Resolução de problemas de natureza discreta.

Atividade Avaliativa Recursão

Nome: Guilherme Santos de Oliveira

Número do Grupo: 10

Entregar um PDF com os enunciados e as devidas soluções(link do git e replit) para cada questão.
Links:
Github: https://github.com/GuiMarolly/Tarefa_Avalia-o_RA02
Replit: https://replit.com/@osantos40/TarefaAvaliacaoRA02

A Interpretação das questões é parte da avaliação.

Coloque os conjuntos de teste nas soluções.

As questões só serão validas se estiverem corretas por completo.

OBS: Fiz algumas observações no código em realação ao que já foi aprendido sobre python no semestre passado.

1. Escreva um algoritmo recursivo para cada uma das alternativas (2,25).

a) S(1) = 10

S(n) = S(n – 1) + 10, para n ≥ 2

```
def S(n):
    if n == 1:
        return 10
    else:
        return S(n - 1) + 10
```

b) A(1) = 2

A(n) = A(n – 1)-1 , para n ≥ 2

```
def A(n):
    if n == 1:
        return 2
    else:
```

```
        return A(n - 1) - 1
```

c) B(1) = 1

B(n) = B(n – 1) + n2, para n ≥ 2

```
def B(n):
    if n == 1:
        return 1
    else:
        return B(n - 1) + n**2
```

d) P(1) = 1

P(n) = n2*P(n – 1) + n - 1, para n ≥ 2

```
def P(n):
    if n == 1:
        return 1
    else:
        return n**2 * P(n - 1) + n - 1
```

e) D(1) = 3

D(2) = 5

D(n) = (n – 1)*D(n – 1) + (n – 2)*D(n – 2), para n > 2

```
def D(n):
    if n == 1:
        return 3
    elif n == 2:
        return 5
    else:
        return (n - 1) * D(n - 1) + (n - 2) * D(n - 2)
```

f) W(1) = 2

W(2) = 5

W(n) = W(n – 1)*W(n – 2), para n > 2

```
def W(n):
    if n == 1:
        return 2
    elif n == 2:
        return 5
    else:
        return W(n - 1) * W(n - 2)
```

g) T(1) = 1

T(2) = 2

T(3) = 3

T(n) = T(n – 1) + 2*T(n – 2) + 3*T(n – 3), para n > 3

```
def T(n):
    if n == 1:
        return 1
    elif n == 2:
        return 2
    elif n == 3:
        return 3
    else:
        return T(n - 1) + 2 * T(n - 2) + 3 * T(n - 3)
```

2. Escreva uma definição recursiva para uma progressão geométrica com termo inicial a e razão r.(0,75)

```
def progressao_geometrica(a, r, n):
    if n == 1:
        return a
```

```
    else:
        return r * progressao_geometrica(a, r, n - 1)
```

3. Uma coleção T de números é definida recursivamente por:

2 ∈ T

Se X ∈ T, então X+3 ∈ T e 2*X ∈ T

Quais dos seguintes números pertencem a T? 6 , 7 , 19 , 12.

Faça um programa recursivo para demonstrar. (0,5)

```
def conjunto_T(num):
    if num == 2:
        return True
    elif num < 2:
        return False
    elif num % 2 != 0:
        return conjunto_T(num - 3) or conjunto_T(num // 2)
    else:
        return False
```

4. Uma coleção M de números é definida recursivamente por:

2 ∈ M e 3 ∈ M

Se X ∈ M e Y ∈ M, então X*Y ∈ M .

Quais dos seguintes números pertencem a M? 6 , 9 , 16 , 21 , 26 , 54 , 72 , 218.

Faça um programa recursivo para demonstrar. (0,5)

```
def conjunto_M(num):
    if num == 2 or num == 3:
        return True
    elif num < 2:
        return False
```

```
    else:
        for i in range(2, num):
            if num % i == 0 and conjunto_M(i) and conjunto_M(num // i):
                return True
        return False
```

5. Uma coleção S de cadeias de caracteres é definida recursivamente por:

a ∈ S e b ∈ S

Se X ∈ S, então Xb ∈ S.

Quais das seguintes cadeias pertencem a S? a , ab , aba , aaab , bbbbb

Faça um programa recursivo para demonstrar. (0,5)

```
def conjunto_S(string):
    if string == "a":
        return True
    elif string == "b":
        return True
    elif len(string) > 1:
        if string[0] == "a":
            return conjunto_S(string[1:])
        elif string[-1] == "b":
            return conjunto_S(string[:-1])
        else:
            return False
    else:
        return False
```

6. Uma coleção W de cadeias de símbolos é definida recursivamente por:

a ∈ W, b ∈ W e c ∈ W

Se X ∈ W, então a(X)c ∈ W.

