Computação I - Python
Laboratório 4
Atividades a serem desenvolvidas na ferramenta Machine Teaching

Seguindo com nossas boas práticas, para cada um dos exercícios a seguir:

- **NOVA NOTAÇÃO**: a partir de agora, podem aparecer na especificação do retorno desejado nos enunciados o nome de uma variável ou parâmetro entre os sinais < e >, como por exemplo <var>. Isso siginifica que espera-se o valor da variável neste lugar, e não seu identificador.
- antes de começar a escrever código, faça o estudo do problema e o planejamento de sua solução.
- lembre de botar a **documentação** direitinho, dizendo o que a função faz, quais suas entradas e qual o **tipo de dado** de cada entrada, bem como do valor de retorno da função; por exemplo, se sua função recebe dois números inteiros, nos parâmetros chamados a e b e retorna a divisão deles (possivelmente um número fracionário):

```
'''Calcula e retorna a divisão de a por b;
   int, int -> float'''
```

- escolha **nomes elucidativos** para suas funções e parâmetros;
- pense em **valores de teste** relevantes para testar sua função. Ela tem alguma resposta esperada para valores negativos? Valores fracionários? Que tal testar também com valores no extremo do conjunto de dados de interesse da função (maiores valores esperados, menores valores esperados)?
- quando estiver com dificuldade para entender algum erro de funcionamento ou resultado inadequado de sua função, não fique paralizado olhando para a tela! Pegue lápis e papel e recorra ao **teste de mesa**.
- **para fazer a entrega desta atividade prática, escreva suas funções na ferramenta Machine Teaching.**

Vamos lá!

---

1 Manipulação de strings

1. Considere que a e b são duas strings à escolha do usuário. Faça uma função chamada **concatenacao** que retorne a concatenação delas no formato *abba*.
2. Escreva uma função definida por **substitui(s, x, i)** que receba uma string s, um caractere x e um número inteiro i entre 0 e o comprimento da string, e retorne uma string igual a s, exceto que o elemento da posição i deve ser substituído pelo caractere x.

3. Escreva uma função chamada **recursiva** que receba uma string e retorne essa string no meio dela mesma. Por exemplo, ao receber a string "*abcd*", a função deve retornar "*ababcdcd*". Outro exemplo: se receber "*abcde*", a função deve retornar "*ababcdecde*"

4. Escreva uma função chamada **hashtag** que receba uma string e insira o caractere "#" no início, no meio e no final dela. Por exemplo, se a entrada for "*abcd*", a saída deve ser "#*ab*#*cd*#". Outro exemplo: se receber "*abcde*", a função deve retornar "#*ab*#*cde*#".

5. Escreva uma função chamada **diff_datas** que receba duas datas no formato "DD/MM/AAAA", sendo a segunda maior que a primeira, calcule e retorne o total de dias passados entre uma data e outra. Considere que todo mês tem 30 dias. E que o ano tem 365 dias.

   Exemplo: Se as datas são "02/03/1982" e "01/02/1983", o total de dias é 334.

2 Manipulação de tuplas

6. Faça uma função chamada **filtra_pares** que receba uma tupla com quatro elementos inteiros como parâmetro, e retorne uma nova tupla contendo apenas os elementos pares da tupla original, na mesma ordem em que se encontravam. Esse tipo de operação onde se selecionam elementos de um conjunto inicial que satisfazem uma determinada propriedade é bastante comum em computação, e se chama **filtragem**.

3 Aplicações

7. Questão OBI (Olimpíada Brasileira de Informática - 2007, Fase 1, Nível 1) - (Detectando Colisões)
   Detecção de colisão é uma das operações mais comuns (e importantes) em jogos eletrônicos. O objetivo, basicamente, é verificar se dois objetos quaisquer colidiram, ou seja, se a interseção entre eles é diferente de vazio. Isso pode ser usado para saber se duas naves colidiram, se um monstro bateu numa parede, se um personagem pegou um item, etc.

   Para facilitar as coisas, muitas vezes os objetos são aproximados por figuras geométricas simples (esferas, paralelepípedos, triângulos etc). Neste problema, os objetos são aproximados por retângulos num plano 2D.

   Escreva uma função chamada colisao que, dados dois retângulos, determine se eles se interceptam ou não. Cada retângulo é determinado pelas coordenadas x e y de dois de seus vértices diametralmente opostos, representando a diagonal que vai da esquerda para a direita e de baixo para cima. Os lados de cada retângulo são sempre paralelos aos eixos x e y.

   Entrada: Os parâmetros de entrada são duas tuplas com quatro valores inteiros cada uma, representando as coordenadas do primeiro retângulo e as coordenadas do segundo retângulo.

   Saída: A sua função deve retornar o valor booleano True caso haja interseção ou False, caso não haja.

   Exemplos
   Entrada: (0,0,1,1), (0,0,1,1) ; Saída: True

   Entrada: (0,0,2,2), (1,1,3,3) ; Saída: True

   Entrada: (0,0,1,1), (2,2,3,3) ; Saída: False