

The background is a dark blue-grey color. It is decorated with various geometric shapes in teal and white. These include circles of different sizes, some with dotted patterns inside; hexagons, some solid teal and some white outlines; triangles; and lines. Some shapes are partially cut off by the edges of the frame. There are also horizontal and vertical dotted lines scattered throughout the design.

Controle de Níveis de Acesso

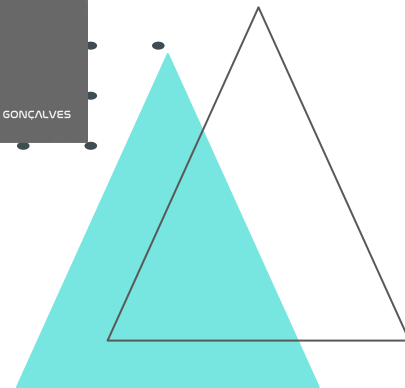
Grupo 3: Bruna, Guilherme B. , Lucas N., Leonardo F. S. e Davi

Sprint Review

- O que foi feito; ✓
- O que não foi feito; ✕
- O que foi acrescentado; +
- O que foi retirado; —



Fonte: luis-goncalves.com



..... **Cadastro de Perfil de Acesso**

Consiste no cadastro de um perfil de acesso, através da criação de um “login” e uma “senha”.

..... Cadastro de Perfil de Acesso

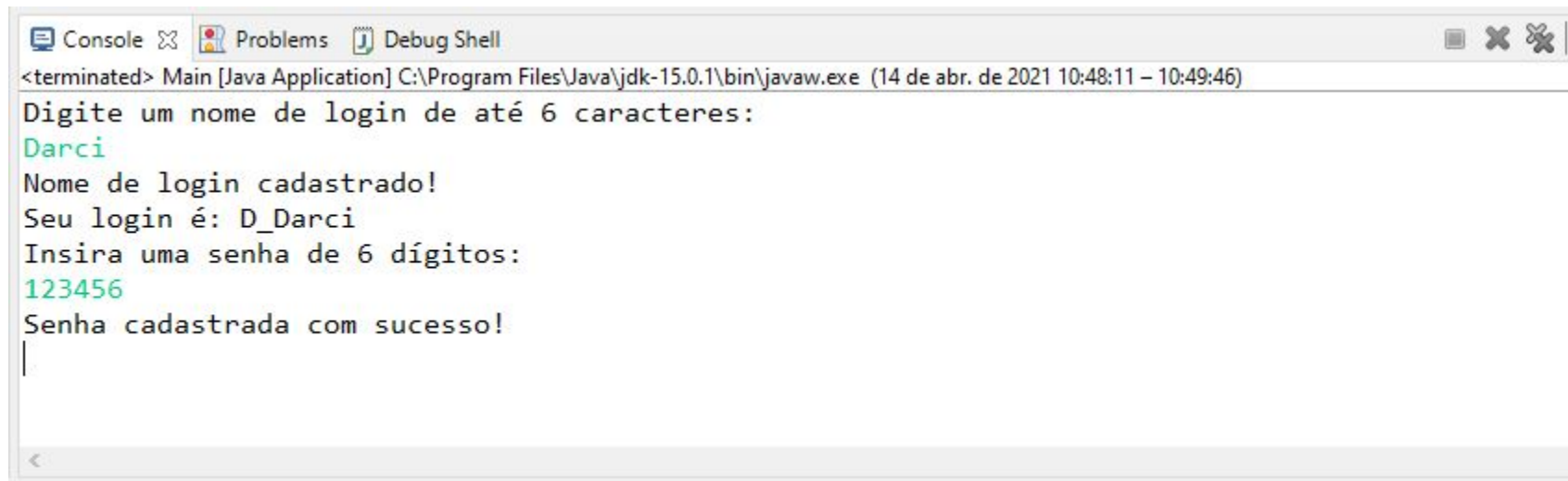
```
public static boolean verificaLogin(String
nome) {
    if (nome.length() <= 6) {
        return true;
    } else {
        return false;
    }
}
```

```
public static boolean verificaSenha(String senha) {
    if (senha.length() == 6) {
        return true;
    } else {
        return false;
    }
}
```