Quais das seguintes cadeias pertencem a S? a(b)c , a(a(b)c)c , a(abc)c , a(a(a(a)c)c)c ,

a(aacc)c

Faça um programa recursivo para demonstrar. (0,5)

```
def conjunto_W(string):
    if string == "a" or string == "b" or string == "c":
        return True
    elif string[0] == "a" and string[-1] == "c":
        return conjunto_W(string[1:-1])
    else:
        return False
```

7. Forneça uma definição recursiva para todas as cadeias binárias (cadeias formadas com os caracteres 0 e 1) contendo um número ímpar de zeros. (0,5 ponto)

```
def cadeia_binaria(n):
    if n == 1:
        return ["0", "1"]
    else:
        string_pequena = cadeia_binaria(n - 1)
        novas_strings = []
        for string in string_pequena:
            novas_strings.append(string + "0")
            novas_strings.append(string + "1")
        return novas_strings
```

8. Escreva o corpo da função recursiva para computar S(n) para uma dada sequência S(1 ponto):

a) 1 , 3 , 9 , 27 , ...

```
def S1(n):
```

```
    if n == 1:
        return 1
    else:
        return 3 ** (n - 1)
```

b) 2 , 1 , 2 , 4 , ...

```
def S2(n):
    if n == 1:
        return 2
    elif n % 2 == 0:
        return S2(n - 1) + 1
    else:
        return S2(n - 1) * 2
```

c) a , b , a + b , a + 2b , 2a + 3b , ...

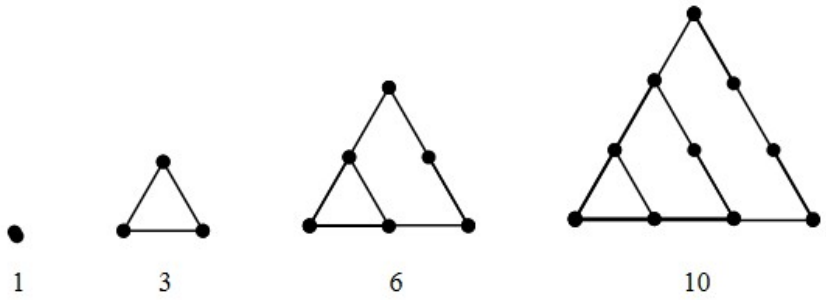
```
def S3(a, b, n):
    if n == 1:
        return a
    elif n == 2:
        return b
    else:
        return S3(a, b, n - 2) + S3(a, b, n - 1)
```

d) p , p – q , p + q , p – 2q , p + 2q , p – 3q , ...

```
def S4(p, q, n):
    if n == 1:
        return p
    elif n == 2:
        return p - q
```

```
    elif n % 2 == 0:
        return S4(p, q, n - 1) + 2 * q
    else:
        return S4(p, q, n - 1) - 2 * q
```

9. Membros antigos da Sociedade de Pitágoras definiram números figurados como sendo o número de pontos em uma certa configuração geométrica. Os primeiros números triangulares são 1, 3, 6 e 10, e são semelhantes ao diagrama da figura abaixo:



Encontre a fórmula para o n-ésimo número triangular e escreva um programa recursivo.(0,5)

```
def triangulo_num(n):
    if n == 1:
        return 1
    else:
        return n + triangulo_num(n - 1
```

10 . Em um experimento, certa colônia de bactérias tem inicialmente uma população igual a 50.000. Uma leitura é feita a cada hora e, no final deste intervalo, há três vezes mais bactérias que antes.

(a) Escreva a definição recursiva para A(n), o número de bactérias presentes no início do n-ésimo período de tempo. (0,25)

```
def A(n):
    if n == 0:
        return 50000
    else:
```

```
        return 3 * A(n - 1)
```

(b). Em quantas leituras a população excederá 153.450.026 bactérias?(0,25)

```
def pop_excesso(pop_limite, pop_atual=50000, horas=0):
    if pop_atual >= pop_limite:
        return horas
    else:
        return pop_excesso(pop_limite, pop_atual * 3, horas + 1)
```

11. (0,5) Considere o algoritmo recursivo:

Lista Rotina (Lista L, inteiro j) {

Se (j == 1)

return L;

Encontre o L[i], o maior item da lista L entre 1 e j;

Troque o L[i] pelo item L[j];

return Rotina (L, j – 1);

}

Para L = [3, 7, 4, 2, 6] faça a chamada Rotina (L, 5);:

a) Represente L e o total de chamadas realizadas à Rotina.

```
def rotina(L, j):
    if j == 1:
        return L
    else:
        max_index = L.index(max(L[:j]))
        L[j-1], L[max_index] = L[max_index], L[j-1]
        return rotina(L, j - 1)
```