Prova I

Nome: Lucas Machado da Silva Santana_____ data.: 06/02/2022

Professor.:Ronilson

1) No exemplo abaixo temos a implementação em Python do problema Torre de Hannoi de forma recursiva para formula $T_{(n)}= 2.T_{(n-1)} + 1$.

```
import sys
class Pilha:
   def __init__(self, capacidade):
      self.capacity = capacidade
      self.top = -1
      self.array = [0]*capacidade
def cria_pilha(capacidade):
   pilha = Pilha(capacidade)
   return pilha
def cheia(pilha):
   return (pilha.top == (pilha.capacity - 1))
def vazia(pilha):
   return (pilha.top == -1)
def push(pilha, item):
   if(cheia(pilha)):
      return
   pilha.top+=1
   pilha.array[pilha.top] = item
def Pop(pilha):
   if(vazia(pilha)):
      return -sys.maxsize
   Top = pilha.top
   pilha.top-=1
   return pilha.array[Top]
def moverDiscosEntreDoisArcos(src, dest, a, b):
   Arco1DiscoSup = Pop(src)
   Arco2Discosup = Pop(dest)
   if (Arco1DiscoSup == -sys.maxsize):
      push(src, Arco2Discosup)
      moverDisco(b, a, Arco2Discosup)
   elif (Arco2Discosup == -sys.maxsize):
      push(dest, Arco1DiscoSup)
      moverDisco(a, b, Arco1DiscoSup)
   elif (Arco1DiscoSup > Arco2Discosup):
      push(src, Arco1DiscoSup)
      push(src, Arco2Discosup)
      moverDisco(b, a, Arco2Discosup)
   else:
```

```
        push(dest, Arco2Discosup)
        push(dest, Arco1DiscoSup)
        moverDisco(a, b, Arco1DiscoSup)
def moverDisco(fromPin, toPin, disco):
    print("Mova o disco", disco, "do '", fromPin, "' para'", toPin, "'")
def Iterativo(num_de_discos, src, aux, dest):
    a, b, c = 'A', 'B', 'C'
    if (num_de_discos % 2 == 0):
        temp = b
        b = c
        c = temp
    total_movimentos = int(pow(2, num_de_discos) - 1)
    for i in range(num_de_discos, 0, -1):
        push(src, i)
    for i in range(1, total_movimentos + 1):
        if (i % 3 == 1):
            moverDiscosEntreDoisArcos(src, dest, a, b)
        elif (i % 3 == 2):
            moverDiscosEntreDoisArcos(src, aux, a, c)
        elif (i % 3 == 0):
            moverDiscosEntreDoisArcos(aux, dest, c, b)
num_de_discos = int(input("Quantos discos deseja considerar? \n -->"))
print("Solução do Problema (",2 ** num_de_discos - 1,"jogadas): \n")
src = cria_pilha(num_de_discos)
dest = cria_pilha(num_de_discos)
aux = cria_pilha(num_de_discos)
Iterativo(num_de_discos, src, aux, dest)
```

```
def hanoi(n, orig, dest, aux):
  if n == 1:
    print("Mover de",orig,"para",dest)
else:
    hanoi(n-1, orig, aux, dest)
print("Mover de",orig,"para",dest)
hanoi(n-1, aux, dest, orig)

 def main():  n = int(input("Digite a quantidade
de discos: "))  hanoi(n, "A", "B", "C")

main()
```

Implemente o algoritmo de Torre de Hannoi em python na formula direta $T_{(n)}= 2^n -1$.

2) Usando a definição formal de Big O prove que $6n^3 \neq O(n^2)$ e que lg_n não é Ω (n)

$6n^3 \neq O(n^2)$

C=10

N	6n^3	<=	C * N^2	
1	216	>	10	
2	1728	>	40	
3	5832	>	90	
4	13824	>	160	
5	27000	>	250	
100	216.000.000	>	100.000	

$lg_n \neq \Omega$ (n)

C= 0,2

N	lgN	>=	C * N	
1	0	<	0,2	
2	0,301	<	0,4	
3	0,477	<	0,6	
4	0,602	<	0,8	
5	0,698	<	1	
100	2	<	20	
1000	3	<	200	

