CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75789 ARIEL GUSTAVO RODRIGUES DA S
24PRG508 - 1 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

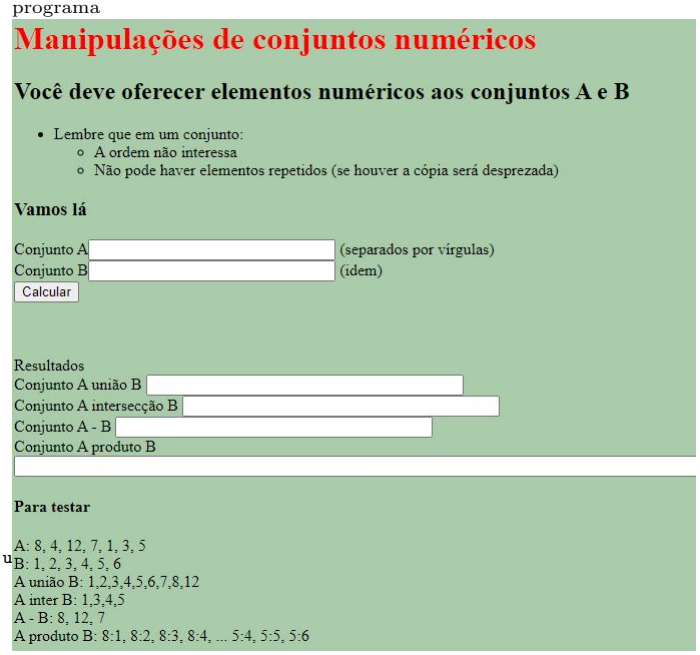
Agora execute o seu programa com os seguintes conjuntos

$A = \{8, 2, 6, 4, 1\}$
$B = \{7, 1, 6, 9, 5\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75789 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75796 CASSIANO MAGNO CHAGAS E SA
24PRG508 - 2 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

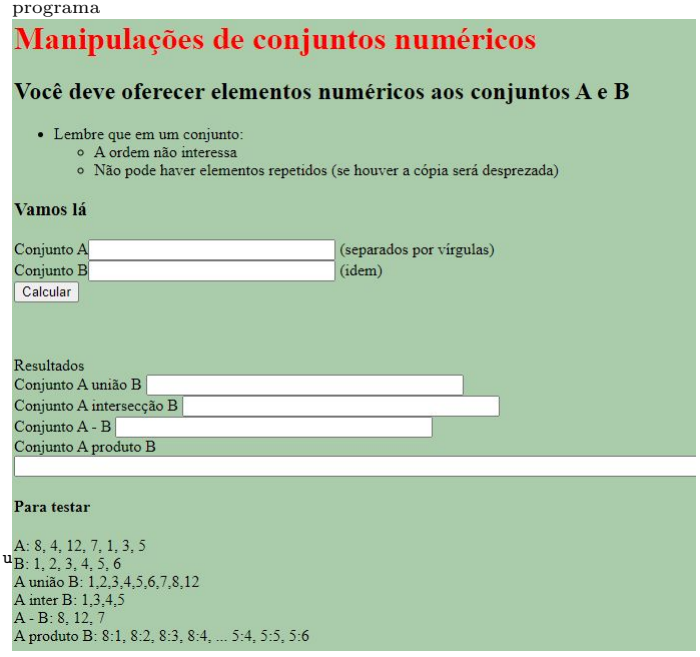
Agora execute o seu programa com os seguintes conjuntos

$A = \{6, 2, 8, 9, 7\}$
$B = \{7, 4, 9, 5, 8\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75796 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75808 CHRISTOPHER BUCH FILIPAK
24PRG508 - 3 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

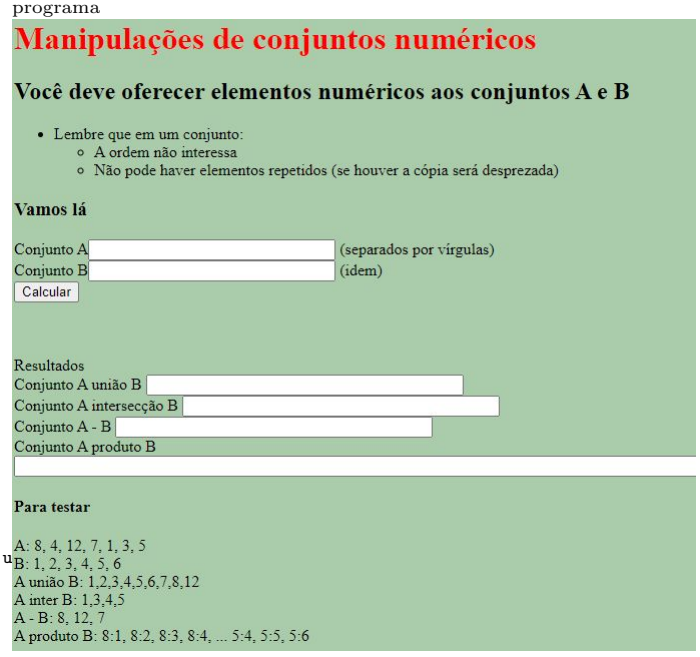
Agora execute o seu programa com os seguintes conjuntos

