








Python-Intermediate

Here you can start digging into our intermediate-level programming questions that will teach you new Python concepts. This category is for intermediate Python developers who already know the basics of Python development and want to expand their knowledge.

Below are some flowcharts on various python statements which will be useful to understand their use .

1. **BREAK**  2. **CONTINUE**  3. **WHILE LOOP**  4. **FOR LOOP**  5. **IF STATEMENT**  6. **IF ELSE**  7. **IF ELSE IF ELSE** 

Becoming a Python expert takes time, but over time you'll master this beautiful programming language. It's worth it!

Reference video: <https://youtu.be/HGOBQPFzWKo>

So lets get started!

1.Lets remove duplicates from list names=['A','B','C','A','D','E','F','G','H','E','D']

```
#code here
names = ['A', 'B', 'C', 'A', 'D', 'E', 'F', 'G', 'H', 'E', 'D']
set_ = set(names)
set2_ = list(set_)
print(set2_)
```

```
['H', 'F', 'G', 'A', 'D', 'B', 'E', 'C']
```

2. Good! Shall we try solving above problem in shhorter way using list comprehension? It will be amazing. Here we go!

```
#code here
temp = []

[temp.append(x) for x in names if x not in temp]
print(temp)
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H']
```

3. define a funtion 'occurence' which takes 2 inputs, one list 'lst' and another one a number 'n' and count the number of occurrences of n in your list lst.

```
#your code
def countX(occurence, x):
    count = 0
    for ele in occurence:
        if (ele == x):
            count = count + 1
    return count

#lets use your function
occurence = ([2,3,4,8,1,3,5,2,1,3,1,2,2])
x = 2
print(countX(occurence, x))
```

4

4. Calculate factorial of a number (Here number=3) using while loop

```
number = 3
#code here
def factorial(n):
    num = 1
    while n >= 1:
        num = num * n
        n = n - 1
    return num
print(factorial(3))
```

6

5. Reverse a given integer number num = 47594038222 and display the count of total number of digits in your revered number using while loop.

```
#code here
num = 47594038222

#print(f"Total number of digits in {str(num)[::-1]} are {len(str(num))}")
```

6. Can you reverse you given num without using loop in just one line?

Hint: apply some slicing trick to do it.

```
num = 47594038222
```

```
#one line code here  
print(f"Total number of digits in {str(num)[::-1]}")
```

Total number of digits in 22283049574

7. Write a function that checks whether a given sequence of number is IP address or not and print all valid possible IP from a given digit. If not an IP then it prints empty list

Note: According to Wikipedia, IPv4 addresses are canonically represented in dot-decimal notation, which consists of four decimal numbers, each ranging from 0 to 255, separated by dots, e.g., 172.16.254.1

```
from IPython.display import YouTubeVideo  
YouTubeVideo('cdUCV4j9guo',width=700, height=400)
```

VALIDATING IP ADDRESS | Python Interview Problem | Microsoft Company



#Code here

```

def range_check(n):
    if n >= 0 & n <= 255:
        return True
    return False

def prefix_zero_check(n):
    if len(n) >= 1:
        if n[0] == '0':
            return True
        return False

def isvalid(s):
    s = s.split('.')

    if len(s) != 4:
        return 0

    for n in s:
        if prefix_zero_check(n):
            return 0
        if len(n) == 0:
            return 0

    try:
        n = int(n)
        if not range_check(n):
            return 0
    except:
        return 0
    return 1

# Check your code with below numbers
A = "255.255.111.35"
B = "255.050.115.36"

print(isvalid(A))
print(isvalid(B))

```

```

1
0

```

8. Solve the quadratic equation $ax^2 + bx + c = 0$ where $a = 4, b = 6, c = 2$

```

#code here
import cmath

```

```

a = 4
b = 6
c = 2

#calculating discriminants

d = (b**2) - (4*a*c)

#soln
sol1 = (-b-cmath.sqrt(d))/(2*a)
sol2 = (-b+cmath.sqrt(d))/(2*a)
print('The solution are {0} and {1}'.format(sol1,sol2))

```

The solution are (-1+0j) and (-0.5+0j)

9. Write a program to Remove Punctuations From a String

```

s="Wow!!!, we are going ahead with our course to learn machine learning ---are'nt you excited? :)"

# code here

punctuations = '''!()-[]{};:'"\<>./?@#$$%^&*~'''
s="Wow!!!, we are going ahead with our course to learn machine learning ---are'nt you excited

no_punct = ""
for char in s:
    if char not in punctuations:
        no_punct = no_punct + char

# display the unpunctuated string
print(no_punct)

```

Wow we are going ahead with our course to learn machine learning arent you excited

10. Write a program which takes 2 digits, X,Y as input and generates a 2-dimensional array. The element value in the i-th row and j-th column of the array should be i*j.

