Universidade São Judas Tadeu - Butantã - Noturno

Nomes:
Guilherme Sandoli- 82324873

Turma: GQS-CCP1AN-BUE1

Turma: GQS-CCP1AN-BUE1

ATIVIDADE 04

Teste de caixa branca e preta

Plano de testes -

1. Caso de Teste 1: Valor presente no vetor (caso básico)
    - Descrição: Verificar se o método encontra um valor presente no vetor.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 3`
    - Saída Esperada: `2` (índice do valor `3`)
2. Caso de Teste 2: Valor no início do vetor
    - Descrição: Verificar se o método encontra um valor que está no início do vetor.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 1`
    - Saída Esperada: `0` (índice do valor `1`)
3. Caso de Teste 3: Valor no final do vetor
    - Descrição: Verificar se o método encontra um valor que está no final do vetor.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 5`
    - Saída Esperada: `4` (índice do valor `5`)
4. Caso de Teste 4: Valor não presente no vetor (menor que o menor)
    - Descrição: Verificar se o método retorna -1 quando o valor não está presente e é menor que todos os elementos.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 0`
    - Saída Esperada: `-1`
5. Caso de Teste 5: Valor não presente no vetor (maior que o maior)
    - Descrição: Verificar se o método retorna -1 quando o valor não está presente e é maior que todos os elementos.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 6`
    - Saída Esperada: `-1`
6. Caso de Teste 6: Valor não presente no vetor (entre valores)
    - Descrição: Verificar se o método retorna -1 para um valor que não está presente, mas está entre dois elementos do vetor.
    - Entrada: `iVet = [1, 2, 3, 4, 5], ik = 3.5`
    - Saída Esperada: `-1`
7. Caso de Teste 7: Vetor vazio
    - Descrição: Verificar o comportamento do método quando o vetor é vazio.
    - Entrada: `iVet = [], ik = 1`
    - Saída Esperada: `-1`
8. Caso de Teste 8: Vetor com um elemento (presente)
    - Descrição: Verificar se o método encontra o valor quando o vetor contém apenas um elemento.
    - Entrada: `iVet = [5], ik = 5`
    - Saída Esperada: `0`
9. Caso de Teste 9: Vetor com um elemento (não presente)
    - Descrição: Verificar se o método retorna -1 quando o vetor contém um elemento que não é igual ao valor buscado.
    - Entrada: `iVet = [5], ik = 3`
    - Saída Esperada: `-1`
10. Caso de Teste 10: Vetor ordenado em ordem decrescente

- Descrição: Verificar o comportamento do método quando o vetor não está ordenado em ordem crescente.
- Entrada: `iVet = [5, 4, 3, 2, 1], ik = 3`
- Saída Esperada: `-1` (deve retornar -1, já que a busca binária não funciona corretamente em vetores não ordenados)

ROTEIRO DE TESTES -

Preparação do Ambiente

1. Configuração do Ambiente
   - Certifique-se de que você tem um ambiente de desenvolvimento configurado com a linguagem de programação que suporta a implementação do método de busca binária (Java, no caso).
2. Implementação do Método
   - Copie e cole a implementação do método `busca_binaria` em uma classe de teste.

Execução dos Casos de Teste

Caso de Teste 1: Valor presente no vetor

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 3)`.
2. Resultado Esperado: `2`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 2: Valor no início do vetor

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 1)`.
2. Resultado Esperado: `0`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 3: Valor no final do vetor

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 5)`.
2. Resultado Esperado: `4`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 4: Valor não presente (menor que o menor)

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 0)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 5: Valor não presente (maior que o maior)

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 6)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 6: Valor não presente (entre valores)

1. Passo: Chamar o método `busca_binaria([1, 2, 3, 4, 5], 3.5)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 7: Vetor vazio

1. Passo: Chamar o método `busca_binaria([], 1)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 8: Vetor com um elemento (presente)

1. Passo: Chamar o método `busca_binaria([5], 5)`.
2. Resultado Esperado: `0`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 9: Vetor com um elemento (não presente)

1. Passo: Chamar o método `busca_binaria([5], 3)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Caso de Teste 10: Vetor ordenado em ordem decrescente

1. Passo: Chamar o método `busca_binaria([5, 4, 3, 2, 1], 3)`.
2. Resultado Esperado: `-1`.
3. Verificar: Se a saída do método é igual ao resultado esperado.