$A = \{5, 1, 7, 6, 4\}$
$B = \{5, 7, 1, 4, 6\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75808 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75815 ERAN MARTINEZ RAMOS
24PRG508 - 4 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

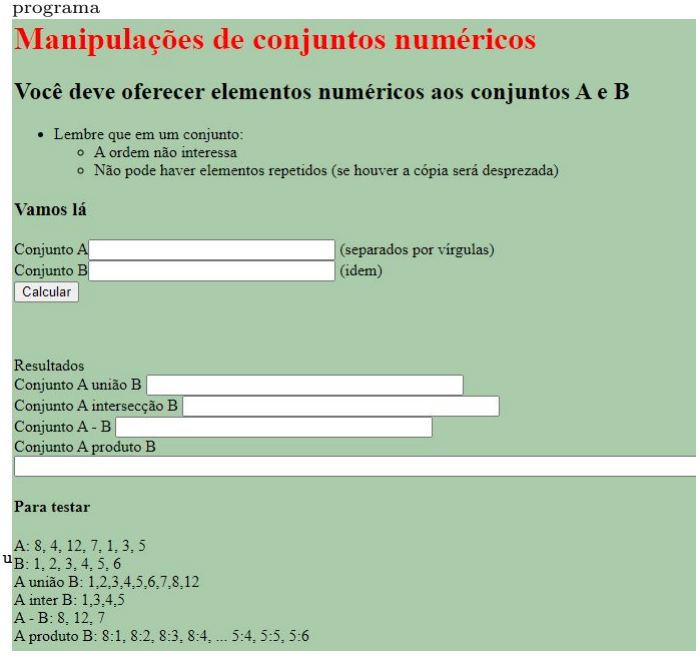
Agora execute o seu programa com os seguintes conjuntos

$A = \{1, 3, 2, 7, 5\}$
$B = \{7, 3, 8, 9, 1\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75815 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75822 HENRIQUE IVANKIO GUSSE
24PRG508 - 5 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

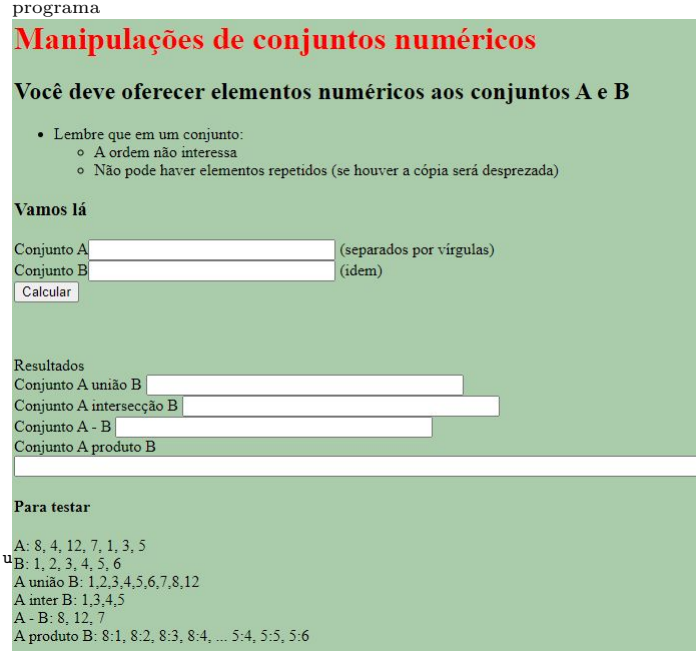
Agora execute o seu programa com os seguintes conjuntos

$A = \{7, 8, 3, 1, 6\}$
$B = \{3, 8, 2, 5, 1\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75822 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75839 IAN DOLABELLA OHATA
24PRG508 - 6 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

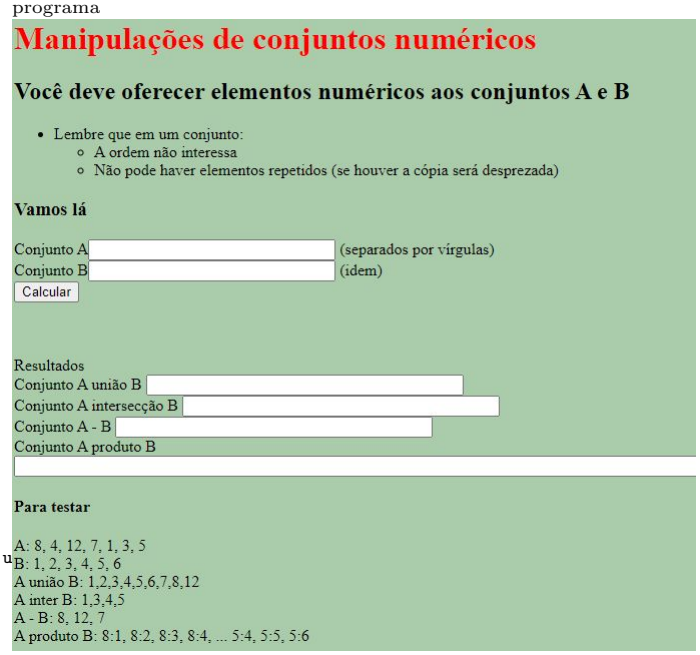
Agora execute o seu programa com os seguintes conjuntos

