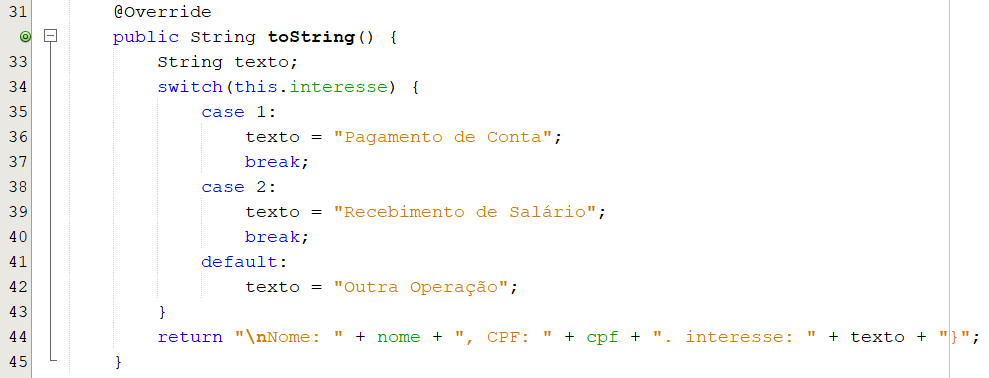
Próximo passo, vamos começar a declarar na **Classe Pessoa**. Observe no Diagrama de Classe os **atributos e métodos.** A classe possui os atributos nome, CPF e interesse privados, os métodos acessores e modificadores sets/ges e o método toString para retornar os dados da Pessoa.



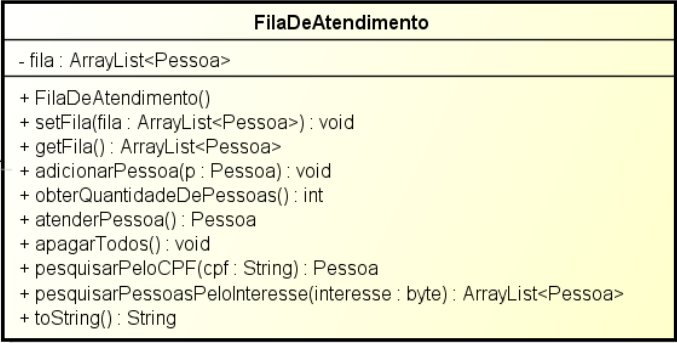
Use o Alt Insert para gerar os getters e setters e vamos criar o método toString:

Vamos usar o **@Override**: ele é muito útil pois ele sobrescreve, sobrepõe os métodos, desta forma otimizamos a memória que o nosso programa utiliza.

Depois disso, vamos criar uma variável do tipo String local e criar as opções do nosso programa usando o Switch Case, com return para os dados que queremos exibir.

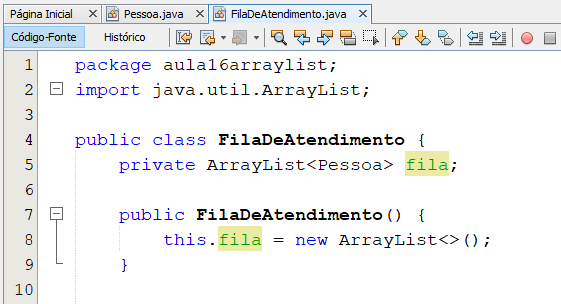


Próxima etapa, será fazer a declaração da Classe **FilaDeAtendimento**. Vamos observar o **Diagrama de Classe**:



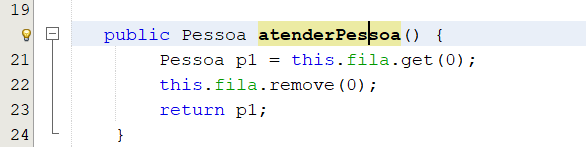
A classe **FilaDeAtendimento** temos um **ArrayList chamado fila que irá armazenar objetos do tipo Pessoa.**

Vamos começar pela declaração dos **ArraysList<>();**

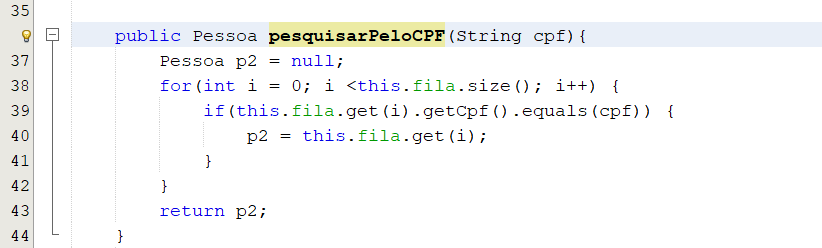


Vamos agora gerar os getters e setters e depois iremos desenvolver os métodos que irão auxiliar na pesquisa dos dados.