Note: i=0,1.., X-1; j=0,1,j^Y-1.

```

digit1=3
digit2=4

```

#code here

```
row_num = int(input("input number of row: "))
col_num = int(input("input number of column:"))

multi_list = [[0 for col in range(col_num)] for row in range(row_num)]

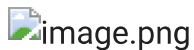
for row in range(row_num):
    for col in range(col_num):
        multi_list[row][col] = row*col

print(multi_list)
```

```
input number of row: 3
input number of column:4
[[0, 0, 0, 0], [0, 1, 2, 3], [0, 2, 4, 6]]
```

11. Write a function 'solve_coin_change' which takes list of denominations and amount and checks number of ways you can make change with coins and a total amount.

Note: You have unlimited amount of mentioned denominations



#code here

```
def solve_coin_change(S,n):
    if (n==0):    #If n =0 then there is one soln(do not include any coin)
        return 1
    if(n<0):      #If n is negative than no soln exists
        return 0
    #if(m<0):      #If there is no coin and n>0, than no soln
        return 0

    return solve_coin_change(S,n) + solve_coin_change(S,n-S)  #count is sum of solutions (i)inc

arry_1 = [1, 5, 10]
m_1 = len(arry_1)

arry_2 =[1, 2, 5]
m_2 = len(arry_2)

# Checking your code
print(solve_coin_change( [1, 5, 10],20))
print(solve_coin_change([1, 2, 5],7))
```

```

-----
RecursionError                                Traceback (most recent call last)
<ipython-input-27-1754b6b6a545> in <module>()
    20
    21 # Checking your code
--> 22 print(solve_coin_change([1, 5, 10],20))
    23 print(solve_coin_change([1, 2, 5],7))

```

----- 1 frames -----
 ... last 1 frames repeated, from the frame below ...

```

<ipython-input-27-1754b6b6a545> in solve_coin_change(S, n)
     9     return 0
    10
--> 11     return solve_coin_change(S,n) + solve_coin_change(S,n-S) #count is sum of
solutions (i)including S[m-1] (ii) excluding S[m-1]
    12
    13

```

RecursionError: maximum recursion depth exceeded in comparison

SEARCH STACK OVERFLOW

12. For given string S and integers P and Q, which denotes the cost of removal of substrings “he” and “lo” respectively from S, the task is to find the maximum cost of removing all occurrences of substrings “he” and “lo”.

example if S = “hellohellohellohello”, P = 6, Q = 4, then cost to remove 'he' and 'lo' is 50

code here

Check code with

```
S = "hellohellohellohello"
```

```
# Costs
```

```
P = 6;
```

```
Q = 4;
```

```
print(MaxCollection(S, P, Q));
```

50

13. Given a binary string S , the task is to find the smallest string possible by removing all occurrences of substrings "01" and "11". After removal of any substring, concatenate the remaining parts of the string.

Example

Input: $S = "0010110"$

Output: String = 0

Explanation: String can be transformed by the following steps:

$0010110 \rightarrow 00110 \rightarrow 010 \rightarrow 0$.

Since no occurrence of substrings 01 and 11 are remaining, the string "0" is of minimum possible length.