3) Demonstre com exemplo para n^2 analise assintótica para Ω,Θ, Big O

Tabela	Assintótica	Para f(n)=	N^2
	Ω	Θ	Big O
N!	não	não	sim
2^N	não	não	sim
N^2	sim	sim	sim
N LOG N	sim	não	não
n	sim	não	não
Log n	sim	não	não
1	sim	não	não

C1=1
C2=3

N	C1 *N!	<=	N^2	<=	C2 * N!
1	1	<	1	<	3
2	2	<	4	<	6
3	6	<	9	<	18
3	24	>	16	<	72
5	120	>	25	<	360
10	3628800	>	100	<	10.886.400

C1=1
C2=3

N	C1 *2^N	<=	N^2	<=	C2 * 2^N
1	2	<	1	<	6
2	4	=	4	<	12
3	8	<	9	<	24
4	16	=	16	<	48
5	32	>	25	<	96
10	1024	>	100	<	3.072

C1=1
C2=3

N	C1 *N^2	<=	N^2	<=	C2 * N^2
1	1	=	1	<	3
2	4	=	4	<	12
3	9	=	9	<	27
4	16	=	16	<	48
5	25	=	25	<	75
10	100	=	100	<	300

C1=1
C2=3

N	C1 * n logn	<=	N^2	<=	C2 * n logn
1	0	<	1	>	0
2	0,602	<	4	>	1,806
3	1,431	<	9	>	4,293
4	2,408	<	16	>	7,224
5	3,494	<	25	>	10,482
10	10	<	100	>	30

C1=1
C2=3

N	C1 * n	<=	N^2	<=	C2 * n
1	1	<	1	<	3
2	2	<	4	<	6
3	3	<	9	=	9
4	4	<	16	>	12
5	5	<	25	>	15
10	10	<	100	>	30
100	100	<	10000	>	300

C1=1
C2=3

N	C1 * logn	<=	N^2	<=	C2 * logn
1	0	<	1	>	0
2	0,301	<	4	>	0,903
3	0,477	<	9	>	1,431
4	0,602	<	16	>	1,806
5	0,698	<	25	>	2,094
10	1	<	100	>	3
100	2	<	10000	>	6

C1=1
C2=3

N	C1 * 1	<=	N^2	<=	C2 * 1
1	1	=	1	<	3
2	1	<	4	>	3
3	1	<	9	>	3
4	1	<	16	>	3
5	1	<	25	>	3
10	1	<	100	>	3
100	1	<	10000	>	3

4) Considere as seguintes funções e coloque as funções em ordem de crescimento assintótico demonstrando graficamente.