Finalização do Teste

1. Registro dos Resultados: Documentar os resultados de cada caso de teste (se passou ou falhou).
2. Análise de Erros: Caso algum teste falhe, analisar o motivo e ajustar o método, se necessário.
3. Relatório Final: Criar um relatório resumindo a execução dos testes, incluindo casos que passaram, falharam e qualquer problema encontrado

PLANO DE TESTE - (EXERCICIO 02)

Casos de Teste

1. Caso de Teste 1: Login e senha corretos
    - Descrição: Verificar se o sistema autentica corretamente um usuário com login e senha válidos.
    - Entrada: Login: `usuario_valido`, Senha: `senha_valida`.
    - Saída Esperada: Mensagem “Login realizado com sucesso” e acesso liberado ao programa.

2. Caso de Teste 2: Login incorreto
    - Descrição: Verificar se o sistema apresenta a mensagem correta quando o login está incorreto.
    - Entrada: Login: `usuario_invalido`, Senha: `senha_valida`.
    - Saída Esperada: Mensagem “Login e/ou Senha incorretos”.

3. Caso de Teste 3: Senha incorreta
    - Descrição: Verificar se o sistema apresenta a mensagem correta quando a senha está incorreta.
    - Entrada: Login: `usuario_valido`, Senha: `senha_invalida`.
    - Saída Esperada: Mensagem “Login e/ou Senha incorretos”.

4. Caso de Teste 4: Login e senha vazios
    - Descrição: Verificar se o sistema trata adequadamente entradas vazias.
    - Entrada: Login: `""`, Senha: `""`.
    - Saída Esperada: Mensagem “Login e/ou Senha incorretos”.

5. Caso de Teste 5: Código de validação correto
    - Descrição: Verificar se o sistema autentica corretamente um usuário com código de validação correto.
    - Entrada: Código de validação: `codigo_correto`.
    - Saída Esperada: Mensagem “Login realizado com sucesso” e acesso liberado ao programa.

6. Caso de Teste 6: Código de validação incorreto
    - Descrição: Verificar se o sistema apresenta a mensagem correta quando o código de validação está incorreto.
    - Entrada: Código de validação: `codigo_incorreto`.

- Saída Esperada: Mensagem “Login não autorizado!”.

7. Caso de Teste 7: Código de validação vazio
   - Descrição: Verificar se o sistema trata adequadamente um código de validação vazio.
   - Entrada: Código de validação: `""`.
   - Saída Esperada: Mensagem “Login não autorizado!”.

8. Caso de Teste 8: Verificação de SMS enviado
   - Descrição: Verificar se o código de validação é enviado via SMS após a inserção de login e senha corretos.
   - Entrada: Login: `usuario_valido`, Senha: `senha_valida`.
   - Ação: Verificar registro de envio de SMS.
   - Saída Esperada: SMS enviado com o código de validação gerado.

9. Caso de Teste 9: Login e senha válidos, mas SMS falha
   - Descrição: Verificar o comportamento do sistema caso haja falha no envio do SMS.
   - Entrada: Login: `usuario_valido`, Senha: `senha_valida`.
   - Ação: Simular falha no envio do SMS.
   - Saída Esperada: Mensagem de erro indicando falha no envio do código de validação.

10. Caso de Teste 10: Acesso ao programa sem validação em duas etapas
    - Descrição: Verificar se o sistema não permite o acesso sem a validação em duas etapas.
    - Entrada: Login: `usuario_valido`, Senha: `senha_valida`, Código de validação: `""`.
    - Saída Esperada: Mensagem “Login não autorizado!”.

Considerações Finais

- Cada caso de teste deve ser executado em um ambiente controlado que simule o funcionamento do sistema.
- É importante documentar os resultados de cada teste, anotando qualquer falha ou comportamento inesperado.
- Um relatório final deve ser gerado com as conclusões dos testes, incluindo as ações corretivas, se necessário.

ROTEIRO DE TESTES -

.

