throws IllegalArgumentException"

O "throws IllegalArgumentException" é uma cláusula usada em Java para indicar que um método pode lançar uma exceção do tipo IllegalArgumentException. Essa exceção é lançada quando um argumento fornecido a um método é inválido.

Em outras palavras, se um método usa essa cláusula, ele está indicando que o método pode falhar e lançar uma exceção se o argumento fornecido não atender aos critérios esperados.

O IllegalArgumentException é uma subclasse da classe Exception e é uma exceção não verificada. Isso significa que o compilador não obriga a captura ou lançamento dessa exceção. No entanto, é uma boa prática de programação tratar e lançar a exceção apropriada em caso de falha do método.

Quando uma exceção IllegalArgumentException é lançada, é possível acessar a mensagem de erro associada a ela usando o método getMessage(). A mensagem de erro é definida pelo programador e deve indicar qual argumento foi inválido e o motivo pelo qual o argumento foi considerado inválido.

É importante observar que o uso excessivo de IllegalArgumentException pode tornar o código confuso e difícil de manter. É recomendável usá-lo apenas quando os argumentos fornecidos para um método são cruciais para sua execução e precisam ser validados. Além disso, é importante documentar claramente a validação necessária para cada argumento e a exceção que pode ser lançada.

O método matches()

O método matches() é utilizado para verificar se uma string corresponde a uma expressão regular. A expressão regular \\d+ significa que a string deve conter um ou mais dígitos (0-9).

No código espaço! cpf.matches("\\d+") é utilizado para verificar se a string cpf contém somente dígitos. Se cpf contiver algum caractere que não seja um dígito, a expressão regular não será correspondida e o resultado será false, o que levará ao lançamento da exceção IllegalArgumentException. Em outras palavras, o objetivo é garantir que o CPF seja composto somente de números.

Onde pode ser usado :

- Verificar se uma senha atende aos critérios de complexidade definidos (por exemplo, deve conter pelo menos uma letra maiúscula, um número e um caractere especial).
- Verificar se um endereço de e-mail tem um formato válido.
- Validar um número de telefone que segue um padrão específico.

- Verificar se um código de barras ou um número de série de um produto segue um formato definido.

Exemplo:

```
public static boolean verificaSenha(String senha) {
    // Verifica se a senha contém pelo menos uma letra maiúscula
    if (!senha.matches(".*[A-Z].*")) {
        return false;
    }

    // Verifica se a senha contém pelo menos um número
    if (!senha.matches(".*\\d.*")) {
        return false;
    }

    // Verifica se a senha contém pelo menos um caractere especial
    if (!senha.matches(".*[!@#$%^&*(),.?\":{}|<>].*")) {
        return false;
    }

    // Verifica se a senha tem pelo menos 8 caracteres
    if (senha.length() < 8) {
        return false;
    }

    // A senha atende a todos os critérios de complexidade
    return true;
}
```