..... Cadastro de Perfil de Acesso

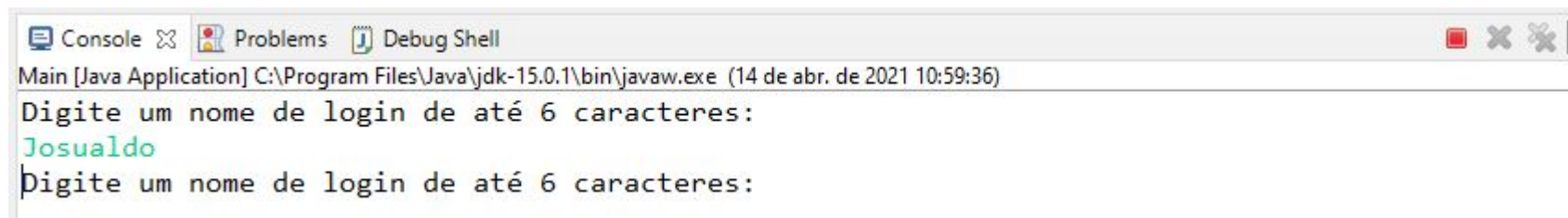
Em execução. Match de nome e senha.



```
<terminated> Main [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (14 de abr. de 2021 10:48:11 – 10:49:46)
Digite um nome de login de até 6 caracteres:
Darci
Nome de login cadastrado!
Seu login é: D_Darci
Insira uma senha de 6 dígitos:
123456
Senha cadastrada com sucesso!
|
```

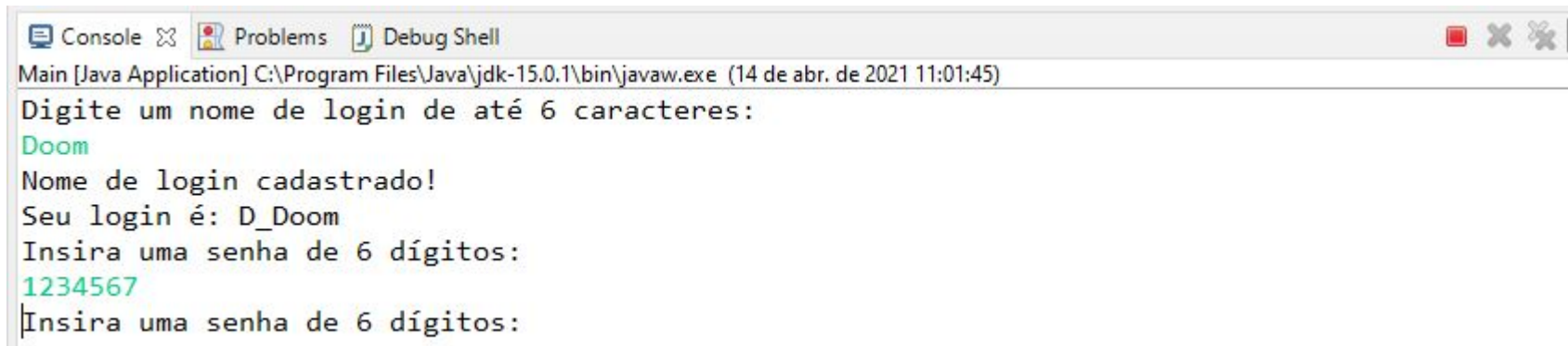
..... Cadastro de Perfil de Acesso

Em execução. Sem match de nome e senha.