a. log n

b. 2^n

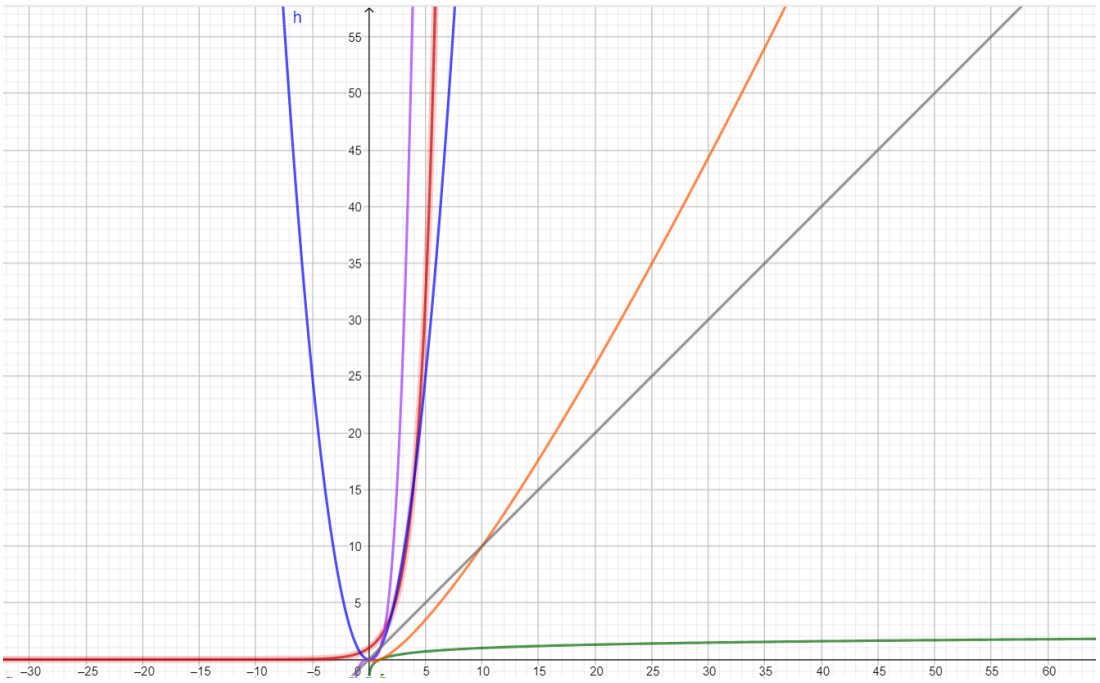
c. n^2

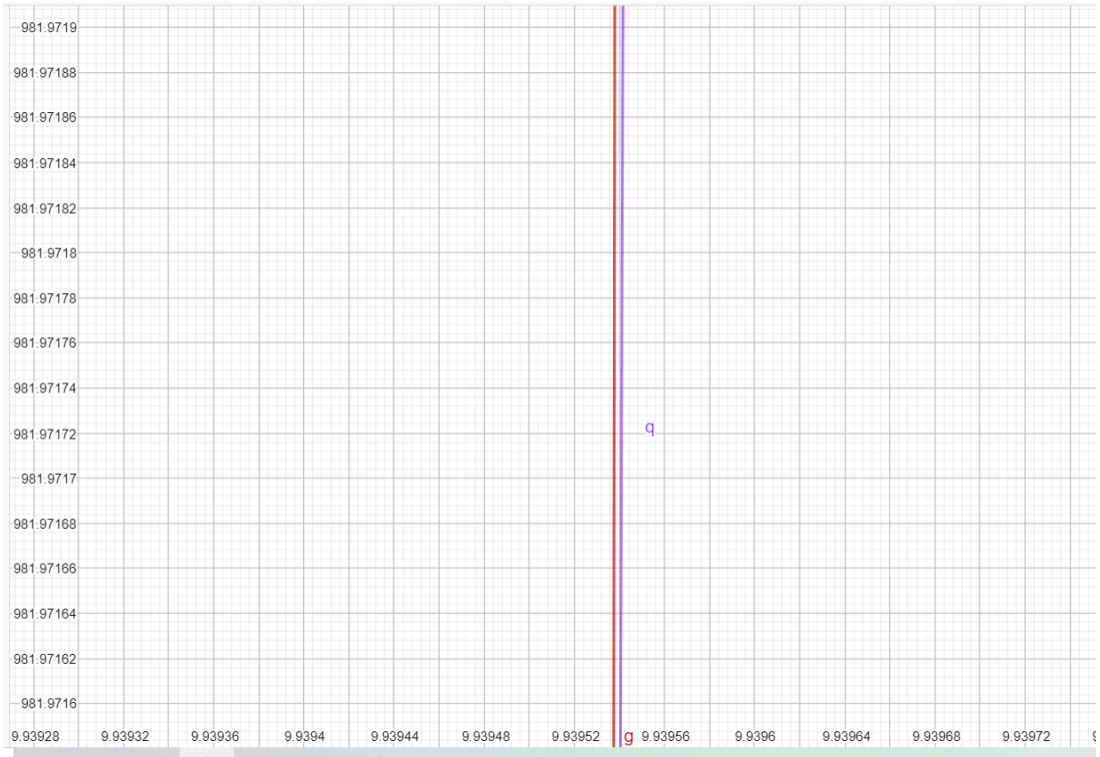
*d. n **

e. n^3

g. n

R: $2^n > n^3 > n^2 > n \log n > n > \log n$





	$f(x) = \log_{10}(x)$
	$g(x) = 2^x$
	$h(x) = x^2$
	$p(x) = x \log_{10}(x)$
	$q(x) = x^3$
	$r(x) = x$