#code here

#check code here

```
s="0010110"
```

```
The final string is: 0
The final string size is: 1
```

14. Write a function such that it accepts two integers n, m as arguments Find the sum of all numbers in range from 1 to m (both inclusive) that are not divisible by n . Return difference between sum of integers not divisible by n with sum of numbers divisible by n

Example

Input

$n:4$

$m:20$

Output

90


```
#code here
n = int(input("n:"))
m = int(input("m:"))

sum1=0
sum2=0
for i in range(1,m+1):
    if i % n ==0:
        sum1+=i
    else:
        sum2+=i
print(abs(sum2-sum1))
```

```
n:4
m:20
90
```

15. Write a function `count_code` that return the number of times that the string "code" appears anywhere in the given string, except we'll accept any letter for the 'd', so "cope" and "cooe" count.

```
count_code('aaacodebbb') → 1
count_code('codexxcode') → 2
count_code('cozexxcope') → 2
```

```
#code here
```

```
import re
```

```
def count_code(str):
    exp = '^co[a-zA-Z]e$'
    count = 0
    for i in range(len(str) - 1):
        if re.match(exp, str[i:i + 4]):
            count = count + 1

    return count

print (count_code('aaacodebbb')) # prints 1
print (count_code('codexxcode')) # prints 2
```

```
print (count_code('cozexxcope')) #prints 3
```

```
1  
2  
2
```

16. Lets have one moderate level question for you.

Find the sum of the series $4 + 44 + 444 + 4444 + \dots$ n terms

here keep $n=5$

```
#code here
```

```
4 44 444 4444 44444  
Sum of above series is: 49380
```

17. Assume s is a string of lower case characters.

Write a program that counts up the number of vowels contained in the string s. Valid vowels are: 'a', 'e', 'i', 'o', and 'u'. For example, if s = 'azcbobobegghakl', your program should print:

```
```Number of vowels: 5``
```

```
#code here
```

```
Number of vowels: 5
```

18. well done. Now we want you to solve above question in just one line. have faith in your, you can do it.

```
one line code here.
```

```
Number of vowels: 5
```

19. Assume s is a string of lower case characters.

Write a program that prints the number of times the string 'bob' occurs in s. For example, if s = 'azcbobobegghakl', then your program should print

```
Number of times bob occurs is: 2
```

```
#code here
```

```
Number of times bob occurs is: 2
```

20. Assume s is a string of lower case characters.

Write a program that prints the longest substring of s in which the letters occur in alphabetical order. For example, if s = 'azcbobobegghakl', then your program should print

```
Longest substring in alphabetical order is: beggh
```

In the case of ties, print the first substring. For example, if s = 'abcbcd', then your program should print

```
Longest substring in alphabetical order is: abc
```

```
#code here
```

```
Longest substring in alphabetical order is: beggh
```

Awesome! Great job so far. Hope you fell in love with python now.

But wait we want to give a treat..:) We have a surprise for you in form of one of the task, in this task you have to create a game in python. You can take up this task as a mini project of your's in python. Hmm.. Sounds good. Isn't it?

So lets gets started.. we have given the instructions below for the task in detail

## ▼ Hangman Game

In this project we are going to create Hangman Game using python

### Q. What is the hangman game?

A:- Hangman is a guessing game. The player will tries to guess a word picked by the computer, by suggesting letters within a certain number of guesses

## Task 1

Create a function **is\_word\_guessed** that takes in two arguments :-

**secret\_word, letters\_guessed**

**returns True** if all of the letter of secret\_word are in letters\_guessed, **returns False** if not

### Assumptions

**secret\_word**: string, the word the user is guessing; assumes all letters are lowercase

**letters\_guessed**: list (of letters), which letters have been guessed so far; assumes that all letters are lowercase

### Example:-

secret\_word = 'apple'

letters\_guessed = ['e', 'a', 'k', 'p', 'l', 's']

is\_word\_guessed() will return True.

secret\_word = 'apple'

letters\_guessed = ['e', 'i', 'k', 'p', 'r', 's']

is\_word\_guessed() will return False.