The screenshot shows an IDE console window with three tabs: 'Console', 'Problems', and 'Debug Shell'. The title bar indicates the application is 'Main [Java Application]' running 'C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe' at '14 de abr. de 2021 10:59:36'. The console output shows a prompt 'Digite um nome de login de até 6 caracteres:' followed by the input 'Josualdo'. A second prompt 'Digite um nome de login de até 6 caracteres:' is shown with the cursor at the end of the line, indicating the input was not accepted.

```
Console Problems Debug Shell
Main [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (14 de abr. de 2021 10:59:36)
Digite um nome de login de até 6 caracteres:
Josualdo
Digite um nome de login de até 6 caracteres:
```



The screenshot shows the same IDE console window at a later time, '14 de abr. de 2021 11:01:45'. The output shows the first prompt 'Digite um nome de login de até 6 caracteres:' followed by the input 'Doom'. This is followed by the messages 'Nome de login cadastrado!' and 'Seu login é: D_Doom'. Then, a second prompt 'Insira uma senha de 6 dígitos:' is shown followed by the input '1234567'. A third prompt 'Insira uma senha de 6 dígitos:' is shown with the cursor at the end of the line, indicating the input was not accepted.

```
Console Problems Debug Shell
Main [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (14 de abr. de 2021 11:01:45)
Digite um nome de login de até 6 caracteres:
Doom
Nome de login cadastrado!
Seu login é: D_Doom
Insira uma senha de 6 dígitos:
1234567
Insira uma senha de 6 dígitos:
```

..... Cadastro de Perfil de Acesso

```
package br.com.proway.senior.cadastroPerfilAcesso;

import java.util.Scanner;

public class Main {

    static Scanner sc;
    static String senha;
    static String nome;

    public static void main(String[] args) {

        sc = new Scanner(System.in);

        System.out.println("Digite um nome de login de até 6 caracteres:");
        nome = sc.next();

        while (verificaLogin(nome) == false) {
            System.out.println("Digite um nome de login de até 6 caracteres:");
            nome = sc.next();
        }

        String login = nome.charAt(0) + "_" + nome;

        System.out.println("Nome de login cadastrado!");
        System.out.println("Seu login é: " + login);

        System.out.println("Insira uma senha de 6 dígitos:");
        senha = sc.next();

        while (verificaSenha(senha) == false) {
            System.out.println("Insira uma senha de 6 dígitos:");
            senha = sc.next();
        }

        System.out.println(senhaCriada());
    }
}
```

```
static String senhaCriada() {
    return "Senha cadastrada com sucesso!";
}

/**
 * Verifica um nome de login(perfil). Recebe um nome de login digitado pelo
 * usuário de até 6 (seis) caracteres e verifica se está dentro dos padrões
 * definidos.
 *
 * @param nome String É um nome de login que o usuário digita
 * @return Retorna um boolean caso o nome de login esteja verdadeiro
 */
public static boolean verificaLogin(String nome) {
    if (nome.length() <= 6) {
        return true;
    } else {
        return false;
    }
}

/**
 * Verifica uma senha. Recebe uma senha digitada pelo usuário de 6 (seis)
 * caracteres e verifica se está dentro dos padrões definidos.
 *
 * @param senha String É uma senha que o usuário digita
 * @return Retorna um boolean caso a senha seja verdadeira
 */
public static boolean verificaSenha(String senha) {
    if (senha.length() == 6) {
        return true;
    } else {
        return false;
    }
}

}
```

Verificação de login

```
static boolean verificalogin(String login, String senha) {  
  
    if (login.equals("Claudio") && senha.equals("2")) {  
        return true;  
    } else {  
        return false;  
    }  
}
```



Verificação de login

```
verificalogin(login, senha);

if (verificalogin(login, senha) == true) {
    System.out.printf("Usuário %s logado com sucesso.", login);
} else
    System.out.printf("Usuário ou senha invalido(s)");

}
```



..... Recuperar / Alterar Senha

```
public static ArrayList<String> alterarSenha(String email) {  
    ArrayList<String> erro = new ArrayList<String>();  
    if(verificarUsuario(email)) {  
        if(gerenciarCodigo(email)) {  
            solicitarNovaSenha(email);  
        }  
        else {  
            erro.add("Código inválido");  
        }  
    }  
    else {  
        erro.add("Usuário inexistente");  
    }  
    return erro;  
}
```

.....