$A = \{6, 9, 4, 5, 1\}$
$B = \{1, 5, 9, 7, 6\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75839 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75846 JOAO BONFIN LINO
24PRG508 - 7 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

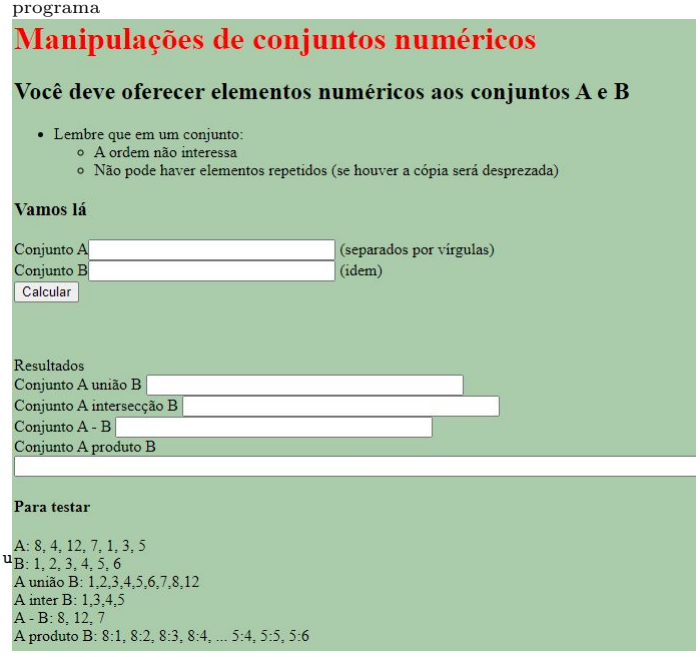
Agora execute o seu programa com os seguintes conjuntos

$A = \{2, 1, 5, 8, 3\}$
$B = \{5, 2, 8, 6, 4\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75846 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75853 JOAO LUCA FERNANDES DE OLIVE
24PRG508 - 8 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

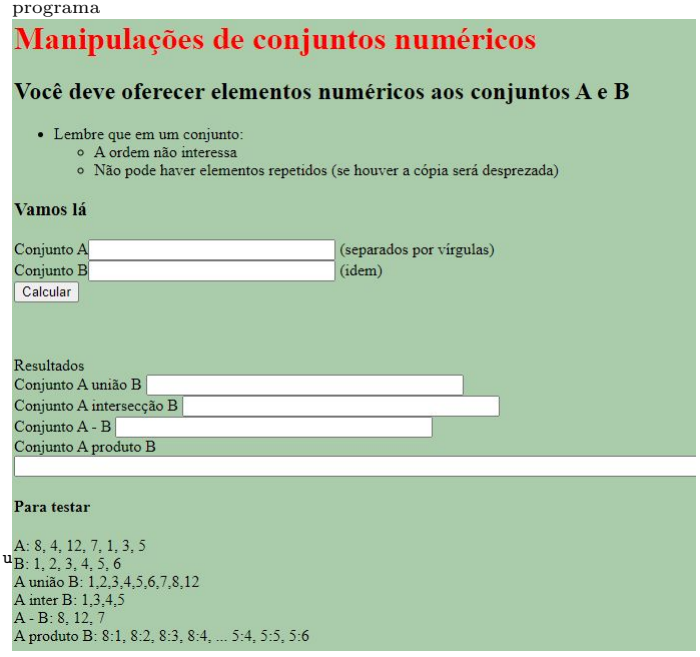
Agora execute o seu programa com os seguintes conjuntos

$A = \{3, 6, 1, 2, 7\}$
$B = \{1, 2, 9, 3, 8\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75853 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75860 JOAO PEDRO DE BRITO DUARTE
24PRG508 - 9 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