Preparação do Ambiente

1. Configuração do Ambiente
    - Assegure-se de que o sistema de login esteja instalado e em execução.
    - Verifique se as credenciais de teste (login, senha e número de celular) estão previamente configuradas no sistema.
    - Certifique-se de que há um serviço de SMS configurado e funcionando para envio de códigos.

Execução dos Casos de Teste

Caso de Teste 1: Login e senha corretos

1. Ação: No campo “Login”, insira `usuario_valido`.
2. Ação: No campo “Senha”, insira `senha_valida`.
3. Ação: Clique no botão “Entrar”.
4. Verificar: Se a mensagem “Login realizado com sucesso” é exibida.
5. Verificar: Se o acesso ao programa é liberado.

Caso de Teste 2: Login incorreto

1. Ação: No campo “Login”, insira `usuario_invalido`.
2. Ação: No campo “Senha”, insira `senha_valida`.
3. Ação: Clique no botão “Entrar”.
4. Verificar: Se a mensagem “Login e/ou Senha incorretos” é exibida.

Caso de Teste 3: Senha incorreta

1. Ação: No campo “Login”, insira `usuario_valido`.
2. Ação: No campo “Senha”, insira `senha_invalida`.
3. Ação: Clique no botão “Entrar”.
4. Verificar: Se a mensagem “Login e/ou Senha incorretos” é exibida.

Caso de Teste 4: Login e senha vazios

1. Ação: No campo “Login”, insira `""`.
2. Ação: No campo “Senha”, insira `""`.
3. Ação: Clique no botão “Entrar”.
4. Verificar: Se a mensagem “Login e/ou Senha incorretos” é exibida.

Caso de Teste 5: Código de validação correto

1. Ação: Após o login bem-sucedido, no campo de código de validação, insira `codigo_correto`.
2. Ação: Clique no botão “Entrar”.
3. Verificar: Se a mensagem “Login realizado com sucesso” é exibida e o acesso ao programa é liberado.

Caso de Teste 6: Código de validação incorreto

1. Ação: Após o login, no campo de código de validação, insira `codigo_incorreto`.
2. Ação: Clique no botão “Entrar”.
3. Verificar: Se a mensagem “Login não autorizado!” é exibida.

Caso de Teste 7: Código de validação vazio

1. Ação: Após o login, no campo de código de validação, insira `""`.
2. Ação: Clique no botão “Entrar”.
3. Verificar: Se a mensagem “Login não autorizado!” é exibida.

Caso de Teste 8: Verificação de SMS enviado

1. Ação: Realize um login bem-sucedido com `usuario_valido` e `senha_valida`.
2. Ação: Verifique o registro do envio de SMS.
3. Verificar: Se o SMS foi enviado com o código de validação gerado.

Caso de Teste 9: Login e senha válidos, mas SMS falha

1. Ação: No campo “Login”, insira `usuario_valido`.
2. Ação: No campo “Senha”, insira `senha_valida`.
3. Ação: Simule uma falha no envio do SMS.
4. Ação: Clique no botão “Entrar”.
5. Verificar: Se uma mensagem de erro indicando falha no envio do código de validação é exibida.

Caso de Teste 10: Acesso ao programa sem validação em duas etapas

1. Ação: No campo “Login”, insira `usuario_valido`.

2. Ação: No campo “Senha”, insira `senha_valida`.
3. Ação: No campo de código de validação, insira `""`.
4. Ação: Clique no botão “Entrar”.
5. Verificar: Se a mensagem “Login não autorizado!” é exibida.

Finalização do Teste

1. Registro dos Resultados: Documentar os resultados de cada teste (se passou ou falhou).
2. Análise de Erros: Caso algum teste falhe, analisar o motivo e ajustar o sistema, se necessário.
3. Relatório Final: Criar um relatório resumindo a execução dos testes, incluindo casos que passaram, falharam e qualquer problema encontrado.

Observações

- Certifique-se de que todas as etapas foram seguidas e que o sistema esteja funcionando corretamente antes de iniciar os testes.
- Pode ser útil adicionar prints ou logs durante a execução dos testes para facilitar a identificação de problemas.