**( a )**

**int letraA(int n, int a) {**

**int i; 1**

**float resultado = 0, aux; 1**

**for (i = 1; i <= n; i++) n + 1**

**{**

**aux = i / pow(a, i); n**

**resultado = resultado + aux; n**

**}**

**return resultado; 1**

**}**

**T(n) = 1 + 1 + (n + 1) + n + n + 1**

**T(n) = (n+1) + (n+1) + (n + 1) + 1**

**T(n) = 3(n+1) + 1 ou T(n) = 3n +4**

**=> O(n)**

**( b )**

**int letraB(int n, int m) {**

**int i, j; 1**

**int resultado = 0; 1**

**for(i = 1; i <= n; i++) n + 1**

**{**

**for(j = 1; j <= m; j++) n . (m + 1)**

**{**

**resultado = resultado + (i \* j); n . m**

**}**

**}**

**return resultado; 1**

**}**

**T(n,m) = 1 + 1 + (n+1) + [n.(m+1)] + (n.m) + 1**

**T(n,m) = [n.m + n] + (n.m) + n + 4**

**T(n,m) = 2n.m + 2n + 4**

**=> O(n.m)**

**( c )**

**int letraC(int n) {**

**int i; 1**

**for(i = 2; i < n; i++) n – 1**

**{**

**if(n == 2 || n == 1) n – 1**

**return 1; 0**

**if(n % i != 0) n – 1**

**continue; n – 2**

**else n – 1**

**return 0; 0**

**}**

**return 1; 1**

**}**

Os Piores casos são os números primos, onde “if(n % i != 0)” permite que entrem n-2 vezes

|  |  |  |
| --- | --- | --- |
| **N** | **#vezes** | **Resultado** |
| 1 | Não chega | ---------- |
| 2 | Não chega | ---------- |
| 3 | 1 | n-2 |
| 4 | 1 | n-3 |
| 5 | 3 | n-2 |
| 6 | 2 | n-4 |
| 7 | 5 | n-2 |
| 8 | 4 | n-4 |
| 9 | 6 | n-3 |
| 10 | 6 | n-4 |
| 11 | 9 | n-2 |
| 12 | 6 | n-6 |
| 13 | 11 | n-2 |

**T(n) = 1 + (n-1) + (n-1) + (n-1) + (n-2) + (n-2) + (n-1) + 1**

**T(n) = 4(n-1) + (n-2) + 2**

**T(n) = 4n – 4 + n – 2 + 2**

**T(n) = 5n – 4**

**=> O(n)**

**( d )**

**void letraD(int n, int m, int A) {**

**int resultado = 0, soma = 0; 1**

**for(j = 1; j <= m; j++) m + 1**

**{**

**for(i = 1; i <= n; i++) m . (n + 1)**

**{**

**soma = soma + A[i][j]; n**

**}**

**if(resultado == 0) m . m**

**{**

**resultado = soma; 0**

**}**

**else m . m**

**{**

**if(soma < resultado) m . m**

**{**

**resultado = soma; m . m**

**}**

**}**

**soma = 0; m . m**

**}**

**return resultado; 1**

**}**

**T(n) = 1 + m + 1 + [m.(n+1)] + n + m² + 0 + m² + m² + m² +m² + 1**

**T(n) = 3 + m + m.n + m + n + 5m²**

**T(n) = 5m² + m.n + 2m + n + 3**

* **O(m.n)**

**( e )**

void letraE(int n, int m, int B) {

int resultado = 0, produto = 1; 1

for(i = 1; i <= n; i++) n + 1

{

for(j = 1; j <= m; j++) n . (m+1)

{

produto = produto \* B[i][j]; n . m

}

if(soma > resultado) n

{

resultado = B[i][j]; n

}

soma = 0; n

}

return resultado; 1

}

**T(n) = 1 + n + 1 + [n.(m+1)] + n.m + n + n + n + 1**

**T(n) = 3 + 4n + n.m + n + n.m**

**T(n) = 2n.m + 5n + 3**

**=> O(n.m)**

void letraF(int n, int m, int C) {

int resultado = 0, produto = 1; 1

for(i = 1; i <= n; i++) n + 1

{

for(j = 1; j <= n; j++) n . (n+1)

{

if(i < j) n . n

{

soma = soma + B[i][j]; (n² - n) / 2

}

}

}

return resultado; 1

}

T**(n) = 1 + n + 1 + [n.(n+1)] + n² + [(n²-n)/2] + 1**

|  |  |  |
| --- | --- | --- |
| i | j | #vezes |
| 1 | 1...n | 0 |
| 2 | 1...n | 1 |
| 3 | 1...n | 2 |
| 4 | 1...n | 3 |
| 5 | 1...n | 4 |
| ... | ... | ... |
| n | 1...n | n-1 |
| = 𝑛2−𝑛 2 | | |

**T(n) = 3 + n + n² + n + n² + [(n²-n)/2] T(n) = 3 + 2n² + 2n + [(n²-n)/2]**

**T(n) = 3 + [(n²+2n)/2] + [(n²-n)/2]**

**T(n) = 2. [(n²-n)/2] + 3**

**T(n) = n² - n + 3**

**=> O(n²)**