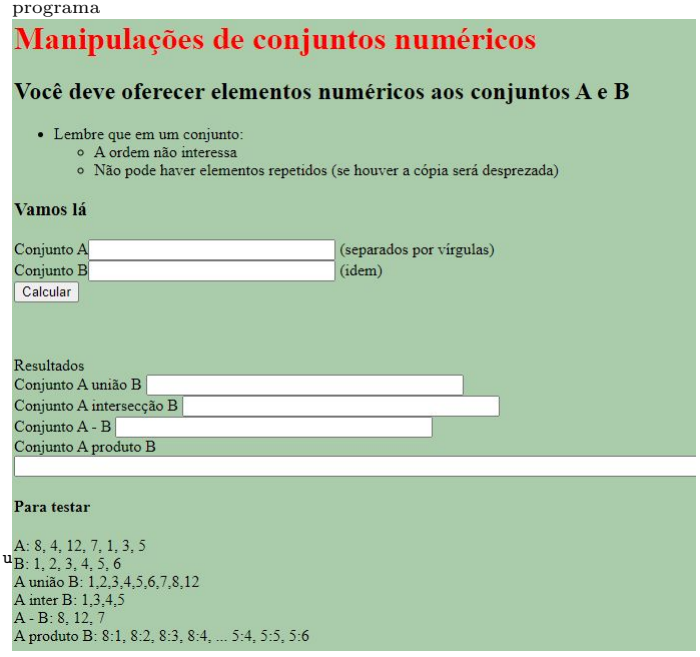
Agora execute o seu programa com os seguintes conjuntos

$A = \{1, 4, 8, 5, 2\}$
$B = \{7, 4, 2, 6, 8\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75860 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75877 JOAO PEDRO MARIANO SUEKE
24PRG508 - 10 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

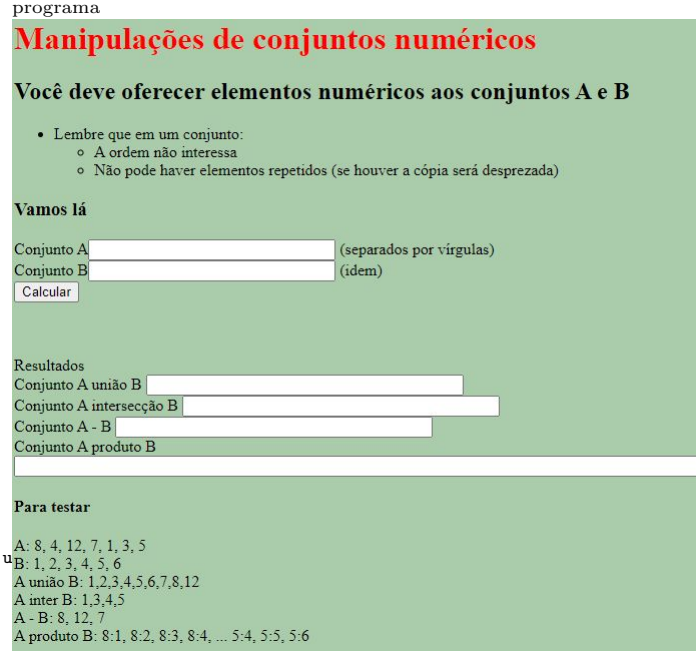
Agora execute o seu programa com os seguintes conjuntos

$A = \{7, 6, 5, 4, 1\}$
$B = \{4, 3, 5, 1, 7\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75877 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75884 JOAO VICTOR GOMES VILELA GER 24PRG508 - 11 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

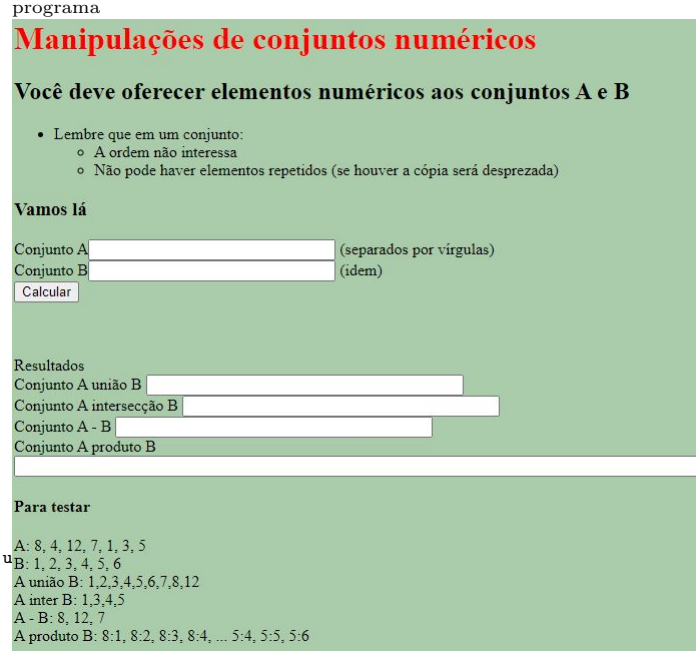
Agora execute o seu programa com os seguintes conjuntos

$A = \{1, 8, 9, 3, 5\}$
$B = \{7, 2, 9, 1, 5\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75884 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75989 LUCAS ANTOSZCZYSZEN
24PRG508 - 12 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

