**Simple Python Tasks**

Try writing programs for each of these concepts - some test cases have been provided.

*All of these can be done in one line if you try hard enough*

1. Print the sum of two inputted numbers, or a list of numbers separated by spaces
   1. ‘15 17’ > 32
   2. ‘1 2 3 4 5’ > 15
2. Print the first *n* Fibonacci numbers
   1. ‘8’ > 1 1 2 3 5 8 13 21
3. Remove all punctuation from a string
   1. ‘ “I’m a fan, to say the least!” ‘ > Im a fan to say the least
4. Capitalize the first letter of each word in a string, and leave the rest lowercase
   1. ‘cApItAlIzE tHe FiRsT lEtTeR’ > Capitalize The First Letter
5. Count the number of vowels in a string
   1. ‘o-oooooooooo aaaae-a-a-i-a-u- jo-oooooooooooo aae-o-a-a-u-u-a- e-eee-ee-eee aaaae-a-e-i-e-a- jo-ooo-oo-oo-oo eeeeo-a-aaa-aaaa’ > 85
6. Check if a string is a palindrome (same backwards as forwards)
   1. ‘a man a plan a canal panama’ > True
   2. ‘tenant’ > False
7. Check if two strings are anagrams (same characters in both strings)
   1. ‘anagram nagaram’ > True
   2. ‘banana ban’ > False
   3. ‘bool lobb’ > False
8. Print all text from a file (if it exists), then append a user input to the end of the file
   1. Check it yourself :)
9. Find all unique permutations of a string
   1. ‘abc’ > abc acb bac bca cab cba
10. Find the longest, increasing, continuous subsequences in a string (by ascii value, not including spaces)
    1. ‘sabertooths or saberteeth’ > abert abert

**Sample Answers to Simple Python Tasks**

1. print(sum(input().split()))
2. a, b = 0, 1  
   for i in range(int(input())):  
    a, b = a + b, a  
    print(a)
3. import string  
   s = input()  
   for i in string.punctuation:  
    s = s.replace(i, '')  
   print(s)
4. s = input()  
   print([i[0].upper() + i[1:].lower() for i in s.split()])
5. s = input()  
   print(sum([s.count(i) for i in 'aeiou']))
6. s = input().replace(' ', '')  
   print(s == s[::-1])
7. s1, s2 = input().split()  
   print(sorted(s1) == sorted(s2))
8. try:  
    print(open('myfile.txt', 'r').read())  
   except:  
    print('File does not exist')  
   open('myfile.txt', 'a').write(input())
9. def perm(to\_do, permed):  
    if len(to\_do) == 1:  
    print(permed + to\_do)  
    else:  
    for i in range(len(to\_do)):  
    perm(to\_do[:i] + to\_do[i+1:], permed + to\_do[i])  
   perm(input(), '')
10. # this is some leftover code I had of it in one line   
    s = input().replace(' ', ''); print(max([[s[i:i + j] for i in range(len(s) - j + 1) if list(s[i:i + j]) == sorted(list(s[i:i + j]))] for j in range(1, len(s) + 1)], key = lambda a : len(a[0]) if len(a) > 0 else -1))

**Problems**

**Hex ascii (Easy)**

I woke up in the forest one day and found this message laying on the ground. I’m not sure what it this means but a wise man said that each 2 letters represent a character.

Message: 

**C64 (Medium)**

You may be able to decipher [26 characters](http://practicalcryptography.com/ciphers/caesar-cipher/), but what about my 64 character cipher?

Alphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789{}'

e = 'v n3h{6z 7V8 4pBC 69j5381 a9A V n37k6z 06v1TyKXzHYAMv7YDwU'

**One in Two (Medium)**

Find all the points that show up in both lists

\* hint: intersection \*

List one:

(138, 's'), (673, 'm'), (755, 's'), (471, 's'), (1516, 'f'), (392, 'm'), (65, 'b'), (266, '}'), (264, 'y'), (340, 'w'), (874, 'g'), (263, 'e'), (713, 'e'), (204, 'v'), (301, ' '), (941, ' '), (1335, 'l'), (626, ' '), (213, 'g'), (1574, 'l'), (432, 'n'), (22, ' '), (1301, 'f'), (133, 'j'), (247, 'e'), (1135, 'a'), (1727, '}'), (426, ' '), (76, 'i'), (128, 'u'), (98, 'w'), (139, 'w'), (392, 'z'), (1166, 'g'), (29, 'o'), (988, 'm'), (768, 's'), (1065, 's'), (589, 'e'), (25, 'm'), (817, 'a'), (140, 'x'), (392, '}'), (51, 'e'), (343, 'g'), (1475, '{'), (1615, 'l'), (1116, 's'), (120, 'e'), (68, 'r'), (387, 'p'), (399, 'e'), (558, 'g'), (381, 's'), (1408, 'g'), (209, 's'), (908, 'e'), (283, ' '), (269, 'k'), (508, 'a'), (209, 'l'), (68, 'p'), (35, 'v'), (39, 'k'), (1220, 'e'), (50, 'p'), (402, 't'), (80, 's'), (122, 'u'), (133, 'z'), (1265, ' '), (431, 's'), (174, 'a'), (1547, 'i'), (387, 'e'), (1694, 'r'), (209, 'u'), (310, 'b'), (390, '{'), (351, 'm'), (1395, 'a'), (331, 'f'), (1011, 'e'), (1654, 'e')

