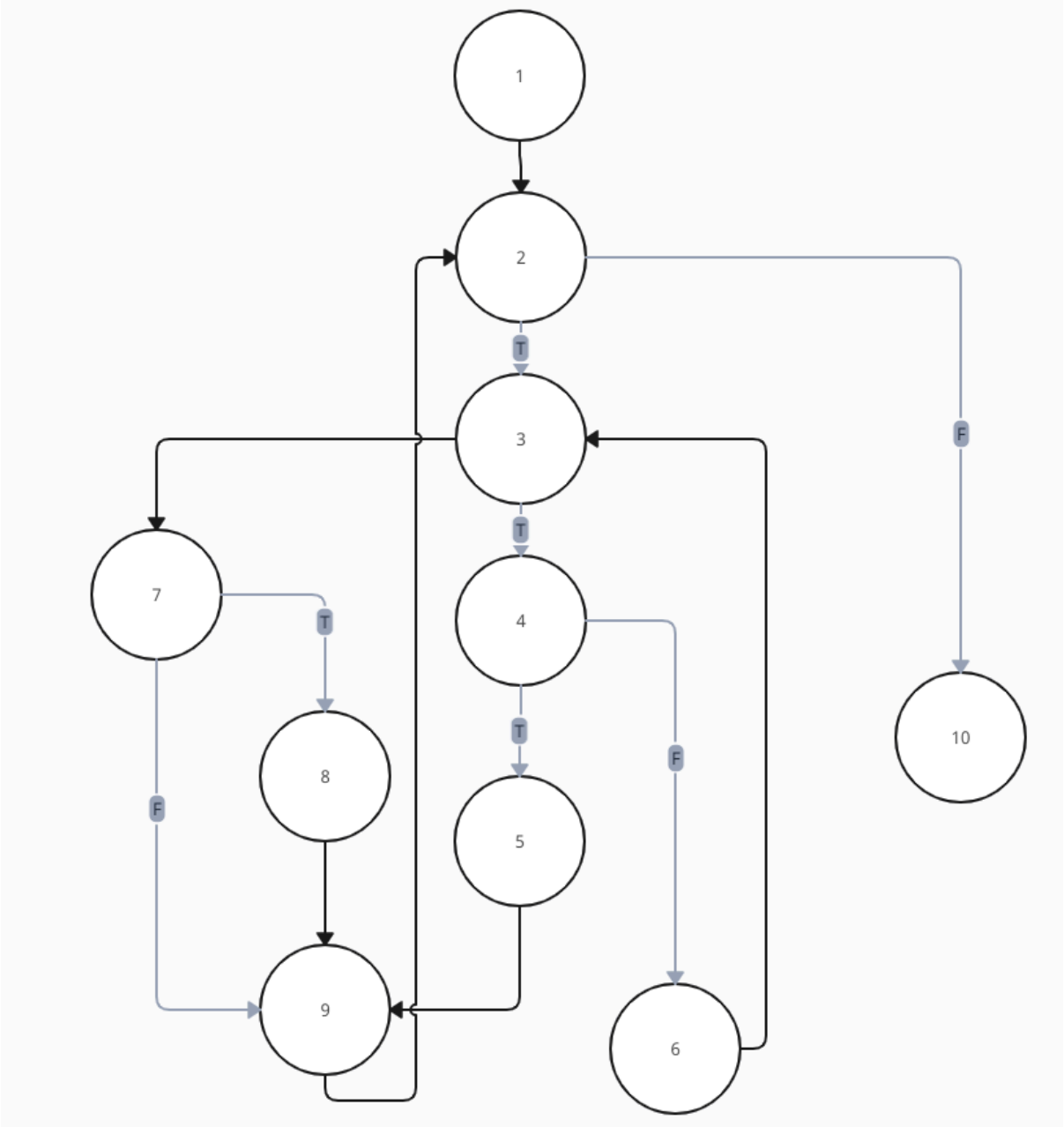
a)



b)

Podemos usar o conceito de complexidade ciclomática, que é **E - N + 2**, onde **E** é o número de arestas e **N** é o número de nós.

- Número de arestas (E): 10
- Número de nós (N): 8

Complexidade ciclomática = 10 - 8 + 2 = 4

Portanto, existem 4 caminhos independentes.

c)

Caminho 1: Entrada principal → ehPrimo chamada → if (num <= 1) verdadeiro → return false → Impressão do resultado

Caminho 2: Entrada principal → ehPrimo chamada → if (num <= 1) falso → Laço for não executa → return true → Impressão do resultado

Caminho 3: Entrada principal → ehPrimo chamada → if (num <= 1) falso → Laço for executa → if (num % i == 0) verdadeiro → return false → Impressão do resultado

Caminho 4: Entrada principal → ehPrimo chamada → if (num <= 1) falso → Laço for executa → if (num % i == 0) falso → return true → Impressão do resultado

d)

- Caminho 1: Número: 0 ou 1 -> Expectativa: Não é primo

- Caminho 2: Número: 2 ou qualquer número primo pequeno (3, 5, 7) ->
Expectativa: É primo

- Caminho 3: Número: 4 (divisível por 2) -> Expectativa: Não é primo

- Caminho 4: Número: 17 (ou qualquer número primo não pequeno, mas executa o laço) ->
Expectativa: É primo

e)

if (num <= 1);

for (int i = 2; i * i <= num; i++);

if (num % i == 0).

f)

1. if (num <= 1):

   - Caso num = 0 ou num = 1 (condição verdadeira)
   - Caso num > 1 (condição falsa)

2. for (int i = 2; i * i <= num; i++):

   - Caso num = 2 (não entra no laço)
   - Caso num > 2 (entra no laço)

3. if (num % i == 0):

- Caso num = 4 (entra no laço e condição verdadeira)
- Caso num = 17 (entra no laço e condição falsa)

g)

Considerando valores limite e os conceitos de números primos:

1. Limite Inferior:
   - num = 0 (não primo)
   - num = 1 (não primo)
   - num = 2 (primo, menor número primo)

2. Limite Superior Próximo (números pequenos):
   - num = 3 (primo)
   - num = 4 (não primo)

3. Valores intermediários e altos para verificar divisibilidade:
   - num = 17 (primo)
   - num = 18 (não primo, divisível por 2)

Portanto, em resumo, os casos de teste serão:

1. num = 0: Esperado: Não é primo
2. num = 1: Esperado: Não é primo
3. num = 2: Esperado: É primo
4. num = 3: Esperado: É primo
5. num = 4: Esperado: Não é primo
6. num = 17: Esperado: É primo
7. num = 18: Esperado: Não é primo

Estes testes cobrem todos os caminhos independentes, condições lógicas e análises de valor limite para garantir uma cobertura adequada do código.