

# Atividade Prática de Teste Estrutural

Alunos:

Bruno Abraham Valério

R.A.: 23000333-2

Henrique Pacheco Alves

R.A.: 23293915-2

Hugo Vinícius Fonseca Zuin

R.A.: 23000248-2

João Fernando Ehlers

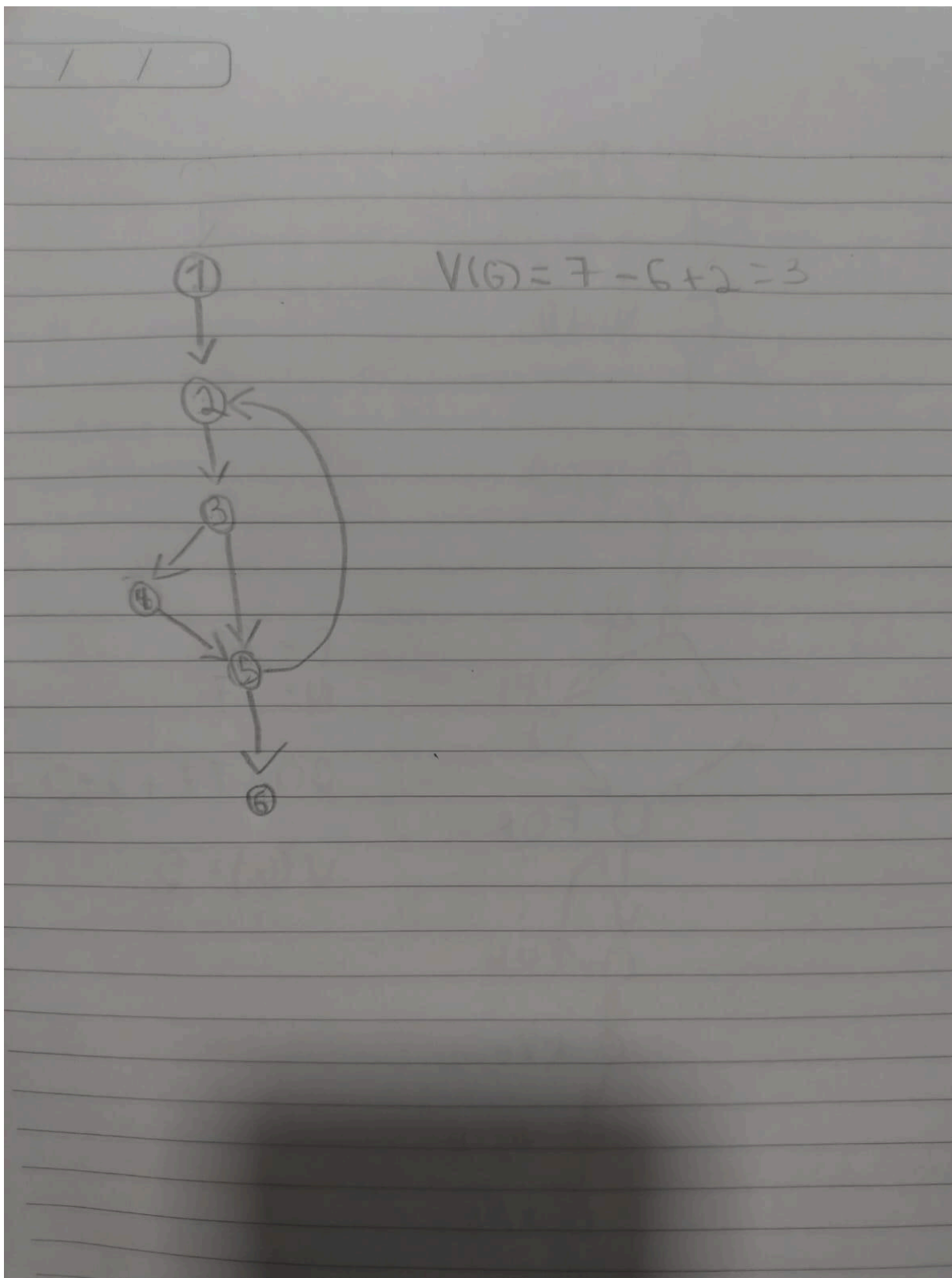
R.A.: 23317130-2

## a) Lista de divisores de um número

```
#include <stdio.h>
```

```
int main() {  
    int n;  
    printf("Digite um numero: ");  
    scanf("%d", &n);  
  
    printf("Divisores de %d: ", n);  
    for(int i = 1; i <= n; i++) {  
        if(n % i == 0) {  
            printf("%d ", i);  
        }  
    }  
    printf("\n");  
    return 0;  
}
```