List two:

(138, 's'), (673, 'm'), (755, 's'), (61, 'v'), (132, 'x'), (471, 's'), (1516, 'f'), (323, 'n'), (353, 't'), (10, '}'), (285, 'y'), (298, 'z'), (874, 'g'), (188, 'm'), (5, 'u'), (35, 'g'), (713, 'e'), (195, 'r'), (336, 'y'), (301, ' '), (941, ' '), (1335, 'l'), (626, ' '), (213, 'g'), (1574, 'l'), (360, '}'), (218, 'g'), (1301, 'f'), (66, 'b'), (247, 'e'), (1135, 'a'), (1727, '}'), (57, 'v'), (14, 'w'), (191, '}'), (1166, 'g'), (27, 'z'), (111, 's'), (988, 'm'), (768, 's'), (1065, 's'), (589, 'e'), (25, 'm'), (817, 'a'), (341, 'b'), (361, 'w'), (273, 'z'), (51, 'e'), (238, 'f'), (1475, '{'), (1615, 'l'), (1116, 's'), (330, 'e'), (245, 'o'), (399, 'e'), (558, 'g'), (402, 'f'), (1408, 'g'), (326, 'u'), (908, 'e'), (12, 'e'), (508, 'a'), (321, 'i'), (247, 'l'), (1220, 'e'), (80, 's'), (367, 'd'), (118, 'c'), (246, 'c'), (1265, ' '), (431, 's'), (174, 'a'), (1547, 'i'), (262, 'o'), (1694, 'r'), (156, 'm'), (282, 'd'), (418, 't'), (351, 'm'), (1395, 'a'), (310, 'i'), (322, '{'), (1011, 'e'), (1654, 'e')

**ABCs (Easy)**

I found this in my grandpa’s coding attic. What was the original message?

Original output:

0102021002050200022602260185013401610189019502100195015101500206020201670137018002100205015801430193019501650163019401510166018201830240

*Note: the x’s in flag weren’t originally there, you have to figure out what characters they were originally was when the code was run*

Program:

flag = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'

a = []

for char in flag:

a += [ord(char)]

b = [a[0]]

for i in range(1, len(a)):

b += [a[i] + a[i - 1]]

c = []

for i in b:

c += [str(i).zfill(4)]

print(''.join(c))

**One Line One Flag (Hard)**

My friend showed me his python program he wrote that encodes a flag with two keys. The problem is, I can’t understand it! I know that the encoded message starts with ‘flag’

The encoded text I received is: oypxypdecnqgdmzgalgdgvgh

*Note: you might need to add the { } to make the original input flag format*

Program:

key1, key2, flag = 'ypsxebjnuqgczvrtwloaihkdmf', 'xxxx', 'xxxxxxxxxxxxxxxxxxxxxxxx'; print(''.join([dict([(key1[i], key1[i:] + key1[:i]) for i in range(len(key1))])[(key2 \* 99)[i]][key1.index(flag[i])] for i in range(len(flag))]))

**Solutions**

**Hex ascii** - flag{t0dd\_h0wwy}

b = [s[i:i + 2] for i in range(0, len(s), 2)] # split to pairs

p = [chr(int(i, 16)) for i in b] # convert to int then char

print(''.join(p))

**C64** - flag{d1CeyDr3amDub}

a='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789{}'

e = 'v n3h{6z 7V8 4pBC 69j5381 a9A V n37k6z 06v1TyKXzHYAMv7YDwU'

for shift in range(64):

p = ''

for i in e:

if i in a:

ind = a.index(i)

p += a[(ind + shift) % 64]

else:

p += i

print(p)

**One in Two** - flag{filler}

a = set(list1).instersection(set(list2))

b = list(a)

b.sort()

for i in b:

print(i[1], end = ' ') # print second part of tuple

**ABCs** - flag{gR4mPs\_d3ck\_HAs\_n0\_baD\_c4rDs}

**One Line One Flag** - flag{notcompletelyuseless}

Key2 was 'aqua'