..... Recuperar / Alterar Senha

```
public static ArrayList<String> alterarSenha(String email) {  
    /**  
     * Controla o processo de alteração de senha  
     *  
     * Verifica se o usuário existe, envia um email para o usuário,  
     * Solicita a nova senha e a salva.  
     *  
     * @param email E-mail do usuário.  
     * @return Lista de erros se houver  
     */  
}
```

..... Recuperar / Alterar Senha

```
public static boolean gerenciarCodigo(String email) {  
    int codigoGerado = gerarCodigo();  
    enviarEmail(email, codigoGerado);  
    int codigoUsuario = solicitarCodigo();  
    return validarCodigo(codigoUsuario, codigoGerado);  
}
```

..... Recuperar / Alterar Senha

```
public static void enviarEmail(String email, int codigoGerado)

public static int solicitarCodigo() : codigo

public static boolean verificarUsuario(String email)

public static int gerarCodigo() : int codigo

public static boolean validarCodigo(int codigoUsuario, int
codigoGerado)

public static void solicitarNovaSenha(String email)

public static void salvarSenha(String senha)
```

..... Cadastro de Permissões

```
public static void cadastrarPermissao(String nomePermissao,  
                                       ArrayList<String> listaPermissoes)  
  
public static void cadastrarPacotePermissoes(String nomePacote,  
                                              ArrayList<String> listaPermissoes,  
                                              ArrayList<Integer> listaPermissoesSelecionadas,  
                                              ArrayList<ArrayList<String>> listaPacotes)
```



..... Cadastro de Permissões

```
ArrayList<String> pacote = new ArrayList<String>();  
  
for(int lista : listaPermissoesSelecionadas) {  
    String item;  
    item = listaPermissoes.get(lista);  
  
    pacote.add(item);  
}  
listaPacotes.add(pacote);
```



..... Cadastro de Permissões

```
@Test
public void testeCadastroPermissao() {
    String nomePermissaoTeste = "Registro ponto";
    ArrayList<String> listaPermissoesTeste = new ArrayList<String>();
    Main.CadastrarPermissao(nomePermissaoTeste,
                             listaPermissoesTeste);

    assertTrue(listaPermissoesTeste.indexOf(nomePermissaoTeste) >= 0);
}
```



..... Verificação de Permissões

Método: packExists

```
public boolean packageExists(ArrayList<String> pack){  
    if(listaPacotes.indexOf(pack) > listaPacotes.size() || listaPacotes.indexOf(pack) < 0 ){  
        return false;  
    }  
    return true;  
}
```

..... Verificação de Permissões

Método: permissionExists

```
public boolean permissionExists(String permission){  
    if(listaPermissao.indexOf(permission) > listaPermissao.size() || listaPermissao.indexOf(permission) < 0){  
        return false;  
    }  
    return true;  
}
```

..... Verificação de Permissões

Método: checkPermissionOnPack

```
public boolean checkPermissionOnPackage(ArrayList<String> pack, String permission){  
    for(int i = 0; i < pack.size(); i++) {  
        if(pack.get(i).equals(permission)) {  
            return true;  
        }  
    }  
    return false;  
}
```

..... Verificação de Permissões

Teste: permissionExists

@Test JUnit

```
public void verificaPermissao() {  
    VerificaAcesso minhaClasse = new VerificaAcesso();  
    minhaClasse.listaPermissao.add("Ponto");  
    minhaClasse.listaPermissao.add("Salário");  
    minhaClasse.listaPermissao.add("Cadastrar Colaborador");  
    minhaClasse.listaPermissao.add("Fechar Mês");  
  
    boolean result = minhaClasse.permissionExists("Ponto");  
    assertEquals(result, true);  
}
```

Sprint Retrospective

- Continuar fazendo;
 - Troca de Conhecimento.
- Fazer mais;
 - Levantar Requisitos;
 - Pesquisas.
- Começar a fazer;
 - Pair programming.
- Parar de fazer;
 - Atraso nas verificações.
- Fazer menos.
 - Individual programming.

