|  |  |  |
| --- | --- | --- |
| *Nome do Aluno:*  *Nome do Aluno:* | | *Matrícula:*  *Matrícula:* |
| *Turma:* 2G | *Período:* 2º | *Data: 21/11/2021* |

**Questão 1 (2,0 pontos)** *Considere o seguinte código em Python. Observe que os números das linhas estão incluídos à esquerda.*

1 def pot(b, p):

2 y = b \*\* p

3 return y

4

5 def quadrado(x):

6 a = pot(x, 2)

7 return a

8

9 n = 5

10 resultado = quadrado(n)

11 print(resultado)

*Quais dos seguintes itens melhor reflete a ordem em que as linhas de código são processadas em Python?*

1. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
2. 1, 2, 3, 5, 6, 7, 9, 10, 11
3. 1, 5, 9, 10, 6, 2, 3, 7, 11
4. 9, 10, 11, 1, 2, 3, 5, 6, 7
5. 9, 10, 5, 6, 1, 2, 3, 7, 11

**Questão 2 (2,0 pontos)** Considere o vetor com 11 elementos abaixo e diga quantas comparações de igualdade realizam os algoritmos de Busca Linear Sequencial e Busca Binária, na tentativa de encontrar no vetor [1, 2, 7, 15, 23, 56, 57, 58, 70, 72, 78] os valores:

|  |  |
| --- | --- |
| a) 3 | Resposta: |
| b) 25 | Resposta: |
| c) 70 | Resposta: |

**Questão 3 (3,0 pontos)** A busca binária (em [inglês](https://pt.wikipedia.org/wiki/L%C3%ADngua_inglesa) *binary search algorithm*) é um [algoritmo de busca](https://pt.wikipedia.org/wiki/Algoritmo_de_busca) em [vetores](https://pt.wikipedia.org/wiki/Vetor) que segue o paradigma de [divisão e conquista](https://pt.wikipedia.org/wiki/Divis%C3%A3o_e_conquista), ou seja, “***dividir para conquista***”. Considere o seguinte vetor ordenado: 61, 174, 197, 217, 309, 448, 503, 557, 629, 701, 831. Para cada iteração do algoritmo de **busca binária iterativa**, tabule os valores das variáveis L, R, m e vetor v[m] para buscar os elementos 61 e 503:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Busca x = 61 | | | | | | Iteração | L | R | m | v[m] | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Busca x = 503 | | | | | | Iteração | L | R | m | v[m] | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |

**Questão 4 (3,0 pontos)** Escreva uma função que receba uma matriz quadrada A (nxn) de números inteiros positivos. Sua função deverá calcular e retornar a soma dos elementos da área hachurada (área mais escura do desenho). A Figura abaixo mostra um exemplo de uma matriz A (12x12), cuja soma da área rachurada é 316. OBS: Resolva o problema usando laço for aninhado.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **3** | **2** | **4** | **3** | **1** | **6** | **4** | **8** | **6** | **7** | **2** | **3** |
| **1** | **2** | **1** | **3** | **4** | **2** | **4** | **3** | **5** | **6** | **4** | **1** |
| **2** | **5** | **4** | **8** | **7** | **5** | **2** | **3** | **3** | **7** | **5** | **1** |
| **0** | **2** | **3** | **4** | **1** | **7** | **8** | **0** | **3** | **3** | **4** | **0** |
| **1** | **5** | **3** | **7** | **9** | **4** | **4** | **3** | **9** | **3** | **4** | **1** |
| **1** | **2** | **6** | **2** | **2** | **4** | **3** | **5** | **2** | **2** | **5** | **0** |
| **4** | **5** | **1** | **2** | **8** | **6** | **7** | **0** | **1** | **5** | **5** | **3** |
| **2** | **4** | **6** | **3** | **2** | **9** | **3** | **6** | **2** | **5** | **3** | **2** |
| **0** | **0** | **1** | **1** | **7** | **3** | **3** | **4** | **9** | **6** | **5** | **3** |
| **1** | **4** | **9** | **1** | **2** | **5** | **7** | **9** | **4** | **4** | **4** | **2** |
| **2** | **1** | **3** | **9** | **3** | **6** | **3** | **4** | **5** | **6** | **9** | **0** |
| **3** | **6** | **1** | **1** | **2** | **7** | **9** | **3** | **9** | **6** | **4** | **1** |