```
def is_word_guessed(secret_word, letters_guessed):
```

```
 # Code here
```

## ▼ Task 2

Create a function **get\_guessed\_word** that takes in two arguments :-

**secret\_word, letters\_guessed**

**returns** string, comprised of known letters, and unknown letters represented by an underscore and a space ( \_ )

### Assumptions

**secret\_word**: string, the word the user is guessing;

**letters\_guessed**: list (of letters), which letters have been guessed so far

### Example:-

secret\_word = 'apple'

letters\_guessed = ['e', 'i', 'k', 'p', 'r', 's']

get\_guessed\_word() will return '\_ pp\_ e'

You want something like a \_ \_ \_ d \_ j and Not a \_\_d\_j,  
which might make it hard for the user to know how many characters are there to be guessed,  
that's why you want to use ' \_ ' instead of ' \_ '

# importing modules needed for the project

import random

import string

def get\_guessed\_word(secret\_word, letters\_guessed):

#       code here

## ▼ Task 3

Create a function **get\_available\_letter** that takes in one arguments

**letters\_guessed**

**returns** string (of letters), comprised of letters that represents which letters have not yet been guessed.

### Assumptions

**letters\_guessed**: list (of letters), which letters have been guessed so far;

### Example:-

letters\_guessed = ['a', 'l', 'm']

get\_available\_letter() will return 'bcdefghijklmnopqrstuvwxyz'

You might want to consider using string.ascii\_lowercase :=

import string

```
print(string.ascii_lowercase)
output:- abcdefghijklmnopqrstuvwxyz
```

```
def get_available_letters(letters_guessed):
 '''
 letters_guessed: list (of letters), which letters have been guessed so far
 returns: string (of letters), comprised of letters that represents which letters have not
 yet been guessed.
 '''
 # FILL IN YOUR CODE HERE
```

Great, We have created all of the Helper functions, now let's start our work on the actual Hangman Game

## Hangman Game

Create a function **hangman** that takes in one argument **secret\_word** :- string, the secret word to guess.

**The function should do the following:-**

Starts up an interactive game of Hangman.

- At the start of the game, let the user know how many letters the secret\_word contains and how many guesses s/he starts with.
- The user should start with 6 guesses
- Before each round, you should display to the user how many guesses s/he has left and the letters that the user has not yet guessed.
- Ask the user to supply one guess per round. Remember to make sure that the user puts in a letter!
- The user should receive feedback immediately after each guess about whether their guess appears in the computer's word.
- After each guess, you should display to the user the partially guessed word so far.
- If the user inputs a letter that hasn't been guessed before and the letter is in the secret word, the user loses no guesses.
- Consonants: If the user inputs a consonant that hasn't been guessed and the consonant is not in the secret word, the user loses one guess if it's a consonant.

- **Vowels:** If the vowel hasn't been guessed and the vowel is not in the secret word, the user loses two guesses. Vowels are a, e, i, o, and u. y does not count as a vowel.
- The game should end when the user constructs the full word or runs out of guesses.
- If the player runs out of guesses before completing the word, tell them they lost and reveal the word to the user when the game ends.
- If the user wins, print a congratulatory message and tell the user their score.
- The total score is the number of guesses\_remaining once the user has guessed the secret\_word times the number of unique letters in secret\_word.

Total score = guesses\_remaining \* number unique letters in secret\_word

## Demo Game 1

```
Welcome to the game Hangman!
I am thinking of a word that is 4 letters long.

You have 6 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Good guess: _ a _

You have 6 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: s
Oops! That letter is not in my word.
Please guess a letter: _ a _

You have 5 guesses left.
Available letters: bcdefghijklmnopqrtuvwxyz
Please guess a letter: t
Good guess: ta_ t

You have 5 guesses left.
Available letters: bcdefghijklmnopqrtuvwxyz
Please guess a letter: e
Oops! That letter is not in my word: ta_ t

You have 3 guesses left.
```

```
Available letters: bcd fghijklmnopquvwxyz Please guess a letter: c
Good guess: tact

Congratulations, you won!
Your total score for this game is: 6
```

## Demo Game 2

```
Welcome to the game Hangman!
I am thinking of a word that is 4 letters long

You have 6 guesses left
Available Letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! That letter is not in my word: _ _ _ _

You have 4 guesses left
Available Letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: b
Oops! That letter is not in my word: _ _ _ _

You have 3 guesses left
Available Letters: cdefghijklmnopqrstuvwxyz
Please guess a letter: c
Oops! That letter is not in my word: _ _ _ _

You have 2 guesses left
Available Letters: defghijklmnopqrstuvwxyz
Please guess a letter: d
Oops! That letter is not in my word: _ _ _ _

You have 1 guesses left
Available Letters: efghijklmnopqrstuvwxyz
Please guess a letter: e
Good guess: e_ _ e