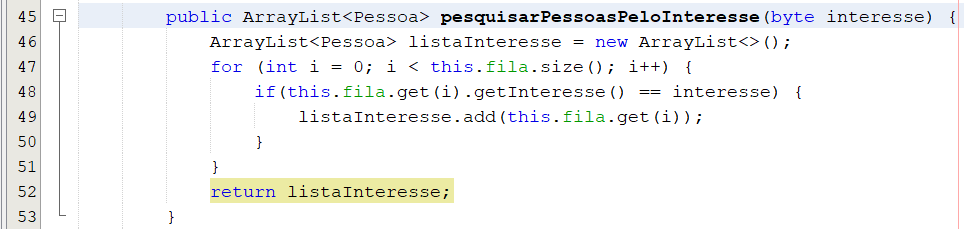
Perceba que o método **atenderPessoa()** **– linha 20 –** neste método criamos uma variável **p1** do tipo pessoa na qual irá armazenar a primeira pessoa da fila, após removemos a primeira pessoa da fila **(simbolizando que a mesma já foi atendida)**. E retornamos a variável p1, como se estivéssemos chamando a mesma para o atendimento.



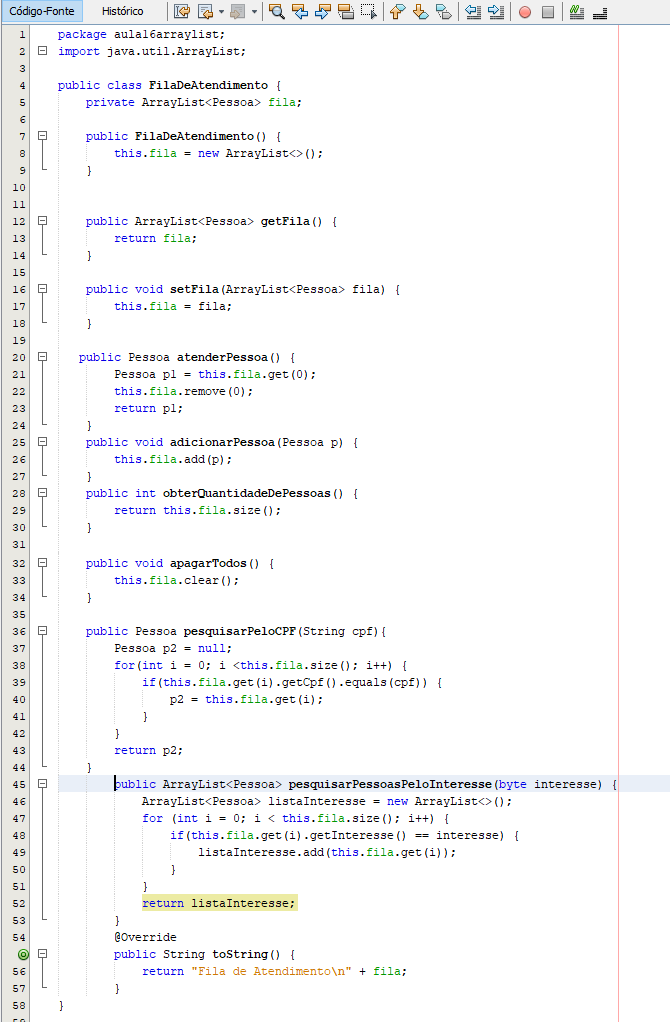
Na linha 36 iniciamos um método que nos permite buscar uma pessoa na fila através do seu CPF, um método de pesquisa específica que irá retornar nenhuma ou apenas uma pessoa, caso tenha o CPF na qual está sendo pesquisado.



Na linha 45 temos o método **pesquisarPessoasPeloInteresse()** no qual o usuário irá informar qual interesse quer pesquisar e o sistema retornará nenhum, uma ou várias pessoas que estiverem com o mesmo interesse, exemplo: visualizar quais as pessoas que desejam pagar conta, quais irão apenas receber salário, ou quais possuem outro interesse qualquer na agência bancária.



**Código completo de FilaDeAtendimento();**



**Classe Sistema (Main):**

Na classe Main, apresentaremos ao usuário as seguintes opções:

|  |
| --- |
| **System.out.println("Digite sua opção:");**  **System.out.println("1 - Adicionar um cliente na fila");**  **System.out.println("2 - Ver cliente da fila");**  **System.out.println("3 - Ver quantidade de clientes que estão na fila");**  **System.out.println("4 - Atender um cliente");**  **System.out.println("5 - Pesquisar um cliente pelo CPF");**  **System.out.println("6 - Pesquisar clientes pelo interesse");**  **System.out.println("7 - Retirar todos os clientes da fila");**  **System.out.println("0 - Sair");** |

Na imagem abaixo do código, percebemos que a **mesma instancia a classe FilaDeAtendimento e dentro do case 1 do menu, instancia a classe Pesso**a. Percebemos também a importância de verificar se há dados antes de executar cada operação, exemplo: no **case 2**: antes de atender uma pessoa, verificamos se há pessoas na fila para atender, caso a fila esteja vazia retornará ao usuário **– linha 50 – “Não há clientes na fila.”**

[**Clique aqui para acessar o código completo da Classe Sistema**](https://drive.google.com/file/d/1Ap6_tKxLHDPga1Ax-30-Y8B1WCYiGFIW/view?usp=sharing)

**Após finalizar o projeto, compacte a pasta e envie para o email:**

[**aulas.qiti@gmail.com**](mailto:aulas.qiti@gmail.com)

**No assunto, colocar Estudo Dirigido e seu nome completo para o caso de ter outra pessoa com o mesmo nome na turma**

**Pode ser entregue até dia 09/06/2020**