Funções

Sumário

Exercício 2 - Contido

Faça uma função que receba como entrada duas listas, **valores1** e **valores2**. A função retorna True se os valores de **valores2** ocorrerem como uma subsequência em **valores1**. A função retorna False caso contrário.

Exemplo:
Entrada:

valores1: [7, 9, 1, 4, 2, 6, 5, 7, 5, 8, 2, 1, 9, 4, 3]
valores2: [4, 2, 6, 5, 7]

Saída:

True

Etapa 1 do exercício 2

Para elaborarmos a solução do exercício, vamos primeiro fazer uma outra função:

Faça uma função chamada **maior_no_intervalo** que receba como entrada uma lista de valores **valores**, um índice **ind_ini** e um valor inteiro **n**. A função retorna o maior valor da lista **valores** entre os índices **ind_ini** e **ind_ini+n**

Exemplos:

Entrada:
valores: [7, 9, 1, 4, 2, 6, 5, 7, 5]
ind_ini: 2
n: 3

Saída:
4

Entrada:
valores: [7, 9, 1, 4, 2, 6, 5, 7, 5]
ind_ini: 2
n: 5

Saída:
6

Solução da etapa 1

```
def maior_no_intervalo(valores, ind_ini, n):

    maior = valores[ind_ini]
    for i in range(0, n):
        # Índice de um elemento a partir do índice inicial
        ind = ind_ini + i
        if valores[ind]>maior:
            maior = valores[ind]

    return maior
```

Etapa 2 do exercício 2

Vamos agora fazer uma outra função:

Faça uma função chamada **compara** que receba como entrada duas listas, **valores1** e **valores2** e também um índice **ind_ini**. A função retorna True se os valores de **valores2** ocorrerem como uma subsequência em **valores1** entre os índices **ind_ini** e **ind_ini+len(valores2)** de **valores1**. A função retorna False caso contrário.

Exemplos:

Entrada:
valores1: [7, 9, 1, 4, 2, 6, 5, 7, 5]
valores2: [4, 2, 6, 5, 7]
ind_ini: 3
Saída:
True

Entrada:
valores1: [7, 9, 1, 4, 2, 6, 5, 7, 5]
valores2: [4, 2, 6, 5, 7]
ind_ini: 2
Saída:
False

Solução da etapa 2

```
def compara(valores1, valores2, ind_ini):
    """Verifica se valores1 entre os índices ind_ini
     e ind_ini+len(valores2) é igual a valores2.
    Parâmetros:
        valores1: Primeira lista de valores
        valores2: Segunda lista de valores
        ind_ini: Primeiro índice a ser considerado em valores1
    Retorna:
        True se forem iguais e False caso contrário
    """
    n = len(valores2)
    for i in range(0, n):
        ind = ind_ini + i
        if valores1[ind]!=valores2[i]:
            return False

    return True
```

Exercício 2 - Contido

Agora podemos fazer o exercício:

Faça uma função que receba como entrada duas listas, **valores1** e **valores2**. A função retorna True se os valores de **valores2** ocorrerem como uma subsequência em **valores1**. A função retorna False caso contrário.

Exemplo:
Entrada:

valores1: [7, 9, 1, 4, 2, 6, 5, 7, 5, 8, 2, 1, 9, 4, 3]
valores2: [4, 2, 6, 5, 7]

Saída:

True

Solução

```
def contido(valores1, valores2):
    """Verifica se valores2 está contida em valores1, isto é, se
    os valores em valores2 ocorrem dentro da lista valores1, na
    mesma ordem.
    Parâmetros:
        valores1: Primeira lista de valores
        valores2: Segunda lista de valores
    Retorna:
        True se estiver contido e False caso contrário
    """
    n1 = len(valores1)
    n2 = len(valores2)

    if n2>n1:
        # valores2 é maior que valores1, não tem como estar contida
        return False

    # n1-n2 é a última posição em valores1 na qual podemos comparar
    # os valores, pois valores2 possui tamanho n2
    for ind_ini in range(0, n1-n2+1):
        # A função compara foi feita no exercício anterior
        iguais = compara(valores1, valores2, ind_ini)
        if iguais:
            # Encontrou valores2 em valores1
            return True

    return False
```

