Técnica de teste que avalia o comportamento interno do componente de software trabalhando diretamente sobre o código-fonte deste componente, aplicando as seguintes estratégias.

- **Caminho de controle:** Esta estratégia busca testar todos os caminhos possíveis através do código, com o objetivo de garantir que as rotas de execução sejam verificadas, incluindo loops e ramificações. Isso ajuda a identificar falhas lógicas e garante que todas as partes do código sejam executadas durante o teste.

```
1       X = Integer.parseInt(args[0]);
2       y = Integer.parseInt(args[1]);
3       while(x < 0){
4               if(y < 0){
5                       y = y + 2
6               }
7               x = x + 1;
8       }
9       c = x + y;
```

CASO DE TESTE	
ENTRADA	COBERTURA
X = 3 Y = 1	57,14 %
X = -1 Y = -1	100 %

- **Teste de condição:** Neste caso, o testador realiza a verificação de todas as condições lógicas no código, como instruções if-else e switch-cases. Esta estratégia é responsável por certificar de que o código está se comportando corretamente em diferentes cenários e condições.

- **Teste de Loop:** Esta técnica é usada para testar as estruturas de loop no código, como for, while e do-while loops. Assim, é possível entender se as etapas estão operando corretamente principalmente em situações de limites de início e término do loop.

- **Cobertura de código:** Uma das principais estratégias no teste de caixa branca, é testar cada linha de código, garantindo que não haja partes não testadas do código.

- **Fluxo de dados:** Analisa como os dados estão sendo utilizados e manipulados dentro do código, verificando se todas as variáveis são inicializadas antes do uso, se existem variáveis que nunca são usadas e se os caminhos de dados são lógicos e eficientes.

- **Baseado em API:** Em casos de uso frequente de APIs, esta estratégia é responsável por testar as interfaces de programação dos aplicativos, permitindo uma interação adequada em parceria com o código principal.