**Grafo de fluxo e complexidade ciclomática:**



Caminhos:

Caminho 1:

1-2-3-4-5-6

Caminho 2:

1-2-3-5-6

Caminho 3:

1-2-3-4-5-2-3-4-5-6

### Casos de teste

- Entrada  $n=1 \rightarrow$  Saída: 1
- Entrada  $n=7 \rightarrow$  Saída: 1 7
- Entrada  $n=12 \rightarrow$  Saída: 1 2 3 4 6 12

---

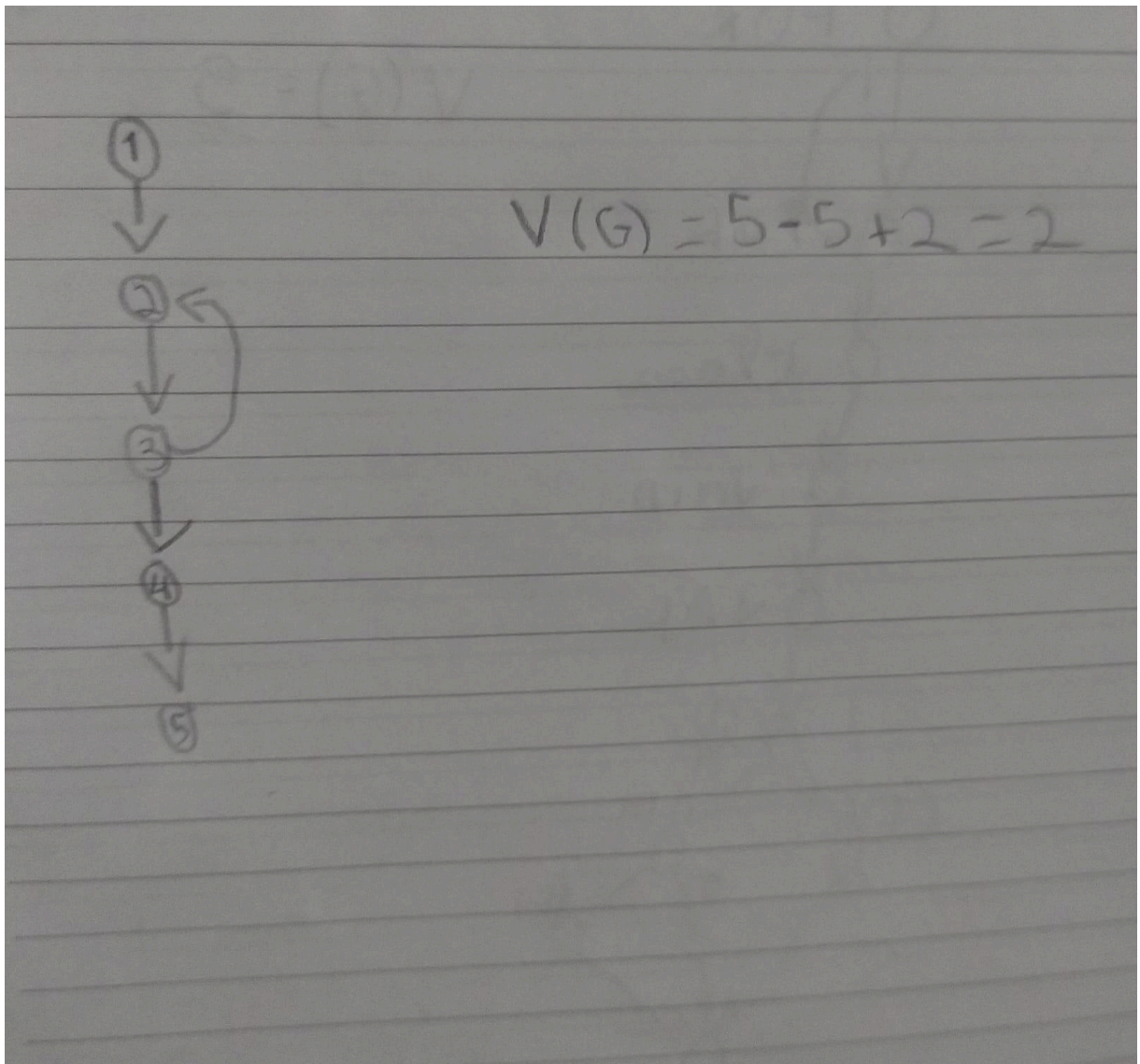
**b) Um algoritmo que lê dois números e calcula o máximo divisor comum pelo método de Euclides.**

```
#include <stdio.h>
```

```
int mdc(int a, int b) {  
    while(b != 0) {  
        int temp = b;  
        b = a % b;  
        a = temp;  
    }  
    return a;  
}
```

```
int main() {  
    int x, y;  
    printf("Digite dois numeros: ");  
    scanf("%d %d", &x, &y);  
  
    printf("MDC de %d e %d = %d\n", x, y, mdc(x, y));  
    return 0;  
}
```