Testes da solução

```
def testes():

    # valores2 contido em valores1
    valores1 = [7, 9, 1, 4, 2, 6, 5, 7, 5, 8, 2, 1, 9, 4, 3]
    valores2 = [4, 2, 6, 5, 7]
    print(contido(valores1, valores2))

    # valores2 não contido em valores1
    valores1 = [7, 9, 1, 4, 2, 6, 5, 7, 5, 8, 2, 1, 9, 4, 3]
    valores2 = [2, 2, 6, 5, 7]
    print(contido(valores1, valores2))

    # valores1 menor que valores2, resultado deve ser False
    valores1 = [7, 9, 1]
    valores2 = [2, 2, 6, 5, 7]
    print(contido(valores1, valores2))

    # duas listas iguais, resultado deve ser True
    valores1 = [7, 9, 1]
    valores2 = valores1
    print(contido(valores1, valores2))

    # valores2 vazia. Qual resultado esperado?
    valores1 = [7, 9, 1]
    valores2 = []
    print(contido(valores1, valores2))
```

Exercício 3 - Número primo

Faça uma função que receba como entrada um número inteiro e retorne True se o número for primo e False caso contrário.

* Em Python, o operador $x\%y$ retorna o resto da divisão do número x pelo número y. Portanto, dado um número x, ele será primo se $x\%y\ != 0$ para todo $2 \leq y < x$.

Solução

```
def primo(numero):
    """Verifica se um número é primo.
    Parâmetros
        numero: Valor inteiro
    Retorna
        True se o número for primo e False caso contrário."""
    if numero<=1:
        return False

    for div in range(2, numero):
        if numero%div==0:
            return False

    return True

def testes():

    print(primo(1))
    print(primo(2))
    print(primo(13))
    print(primo(30))
```

Solução 2 mais eficiente

```
def primo(numero):
    """Verifica se um número é primo.
    Parâmetros
        numero: Valor inteiro
    Retorna
        True se o número foir primo e False caso contrário."""

    if numero<=1:
        return False

    # Um número nunca é divisível por um valor maior que a
    # metade do número
    max_div = int(numero/2)
    for div in range(2, max_div+1):
        if numero%div==0:
            return False

    return True
```

Exercício 4 - Próximo primo

Faça uma função que receba como entrada um número inteiro e retorne o menor número primo maior que o número de entrada.

Solução

```
def proximo_primo(numero):
    """Encontra o próximo valor primo maior que o numero de entrada.
    Parâmetros
        numero: Valor inteiro
    Retorna
        prox_primo: Número primo encontrado
        """
    if numero<=1:
        return 2

    prox_primo = numero + 1  # Primeiro valor a ser testado
    # A função primo foi escrita no exercício anterior
    while primo(prox_primo)==False:
        prox_primo += 1

    return prox_primo

def testes():

    print(proximo_primo(1))
    print(proximo_primo(2))
    print(proximo_primo(10))
    print(proximo_primo(30))
    print(proximo_primo(1000000))
```

Exercício 5 - Repetido

Faça uma função que receba como entrada uma lista de valores e identifique se a lista possui ao menos dois valores repetidos. A função retorna True se houverem valores repetidos e False caso contrário.

Solução

```
def repetido(valores):
    """Testa se uma lista possui valores repetidos.
    Parâmetros
        valores: Lista de valores
    Retorna
        True se houver ao menos um valor repetido. False
        caso contrário"""

    n = len(valores)
    # Temos dois laços *for* para percorrer todos os pares
    # possíveis de elementos da lista
    for indice1 in range(0, n):
        for indice2 in range(0, n):
            # Comparamos apenas elementos com índices distintos
            if indice1!=indice2:
                if valores[indice1]==valores[indice2]:
                    return True

    return False
```

Solução 2

```
def repetido(valores):
    """Testa se uma lista possui valores repetidos.
    Parâmetros
        valores: Lista de valores
    Retorna
        True se houver ao menos um valor repetido. False
        caso contrário"""

    n = len(valores)
    for indice1 in range(0, n-1):
        # Dado indice1, os valores anteriores já foram comparados.
        # Então precisamos apenas comparar com os valores começando
        # em indice1+1
        for indice2 in range(indice1+1, n):
            if valores[indice1]==valores[indice2]:
                return True

    return False
```