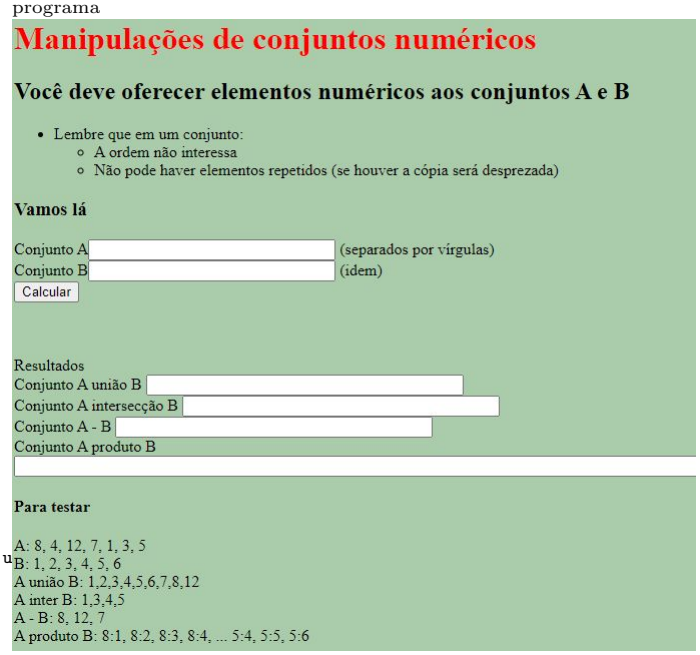
Agora execute o seu programa com os seguintes conjuntos

$A = \{7, 1, 5, 6, 4\}$
$B = \{7, 2, 9, 3, 5\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75989 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75891 LUIZ FELIPE GUEDES BUCCHERI
24PRG508 - 13 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

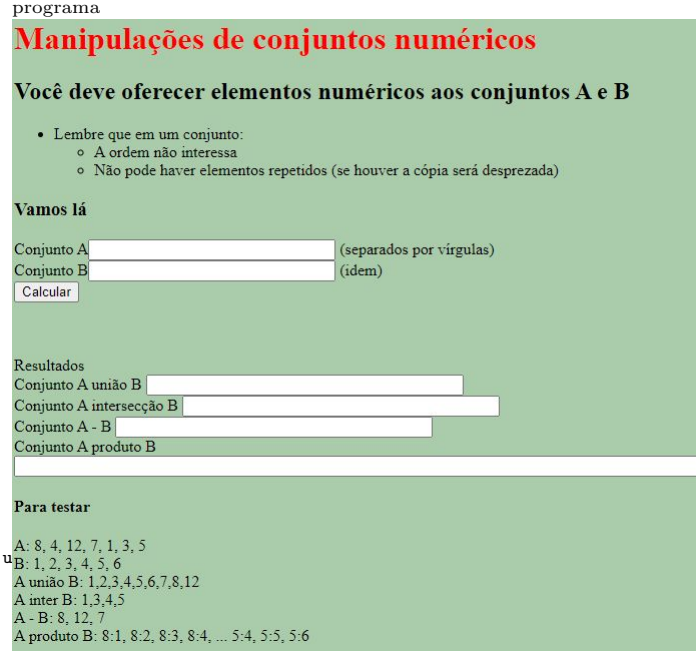
Agora execute o seu programa com os seguintes conjuntos

$A = \{9, 4, 8, 2, 6\}$
$B = \{2, 3, 9, 1, 7\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75891 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75903 MANUELLA LEAL DE MEIRELLES F 24PRG508 - 14 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

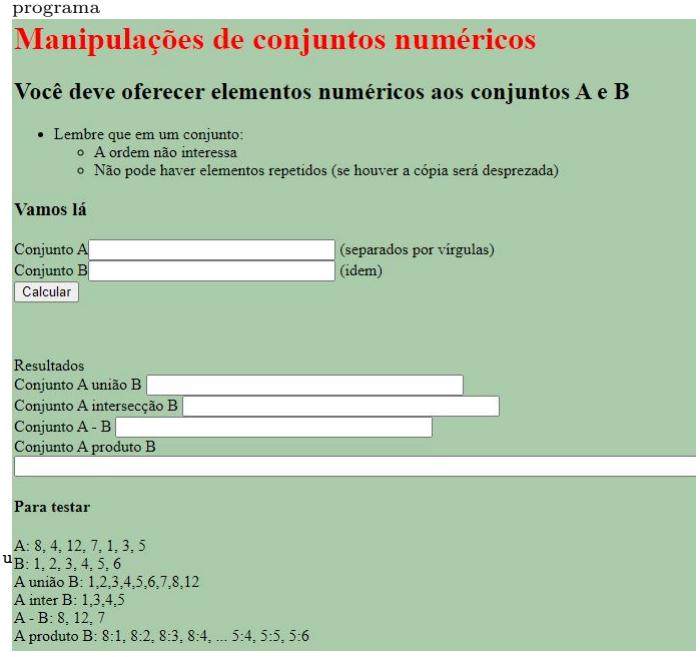
Agora execute o seu programa com os seguintes conjuntos

