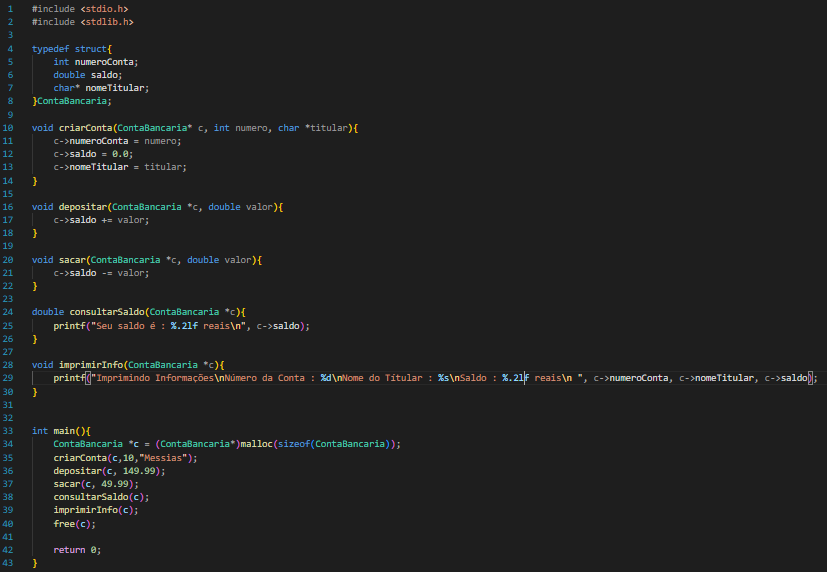
**Lista de Laboratório de Programação II**

**Nome :** Messias Feres Curi Melo

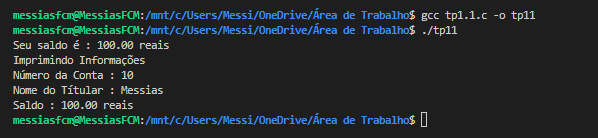
**Matrícula** : 2022003764

**Questão 1.1**

Código em C:

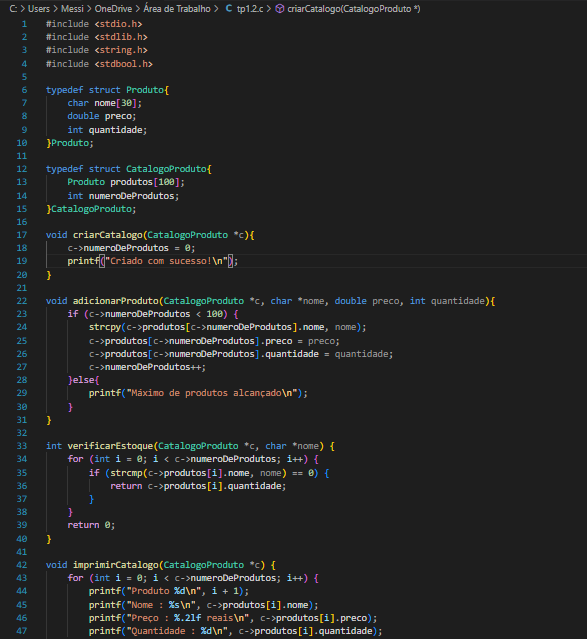


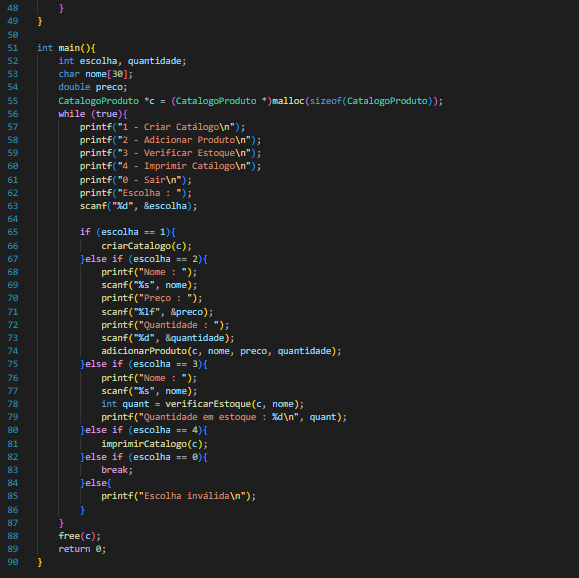
Exibição no terminal:



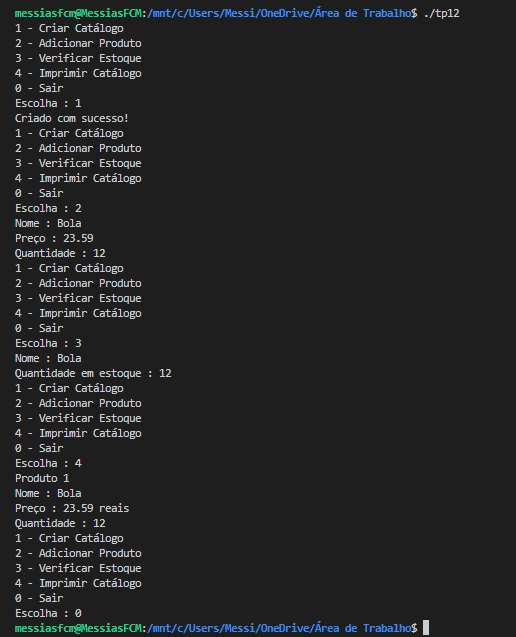
**Questão 1.2**

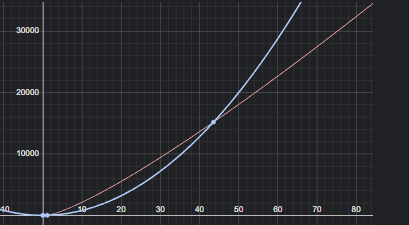
Código em C





Exibição no terminal:



**Questão 2.1**

8(n^2) <= 64n lg(n)

(n^2) <= 8n lg(n)

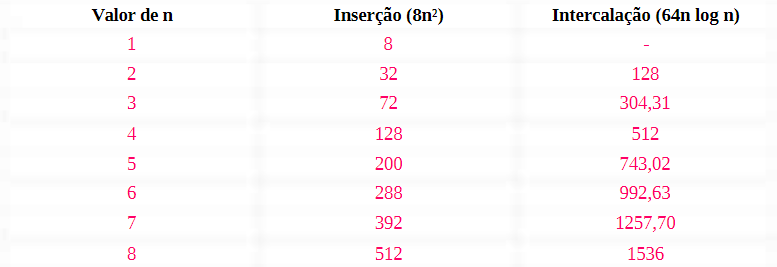
n <= 8 lg(n)

n/(lg(n)) <= 8

**n <= 43**

**Tabela e gráfico abaixo para demonstração.**

****

****

**…**

****

**Questão 2.2**

(100n^2) <= (2^n)

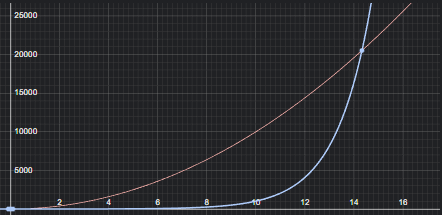
lg(100n^2) <= n

lg(100) + 2lgn <= n

6.65 + 2lgn <= n

**n <= 14.325**

**Resposta**: quando **n > 14.325**, o polinomial será melhor que o exponencial, gráfico abaixo para demonstração





**Questão 2.3**

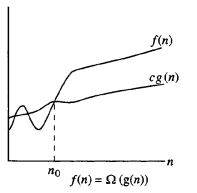
Formalmente, a função **O(f(n))** é um **limitante assintótico superior** para g(n).

Para todos os valores de n à direita de n0, o valor de g(n) reside em c·f(n) ou abaixo desse.

**Questão 2.4**

**g(n)** é **Ω(f(n))** significa que **g(n)** é assintoticamente **limitada inferiormente** por uma função f(n) multiplicada por uma constante positiva, para valores suficientemente grandes de n.

Para todos os valores de n à direita de n0, o valor de g(n) reside em c·f(n) ou acima desse.



**(Imagem com o f invertido com o g)**

**Questão 2.5**

Pois a notação O é usada para calcular um **limite superior** ou uma cota superior assintótica em relação ao crescimento do tempo de execução de um algoritmo à medida que a entrada (n) aumenta e **não medir um mínimo** ou no caso um limite inferior.

**Questão 2.6**

n^2-n+500 < 47n+47

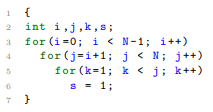
n^2-48n+453 < 0

delta = 2304 - 1812 = 492(raiz)

n1 = (48+22)/2 => **35**

n2 = (48-22)/2 => **13**

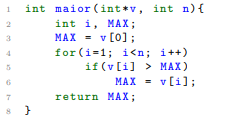
Então o algoritmo A leva menos tempo que B quando ‘n’ está próximo do intervalo de **13 < n < 35**, sendo o A mais eficiente nesse intervalo.



**Questão 2.7**

A complexidade será O(n^3), por ter 3 for aninhados essa será a complexidade do pior caso.

**Questão 2.8**



**(v[i] > MAX)**

**O(n)** - o algoritmo é **O(n)**, já que seu pior caso, ele cresce linearmente com o tamanho da entrada.

**Ω(n)** - o algoritmo é **Ω(n)**, já que seu melhor caso, também cresce linearmente com o tamanho da entrada.

**Θ(n)** - o algoritmo possui uma complexidade de tempo linear em todos os casos. Como já mostrado, o algoritmo é O(n) e Ω(n), o que significa que ele é **Θ(n)**.