**Grafo de fluxo e complexidade ciclomática:**



Caminhos:

caminho 1:

1-2-3-4-5

caminho 2:

1-2-3-2-3-4-5

### Casos de teste

- Entrada  $x=10, y=0 \rightarrow$  Saída: 10
- Entrada  $x=48, y=18 \rightarrow$  Saída: 6

---

c) Um algoritmo que lê as 4 notas de um aluno e diga se ele passou por média, está em final ou reprovou

```

#include <stdio.h>

int main() {
    float n1, n2, n3, n4, media;

    printf("Digite as 4 notas: ");
    scanf("%f %f %f %f", &n1, &n2, &n3, &n4);

    media = (n1 + n2 + n3 + n4) / 4.0;

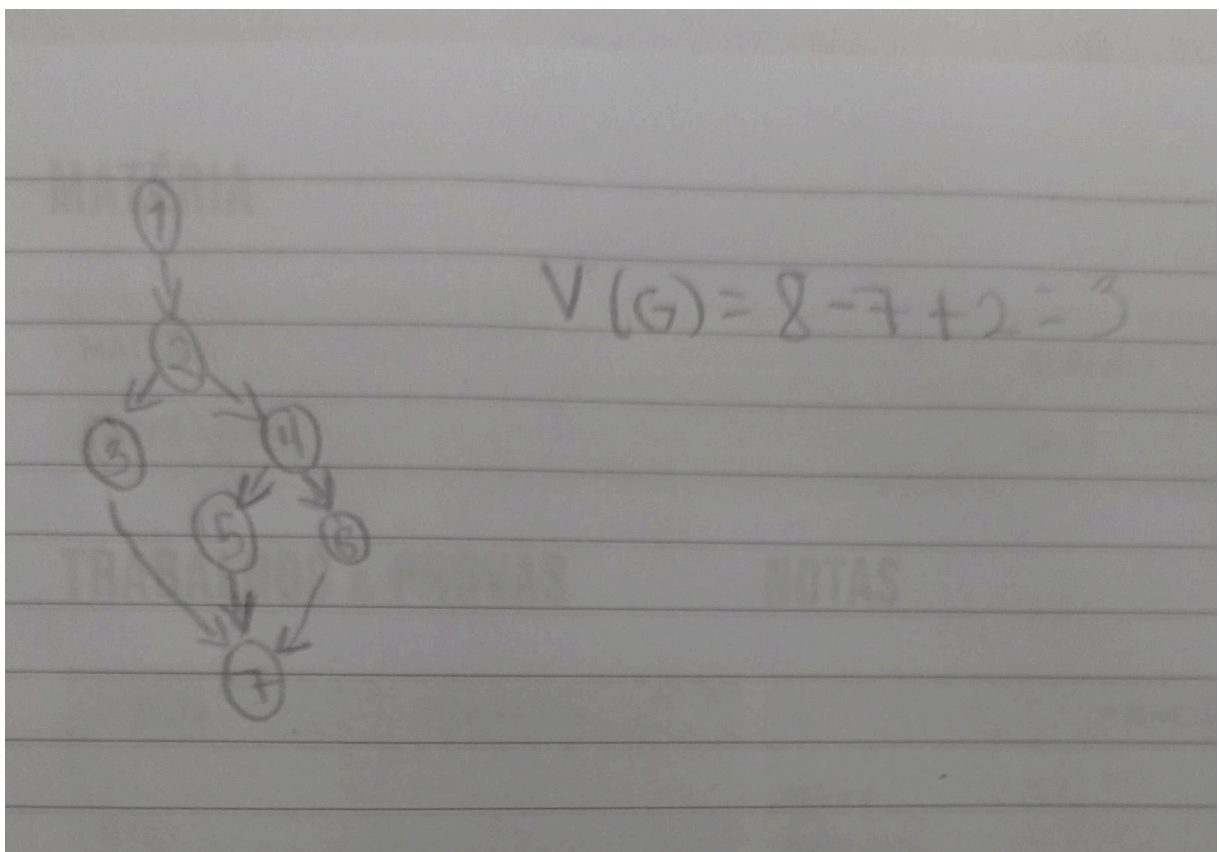
    printf("Media = %.2f\n", media);

    if(media >= 7.0)
        printf("Aluno aprovado!\n");
    else if(media >= 4.0)
        printf("Aluno em exame final!\n");
    else
        printf("Aluno reprovado!\n");

    return 0;
}