$A = \{1, 5, 4, 3, 7\}$
$B = \{6, 5, 4, 7, 9\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75903 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75910 MARIA EDUARDA DA SILVA
24PRG508 - 15 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

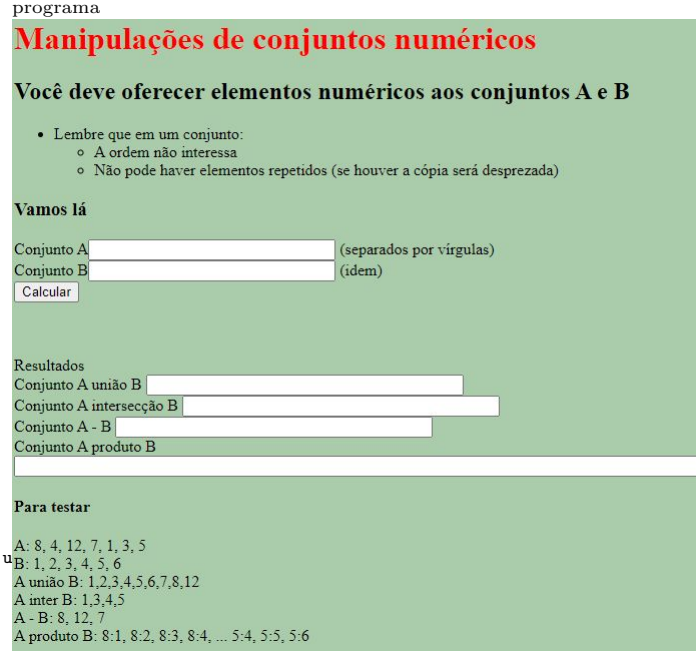
Agora execute o seu programa com os seguintes conjuntos

$A = \{3, 2, 4, 1, 5\}$
$B = \{2, 7, 3, 8, 9\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75910 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75927 MATHEUS AITA FABRICIO DE CAR
24PRG508 - 16 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

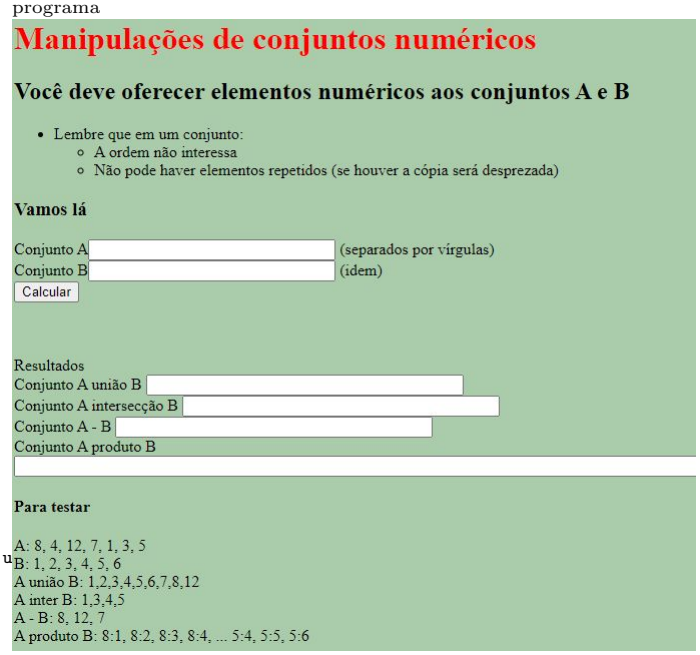
Agora execute o seu programa com os seguintes conjuntos

$A = \{2, 7, 6, 4, 3\}$
$B = \{3, 1, 4, 6, 9\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75927 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75934 MATHEUS LEANDRO DE BITENCOUR 24PRG508 - 17 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

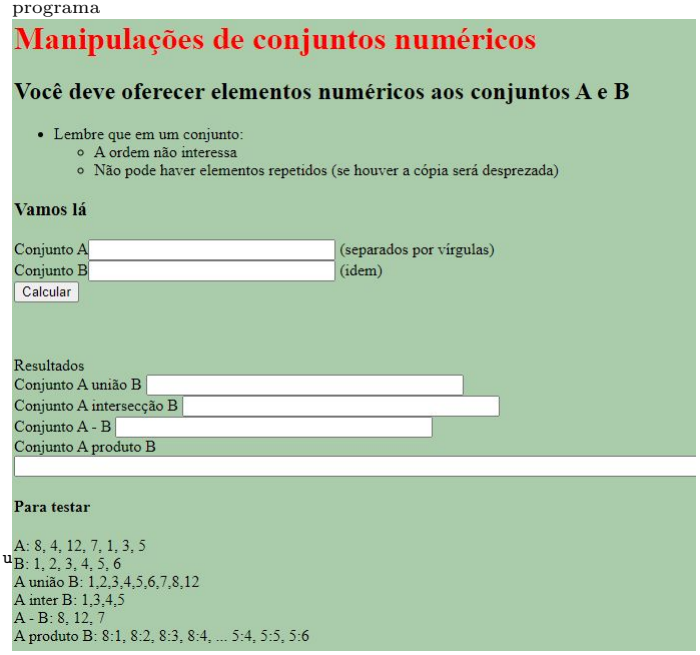
Agora execute o seu programa com os seguintes conjuntos