You have 1 guesses left
Available Letters: fghijklmnopqrstuvwxyz
Please guess a letter: f
Oops! That letter is not in my word: e_ _ e
```



-----  
 Sorry, you ran out of guesses. The word was else.

## ▼ Hints

**Try to keep the output as similar to the one's above**

**Hints:-**

1. Consider writing additional helper functions if you need them.
2. There are four important pieces of information you may wish to store:
  - a. `secret_word`: The word to guess. This is already used as the parameter name for the hangman function.
  - b. `letters_guessed`: The letters that have been guessed so far. If they guess a letter that is already
  - c. `guesses_remaining`: The number of guesses the user has left. Note that in our example game, the pen

```
def no_of_unique_words(string):
 #code here
```

```
def hangman(secret_word):
 #code here
```

```
When you've completed your hangman function, test it out to see it is working fine
(hint: you might want to pick your own
secret_word while you're doing your own testing by changing 'apple' to any other string)
hangman('apple')
```

```
Welcome to the game Hangman!
I am thinking of a word that is 5 letters long.

You have 6 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Good guess: a_ _ _ _

You have 6 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: a
You've already guessed that letter

You have 6 guesses left.
```

```

Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: c
Oops! That letter is not in my word: a_ _ _ _

You have 5 guesses left.
Available letters: bdefghijklmnopqrstuvwxyz
Please guess a letter: d
Oops! That letter is not in my word: a_ _ _ _

You have 4 guesses left.
Available letters: befghijklmnopqrstuvwxyz
Please guess a letter: e
Good guess: a_ _ _ e


You have 4 guesses left.
Available letters: bfghijklmnopqrstuvwxyz
Please guess a letter: f
Oops! That letter is not in my word: a_ _ _ e

You have 3 guesses left.
Available letters: bghijklmnopqrstuvwxyz
Please guess a letter: g
Oops! That letter is not in my word: a_ _ _ e

You have 2 guesses left.
Available letters: bhijklmnopqrstuvwxyz
Please guess a letter: h
Oops! That letter is not in my word: a_ _ _ e

You have 1 guesses left.
Available letters: bijklmnopqrstuvwxyz
Please guess a letter: i
Oops! That letter is not in my word: a_ _ _ e

```


 Congratulations all of your hardwork paid of, you have just created the Hangman Game

**now go ahead and play the game you created, run the cell below**

```

words = ['army', 'beautiful', 'became', 'if', 'actually', 'beside', 'between', 'come', 'eye', 'f

#choose a random word from list using random.choice() function
random_word =

#check you function
hangman(random_word)

Welcome to the game Hangman!
I am thinking of a word that is 7 letters long.

```

```

You have 6 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: j
Oops! That letter is not in my word: _ _ _ _ _ _

You have 5 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: k
Oops! That letter is not in my word: _ _ _ _ _ _

You have 4 guesses left.
Available letters: abcdefghilmnopqrstuvwxyz
Please guess a letter: l
Oops! That letter is not in my word: _ _ _ _ _ _

You have 3 guesses left.
Available letters: abcdefghimnopqrstuvwxyz
Please guess a letter: m
Oops! That letter is not in my word: _ _ _ _ _ _

You have 2 guesses left.
Available letters: abcdefghinopqrstuvwxyz
Please guess a letter: o
Oops! That letter is not in my word: _ _ _ _ _ _

Sorry, you ran out of guesses. The word was between

```

---

Wohooo! You did a really good job. We hope you enjoyed your first mini python project here.

We understand it was bit challenging and time taking, but it's worth it, coz here you are to achieve your goal and this course will help you in all possible way to make your dreams come true.

---

Bam!, you completed the second milestone challenge too:))!!

## Feedback

We hope you've enjoyed this course so far. We're committed to help you use "AI for All" course to its full potential, so that you have a great learning experience. And that's why we need your help in form of a feedback here.

Please fill this feedback form

[https://docs.google.com/forms/d/e/1FAIpQLSfjBmH0yJSSA34lhSVx4h2eDMgOAeG4Dk-yHid\\_NMTk3Hq5g/viewform](https://docs.google.com/forms/d/e/1FAIpQLSfjBmH0yJSSA34lhSVx4h2eDMgOAeG4Dk-yHid_NMTk3Hq5g/viewform)

[Colab paid products](#) - [Cancel contracts here](#)