```

### Grafo de fluxo e complexidade ciclômática:



Caminhos:

caminho 1:

1-2-3-7

caminho 2:

1-2-4-5-7

caminho 3:

1-2-4-6-7

## Casos de teste

- Notas = 8, 9, 7, 10 → Média = 8.5 → Aprovado
- Notas = 6, 5, 4, 5 → Média = 5 → Final
- Notas = 2, 3, 1, 4 → Média = 2.5 → Reprovado

---

**d) Um algoritmo em que dado dois números  $n$  e  $k$  ( $n < k$ ), calcule e apresente a combinatória de  $n$  elementos tomados  $k$  a  $k$**

```
#include <stdio.h>
```

```
long long fatorial(int n) {  
    long long fat = 1;  
    for(int i = 2; i <= n; i++) {  
        fat *= i;  
    }  
    return fat;  
}
```

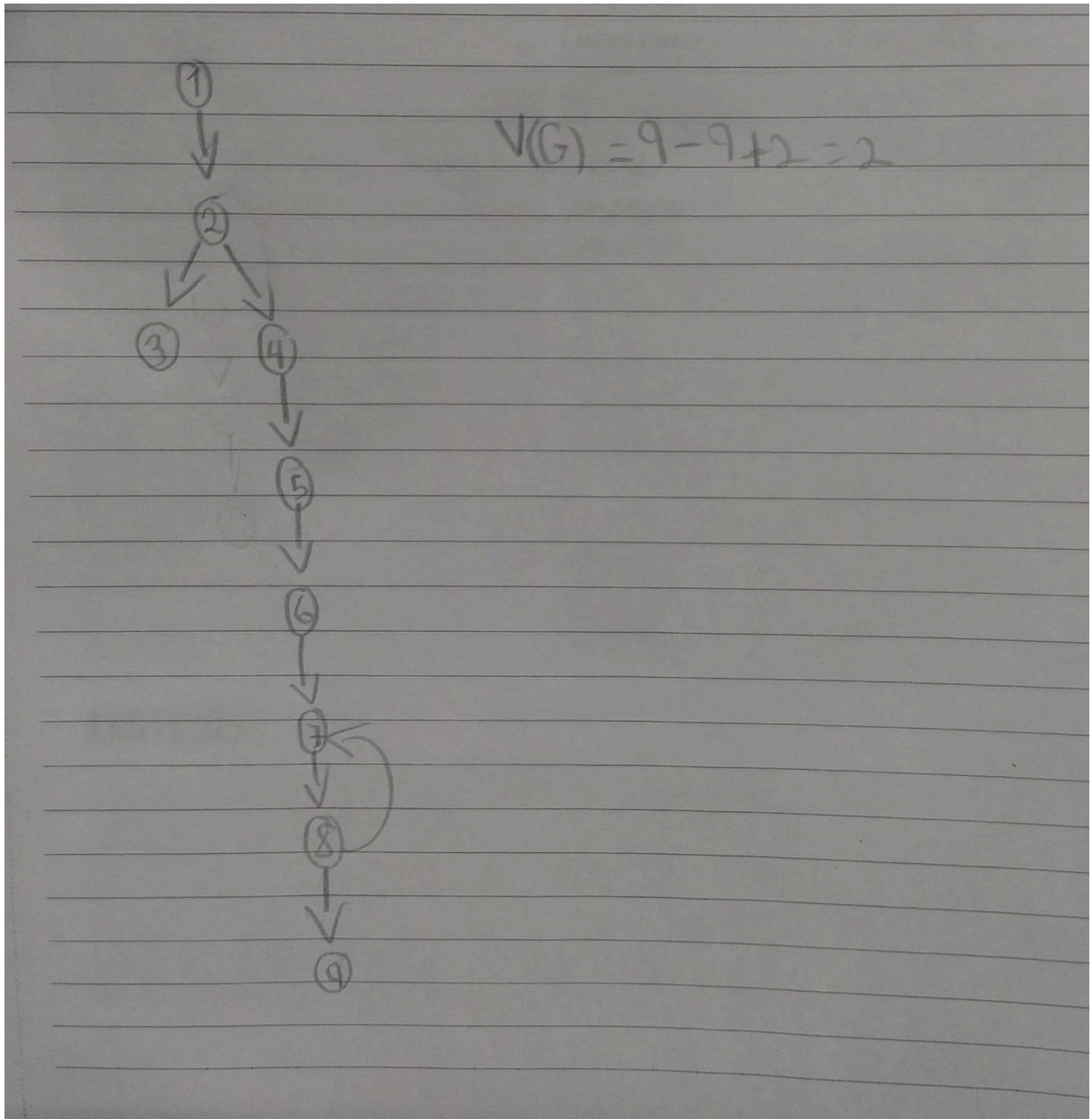
```
long long combinatoria(int n, int k) {  
    return fatorial(n) / (fatorial(k) * fatorial(n - k));  
}
```

```
int main() {  
    int n, k;  
    printf("Digite n e k (n >= k): ");  
    scanf("%d %d", &n, &k);  
  
    if(k > n) {  
        printf("Erro: k nao pode ser maior que n.\n");  
    } else {  
        printf("C(%d, %d) = %lld\n", n, k, combinatoria(n, k));  
    }
```



```
}  
  
return 0;  
}
```