$A = \{7, 2, 8, 3, 4\}$
$B = \{2, 6, 5, 4, 8\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75934 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75941 MAYCON TAVARES WOLPE
24PRG508 - 18 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

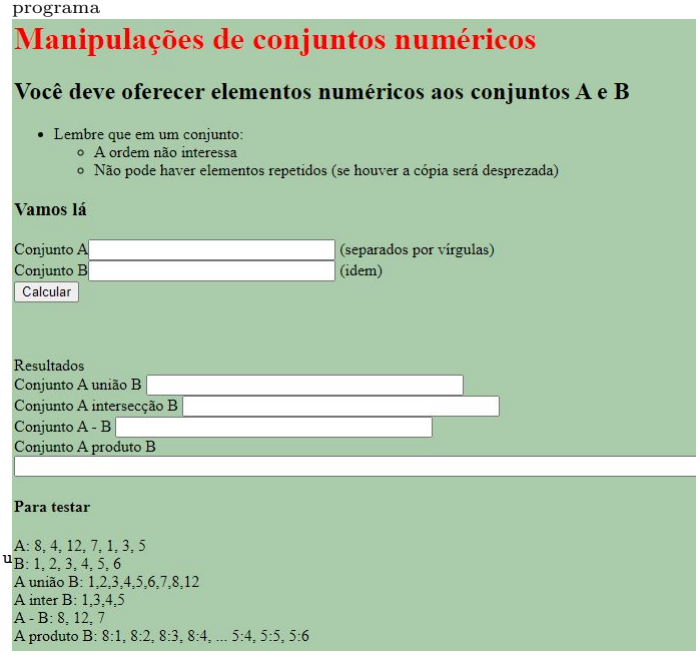
Agora execute o seu programa com os seguintes conjuntos

$A = \{3, 2, 4, 7, 9\}$
$B = \{8, 9, 2, 5, 1\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75941 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75958 PEDRO TOLEDO LEAL
24PRG508 - 19 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

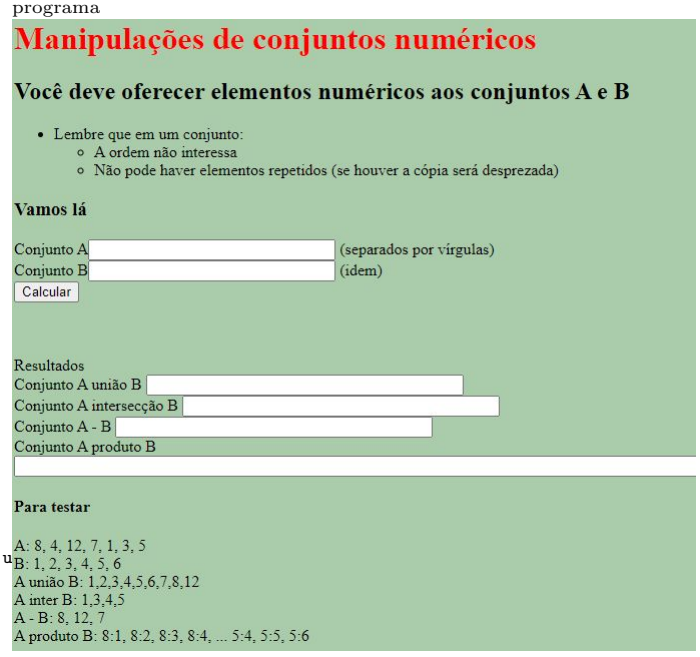
Agora execute o seu programa com os seguintes conjuntos

$A = \{2, 7, 4, 9, 5\}$
$B = \{8, 9, 5, 1, 4\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75958 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75965 PHELIPE GABRIEL LIMA DA SILV
24PRG508 - 20 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

