Solution 2feb,2022

1.Write a Python program to find the elements in a given set that are not in another set.

Solution:

```
>>> sn1 = {1,2,3,4,5}
>>> sn2 = {4,5,6,7,8}
>>> print("Original sets:")
>>> print(sn1)
>>> print(sn2)
>>> print("Difference of sn1 and sn2 using difference():")
>>> print(sn1.difference(sn2))
>>> print("Difference of sn2 and sn1 using difference():")
>>> print(sn2.difference(sn1))
>>> print("Difference of sn1 and sn2 using - operator:")
>>> print(sn1-sn2)
>>> print("Difference of sn2 and sn1 using - operator:")
>>> print(sn2-sn1)
```

2 .Write a Python program to remove the intersection of a 2nd set from the 1st set.

Solution:

```
>>> sn1 = {1,2,3,4,5}
>>> sn2 = {4,5,6,7,8}
>>> print("Original sets:")
>>> print(sn1)
>>> print(sn2)
>>> print("\nRemove the intersection of a 2nd set from the 1st set using
difference_update():")
>>> sn1.difference_update(sn2)
>>> print("sn1: ",sn1)
>>> print("sn2: ",sn2)
>>> sn1 = {1,2,3,4,5}
>>> sn2 = {4,5,6,7,8}
>>> print("\nRemove the intersection of a 2nd set from the 1st set using -= operator:")
>>> sn1-=sn2
>>> print("sn1: ",sn1)
>>> print("sn2: ",sn2)
```

3 .Write a Python program to create a new deque with three items and iterate over the deque's elements.

Solution:

```
>>> from collections import deque
>>> dq = deque('aeiou')
>>> for element in dq:
>>>    print(element)
```

4Write a Python program to convert string values of a given dictionary, into integer/float datatypes.

Solution:

```
>>> def convert_to_int(lst):
>>>     result = [dict([a, int(x)] for a, x in b.items()) for b in lst]
>>>     return result

>>> def convert_to_float(lst):
>>>     result = [dict([a, float(x)] for a, x in b.items()) for b in lst]
>>>    return result
>>> nums =[{ 'x':'10' , 'y':'20' , 'z':'30' }, { 'p':'40', 'q':'50', 'r':'60'}]
>>> print("Original list:")
>>> print(nums)
>>> print("\nString values of a given dictionary, into integer types:")
>>> print(convert_to_int(nums))
>>> nums =[{ 'x':'10.12', 'y':'20.23', 'z':'30'}, { 'p':'40.00', 'q':'50.19', 'r':'60.99'}]
>>> print("\nOriginal list:")
>>> print(nums)print("\nString values of a given dictionary, into float types:")
>>> print(convert_to_float(nums))
```

5.Write a Python program to get the length of an array.

Solution:

```
>>> from array import array
>>> num_array = array('i',  [10,20,30,40,50])
>>> print("Length of the array is:")
>>> print(len(num_array))
```

6. Write a Python program to find all n-digit integers that start or end with 2?

Solution:

```
>>> def test(n):
>>>     ans = []
>>>     for i in range(10 ** (n - 1), 10 ** n):
>>>        assert len(str(i)) == n
>>>        if str(i).startswith("2") or str(i).endswith("2"):
>>>            ans.append(i)
>>>     return ans
>>> n = 1
>>> print("Number:",n)
>>> print("All",n,"- digit integers that start or end with 2:")
>>> print(test(n))
>>> n = 2
>>> print("\nNumber:",n)
>>> print("All",n,"- digit integers that start or end with 2:")
>>> print(test(n))
>>> n = 3
>>> print("\nNumber:",n)
>>> print("All",n,"- digit integers that start or end with 2:")
>>> print(test(n))
```

7.write a program For each triple of eaten, need, stock write a Python program to get a pair of total appetite and remaining.

Solution:

```
>>> def test(nums):
```

```
>>>     return [[a+min(b, c), max(0, c-b)] for a, b, c in nums]
>>>
>>> nums = [[2, 5, 6], [3, 9, 22]]
>>> print("Original list (triple) of lists:")
>>> print(nums)
>>> print("Each triple of eaten, need, stock return a pair of total appetite and remaining:")
>>> print(test(nums))
>>> nums = [[2, 3, 18], [4, 9, 2], [2, 5, 7], [3, 8, 12], [4, 9, 106]]
>>> print("\nOriginal list (triple) of lists:")
>>> print(nums)
>>> print("Each triple of eaten, need, stock return a pair of total appetite and remaining:")
>>> print(test(nums))
>>> nums = [[1, 2, 3], [4, 5, 6]]
>>> print("\nOriginal list (triple) of lists:")
>>> print(nums)
>>> print("Each triple of eaten, need, stock return a pair of total appetite and remaining:")
>>> print(test(nums))
```

8.Write a Python program to find all integers <= 1000 that are the product of exactly three primes. Each integer should represent as the list of its three prime factors.

Solution:

```
>>> def test(n):
>>>     ps = [p for p in range(2,n) if all(p % sat != 0 for sat in range(2, p))]
>>>    return [[p, q, r] for p in ps for q in ps for r in ps if p*q*r <= n]
>>> n = 10
>>> print("Number:",n)
>>> print("Find all integers <= said number that are the product of exactly three primes:")
>>> print(test(n))
>>> n = 50
>>> print("\nNumber:",n)
>>> print("Find all integers <= said number that are the product of exactly three primes:")
>>> print(test(n))
>>> n = 1000
>>> print("\nNumber:",n)
>>> print("Find all integers <= said number that are the product of exactly three primes:")
>>> print(test(n))
```

9.Write a Python program to find the sum of the magnitudes of the elements in the array with a sign that is equal to the product of the signs of the entries

Solution:

```
>>> def test(nums):
>>>     tot = sum(abs(i) for i in nums)
>>>     if all(nums):
>>>         return tot if sum(i < 0 for i in nums) % 2 == 0 else -tot
>>>     return 0

>>> nums = [1, 3, -2]
>>> print("Original list of numbers:")
>>> print(nums)
>>> print("Sum of the magnitudes of the elements in the array with a sign that is equal to the
product of the signs of the entries:")
>>> print(test(nums))
>>> nums = [1, -3, 3]
>>> print("\nOriginal list of numbers:")
>>> print(nums)
>>> print("Sum of the magnitudes of the elements in the array with a sign that is equal to the
product of the signs of the entries:")
>>> print(test(nums))
>>> nums = [10, 32, 3]
>>> print("\nOriginal list of numbers:")
>>> print(nums)
>>> print("Sum of the magnitudes of the elements in the array with a sign that is equal to the
product of the signs of the entries:")
>>> print(test(nums))
>>> nums = [-25, -12, -23]
>>> print("\nOriginal list of numbers:")
>>> print(nums)
>>> print("Sum of the magnitudes of the elements in the array with a sign that is equal to the
product of the signs of the entries:")
>>> print(test(nums))
```

10.Write a Python program to find an integer exponent x such that a^x = n.

Solution:

```
>>> a = 2
>>> n = 1024
>>> print("a = ",a,": n = ",n)
>>> print("Find an integer exponent x such that a^x = n:")
>>> print(test(n,a))
>>> a = 3
>>> n = 81
>>> print("a = ",a,": n = ",n)
>>> print("\nFind an integer exponent x such that a^x = n:")
>>> print(test(n,a))
>>> a = 3
>>> n =
12900700781701026662481960358450703949334417416449930858101164413445974926
42263849
>>> print("a = ",a,": n = ",n)
>>> print("\nFind an integer exponent x such that a^x = n:")
>>> print(test(n,a))
```