**Grafo de fluxo e complexidade ciclômática:**



Caminhos:

caminho 1:

1-2-3

caminho 2:

1-2-4-5-6-7-8-9

caminho 3:

1-2-4-5-6-7-8-7-8-9

### Casos de teste

- Entrada  $n=3, k=5 \rightarrow$  Saída: erro
- Entrada  $n=5, k=2 \rightarrow$  Saída: 10
- Entrada  $n=10, k=5 \rightarrow$  Saída: 252

### PLANO DE TESTE

#### ITENS A TESTAR / ABORDAGEM:

Nº	Item	Especificação	Abordagem
1	Lista de divisores	validar saída para casos básicos, primos e compostos.	
2	MDC	validar execução com $y=0$ e entradas diferentes.	
3	Média de aluno	validar ramos (aprovado, final, reprovado).	
4	Combinatória	validar entrada inválida e casos válidos.	

#### CRONOGRAMA DE TESTES

ID	Tarefa	início	fim	esforço	pessoa	Obs
01	Criar casos de teste	Dia 1	Dia 1	2h	Aluno	-
02	Executar testes manuais	Dia 1	Dia 2	3h	Aluno	-
03	Validar saídas	Dia 2	Dia 2	1h	Aluno	-



**AMBIENTE DE TESTE**

Descrição	Descrição
Hardware	PC comum (Windows/Linux)
Software	GCC / IDE Code::Blocks ou VS Code
Ferramental	Compilador C, terminal

**IDENTIFICAÇÃO DE CASO DE TESTE / IDENTIFICAÇÃO DE PROCEDIMENTO DE TESTE**

Nº	Caso de teste	Identificação de caso de teste		Procedimento	Identificação do Procedimento do teste
1	Divisores simples	CT-01: $n = 1$		Executar o programa de divisores com entrada $n=1$ e verificar saída	PT-01
2	Divisores primo	CT-02: $n = 7$		Executar o programa de divisores com entrada $n=7$ e verificar saída (esperado: "1 7")	PT-02
3	Divisores composto	CT-03: $n = 12$		Executar o programa de divisores com entrada $n=12$ e verificar saída (esperado: "1 2 3 4 6 12")	PT-03
4	MDC com $y=0$	CT-04: $x=10, y=0$		Executar o programa de MDC com entrada (10,0) e verificar saída (esperado: "10")	PT-04
5	MDC normal	CT-05: $x=48, y=18$		Executar o programa de MDC com entrada (48,18) e verificar saída (esperado: "6")	PT-05

6	Aluno aprovado	CT-06: notas 8,9,7,10		Executar o programa de notas e verificar saída (esperado: "Aprovado")	PT-06
7	Aluno em final	CT-07: notas 6,5,4,5		Executar o programa de notas e verificar saída (esperado: "Final")	PT-07
8	Aluno reprovado	CT-08: notas 2,3,1,4		Executar o programa de notas e verificar saída (esperado: "Reprovado")	PT-08
9	Combinatória inválida	CT-09: n=3, k=5		Executar o programa de combinatória com entrada inválida (3,5) e verificar saída (esperado: "Erro: k não pode ser maior que n")	PT-09
10	Combinatória válida	CT-10: n=5, k=2		Executar o programa de combinatória com entrada (5,2) e verificar saída (esperado: "10")	PT-10
11	Combinatória válida alta	CT-11: n=10, k=5		Executar o programa de combinatória com entrada (10,5) e verificar saída (esperado: "252")	PT-11

