MATA50 - Exercícios da Semana 02

- Grupo: UVA
- Autores: Paulo Henrique Santos Carvalho Medrado(Responsável), João Lucas Lima de Melo, Ricardo Sales Rios

Instruções

1. Altere o nome do notebook/documentação para o nome do vegetal do seu grupo.
2. Incluia o nome completo dos autores na lista de autores que auxiliaram na elaboração deste notebook. Destaque o responsável como sendo o(a) primeiro(a) autor(a).
3. Compartilhe este documento com todos os membros do grupo (que participaram da elaboração deste documento) e também com januario.ufba@gmail.com (dando permisão para edição)

Exercícios

1. Escreva um programa recursivo para obter o resultado da multiplicação de números sobre o conjunto dos naturais. Dica: $a \times b = a + a + \ldots + a (b$ vezes).

```
#dado um valor a, b, a funcao prod(a, b) retornara a + a + ... + a (b vezes)
def prod(a, b):
    #caso base, onde nao existem mais elementos a serem somados
    if b == 0:
        return 0
    #para os demais casos, eh retornado (a atual) + (instancia inferior de prod)
    return a + prod(a, b-1)

#entrada de dados
a = int(input())
b = int(input())

#chamada da funcao e impressao do resultado
print("Resultado:\t", prod(a,b))
```

```
2
3
Resultado:       6
```

2. Escreva um programa recursivo para obter o resultado da função $f : \mathbb{N} \to \mathbb{N}$ em que $f(n) = \sum_{k=1}^{n} k$

```
#dado um valor n, sum(n) retornara os valores 1 + 2 + ... + n - 1 + n
def sum(n):
    #caso base, onde o valor a ser somado eh 0
    if n == 0:
        return 0
    #para os demais casos, eh retornado (n atual) + (instancia inferior de sum)
    return n + sum(n-1)

#entrada de dados
n = int(input())

#chamada da funcao e impressao do resultado
print("Resultado:\t", sum(n))
```

```
 4
 Resultado:       10
```

Clique duas vezes (ou pressione "Enter") para editar

3. Para cada um dos problemas resolvidos URI, apresente:

- Uma brevíssima descrição do problema, em suas próprias palavras.
- A solução comentada e documentadas
- Utilize blocos de texto e código separados para cada atividade.

Combinador: Problema desenvolvido para realizar um programa que combina duas strings distintas, um caracter de cada string por vez. É possível combinar palavras de até 50 caracteres.

```
#Combinador
#entrada para os pares de string
n = int(input())

#variaveis auxiliares
output = ""
strings = []

#iteracao para a quantidade de pares de string
for j in range(n):
    #entrada das palavras que compoem o par, atribuidos a s1 e s2
    s1, s2 = input().split()

    #iteracao dos caracteres para o menor tamanho das palavras que compoem o par
    for i in range(min(len(s1), len(s2))):
        #output recebe o caracter de indice i se s1 e s2, respectivamente
        output += s1[i] + s2[i]

    #output recebe os caracteres remanescentes de s1 ou s2
    output += s1[i+1:] + s2[i+1:]
    #output é salvo em strings
    strings.append(output)
    #output é resetado, possibilitando o armazenamento da string resultante da
```

```
    #proxima iteracao
    output = ""

#iteracao sobre as strings de saida e impressao de valores
for i in range(n):
    print(strings[i])
```

UM-DOIS-TRES ou, melhor dizendo, Um passo de cada vez. Agora, estamos lidando com um problema para entender o que foi escrito. Uma criança, meu irmão, realmente ainda tem dificuldade para escrever, até porque o início da vida intelectual é realmente complicado. Sendo assim, precisamos encontrar uma maneira de desenvolver um programa que consiga lidar com todos os erros possível do garoto e, com isso, descobrir cada número que ele tentou escrever.

```
#Um-Dois-Tres
#entrada para a quantidade de palavras
n = int(input())

#estrutura auxiliar para armazenar os valores das palavras
words = []

#iterando sobre a quantidade de palavras
for i in range(n):
    #entrada da palavra a ser analisada
    aux = str(input())

    #independente da permutacao de letras, 'three' sempre tera 5 caracteres.
    # como o tamanho das palavras se mantem, eh garantido que se comprimento
    # (string) == 5. entao a palavra tem, necessariamente, valor 3
    if len(aux) == 5:
        words.append(3)
    else:
        #para a palavra ser representacao de 'one', ela precisa ter, ao menos,
        #duas letras em comum. decidimos fixar 'o' (buscando o 'n' e 'e')
        #e o 'e' (buscando o 'o' e 'n') na string. se alguma dessas duas
        #condicoes for satisfeita, a palavra tem, necessariamente, valor 1
        if aux[0] == 'o' and (aux[1] == 'n' or aux[2] == 'e') or (aux[0] == 'o' or aux[1]
            words.append(1)
        else:
            #se nao tem valor 3 nem valor 1, obrigatoriamente a palavra tem
            #valor 2
            words.append(2)

#impressao e iteracao sobre os valores armazenados
for count in words:
    print(str(count))
```

Fatorial: Quem já estudou análise combinatoria sabe, nem sempre é facil achar o fatorial de um número. Com isso, precisamos achar uma maneira pratica para achar o fatorial de qualquer número. Então, iremos desenvolver um codigo para facilitar a fatoração de qualquer número.

```
#Fatorial
#entrada do valor a ser calculado o fatorial
n = int(input())

#variavel auxiliar para o calculo do fatorial
fat = 1

#iteracao de 0 a n (inclusivo)
for i in range(n+1):
    if i >= 1:
        #para valores iguais ou maiores que um, o produto de cada elemento da
        #iteracao eh atribuido a fat
        fat *= i

#impressao do valor obtido pela iteracao
print(str(fat))
```

```
3
6
```