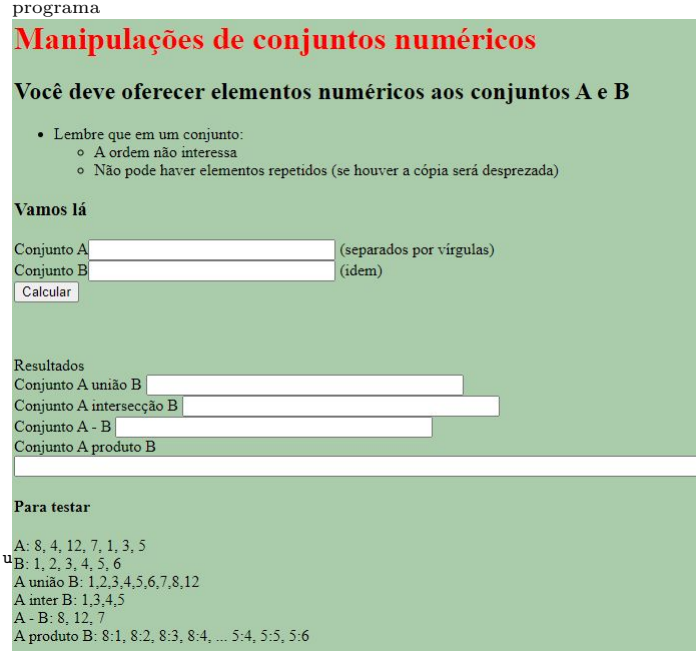
Agora execute o seu programa com os seguintes conjuntos

$A = \{3, 8, 5, 9, 2\}$
$B = \{6, 9, 1, 4, 5\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75965 - ga/ a

A tela A seguir, um possivel visual da tela do programa



CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 27/10/2024 - 09:47:22.8 Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Conjuntos em Python VIVXh97p V: 1.01
75972 REBECA CABRAL DOS SANTOS
24PRG508 - 21 entregar ate 14/nov/24 _____ / / _____

Conjuntos em Python

Ao lidar com conjuntos em Python, pode-se lançar mão do objeto `set` presente na linguagem, os literais de conjunto ({}) ou, usando listas, implementar as funcionalidades associadas a conjuntos *na mão*.

No material que segue existem as 2 coleções de funcão. As do primeiro tipo são `criarConjunto1`, `uniao1`, `intersecao1` e `diferenca1`. Já as do segundo tipo são `criarConjunto2`, `uniao2`, `intersecao2` e `diferenca2`. Ao programar esta folha, escolha qual quer implementar.

Por óbvio, ambas implementações precisam dar o mesmo resultado.

A encomenda Escreva um programa em Python que solicite ao operador dois conjuntos A e B, ambos formados por numeros inteiros, separados por vírgulas. Note que o sistema – por simplicidade – não verifica as digitações. Então, se algo errado ou esquisito, for digitado, o programa deverá ter um comportamento inesperado, errado ou errático.

Depois de ter convertido as entradas em conjuntos (eliminando eventuais elementos duplicados), o programa deverá calcular 4 novos conjuntos:

- União: $A \cup B$
- Intersecção: $A \cap B$
- Diferença: $A - B$
- Produto cartesiano: o conjunto de todos os pares (x_1, x_2) onde $x_1 \in A$ e $x_2 \in B$.

O código usando objetos set

```
def criarConjunto1(arr):
   conjunto = set(arr)
   return conjunto
ou
def criarConjunto1(arr):
   conjunto = {arr}
   return conjunto

def uniao1(conjuntoA, conjuntoB):
   return conjuntoA | conjuntoB
   # ou return conjuntoA.union(conjuntoB)

def intersecao1(conjuntoA, conjuntoB):
   return conjuntoA & conjuntoB
   # ou return conjuntoA.intersection(conjuntoB)

def diferenca1(conjuntoA, conjuntoB):
   return conjuntoA - conjuntoB
   # ou return conjuntoA.difference(conjuntoB)

def prodcartes(conjuntoA, conjuntoB):
   return [(x,y) for x in conjuntoA for y in conju
```

Código usando apenas listas

```
def criarconj2(a):
    conj=[]
    for i in range(len(a)):
        for j in range(i+1,len(a)):
            if a[j]==a[i]:
                a[j]=-9999
    for i in range(len(a)):
        if a[i]!=-9999:
            conj.append(a[i])
    return conj

def uniao2(a,b):
    conj=[]
    for i in range(len(a)):
        conj.append(a[i])
    for i in range(len(b)):
        if b[i] not in a:
            conj.append(b[i])
    return conj

def interseccao2(a,b):
    conj=[]
    for i in range(len(a)):
        for j in range(len(b)):
            if a[i]==b[j]:
                conj.append(a[i])
    return conj

def diferenca2(a,b):
    conj=[]
    for i in range(len(a)):
        if a[i] not in b:
            conj.append(a[i])
    return conj
print(diferenca2([1,3,5,8,10],[1,2,3,4]))
print(set([1,3,5,8,10])-set([1,2,3,4]))
```

☞ Para você fazer

Primeiro implemente este aplicativo em Python. Mostre para o professor.

Avaliação

Agora execute o seu programa com os seguintes conjuntos

$A = \{4, 6, 1, 7, 9\}$
$B = \{7, 5, 9, 8, 4\}$
Responda aqui:

$A \cup B$	
$A \cap B$	
$A - B$	
Produto cartesiano	

508-75972 - ga/ a

A tela A seguir, um possivel visual da tela do programa