#### CASO DE TESTE

Identificação	CT-A1	
Itens a Testar	Impressão correta dos divisores de um número.	
Entradas		
	Campo	Valor
	Número	12
Saídas Esperadas		
	Campo	Valor
	Divisores	1 2 3 4 6 12

Ambiente	Linguagem C, console interativo.
Procedimento	Executar o programa, inserir o número 12 e verificar a lista de divisores.
Dependência	Nenhuma.

#### PROCEDIMENTO DE TESTE

Identificação	PT-A1
Objetivo	Validar que o programa imprime corretamente os divisores de um número.
Requisitos	Programa compilado em C e rodando no terminal.
Fluxo	<p>Iniciar execução do programa.</p> <p>Inserir o número 12.</p> <p>Observar a saída.</p> <p>Conferir se a saída corresponde a 1 2 3 4 6 12.</p>

#### CASO DE TESTE

Identificação	CT-B1						
Itens a Testar	Cálculo correto do MDC entre dois números.						
Entradas	<table border="1"> <thead> <tr> <th>Campo</th><th>Valor</th></tr> </thead> <tbody> <tr> <td>Número 1</td><td>12</td></tr> <tr> <td>Número 2</td><td>8</td></tr> </tbody> </table>	Campo	Valor	Número 1	12	Número 2	8
Campo	Valor						
Número 1	12						
Número 2	8						
Saídas Esperadas	<table border="1"> <thead> <tr> <th>Campo</th><th>Valor</th></tr> </thead> <tbody> <tr> <td>MDC</td><td>4</td></tr> </tbody> </table>	Campo	Valor	MDC	4		
Campo	Valor						
MDC	4						
Ambiente	Linguagem C, console interativo.						
Procedimento	Executar o programa, inserir 12 e 8, e verificar o valor do MDC.						
Dependência	Nenhuma.						

#### PROCEDIMENTO DE TESTE

Identificação	PT-B1
Objetivo	Validar que o programa retorna o MDC correto usando o algoritmo de Euclides.
Requisitos	Programa compilado em C e rodando no terminal.

Fluxo	<p>Iniciar execução do programa.</p> <p>Inserir 12 e 8.</p> <p>Observar a saída.</p> <p>Conferir se o resultado é MDC = 4.</p>
-------	--

#### CASO DE TESTE

Identificação	CT-C1	
Itens a Testar	Cálculo correto da média e classificação do aluno.	
Entradas		
	Campo	Valor
	Nota 1	8
	Nota 2	7
	Nota 3	9
	Nota 4	10
Saídas Esperadas		
	Campo	Valor
	Média	8,5
	Resultado	Aprovado
Ambiente	Linguagem C, console interativo.	
Procedimento	Executar o programa, inserir as notas e verificar se a média e o resultado estão corretos.	
Dependência	Nenhuma.	

#### PROCEDIMENTO DE TESTE

Identificação	PT-C1
Objetivo	Validar se o programa classifica corretamente o aluno de acordo com a média.
Requisitos	Programa compilado em C e rodando no terminal.
Fluxo	<p>Iniciar execução do programa.</p> <p>Inserir notas 8, 7, 9, 10.</p> <p>Observar a saída.</p> <p>Conferir se a saída indica média 8,5 e resultado Aprovado.</p>

#### CASO DE TESTE

Identificação	CT-D1	
Itens a Testar	Cálculo correto da combinação de n elementos tomados k a k.	
Entradas		
	Campo	Valor
	n	5
	k	2
Saídas Esperadas		
	Campo	Valor
	C(5,2)	10
Ambiente	Linguagem C, console interativo.	
Procedimento	Executar o programa, fornecer n = 5 e k = 2, e verificar o resultado.	
Dependência	Nenhuma.	

#### PROCEDIMENTO DE TESTE

Identificação	PT-D1
Objetivo	Validar se o programa calcula corretamente combinações.
Requisitos	Programa compilado em C e rodando no terminal.
Fluxo	Iniciar execução do programa.
	Inserir valores n = 5, k = 2.
	Observar a saída.
	Conferir se a saída é 10.