Solutions

June 25, 2025

1 Solutions for Python Interview Questions

[1]:
```
# To check is number is + ve or - ve or nuetral
while True:
    Num_user= input("Enter a Number or done to exit: ")
    if Num_user.lower() == 'done':
        print("Exiting the loop")
        break

    try:
       Num=float(Num_user)
       if Num == 0:
            print("Nuetral")
       elif Num > 0:
            print("+ Ve Number")
       else:
            print("-ve number")
    except:
        print("Invalid Input; Please Enter a Number or done to exit")
```

```
Enter a Number or done to exit:  1

+ Ve Number

Enter a Number or done to exit:  -1

-ve number

Enter a Number or done to exit:  0

Nuetral

Enter a Number or done to exit:  done

Exiting the loop
```

[45]:
```
#check number is even or odd

while True:
    Num_user = input("Enter a number or done to exit: ")
    if Num_user == 'done':
        print(2 * '\n ..................')
        print("\n Exiting the loop")
        break

    try:
        num = float(Num_user)
        if num % 2 == 0:
            print(f"{num} is a Even Number")
        else:
            print(f"{num} is a Odd Number")
    except ValueError:
        print("Invalid Input, Try a Number or done to exit")
```

```
Enter a number or done to exit:  4

4.0 is a Even Number

Enter a number or done to exit:  6

6.0 is a Even Number

Enter a number or done to exit:  7

7.0 is a Odd Number

Enter a number or done to exit:  done


 …
 …

 Exiting the loop
```

[11]:
```
# year is leap or not
def year_check(year)done:
    while True:
          year_user =input("Enter the Year you want to check or done to exit: ")
          if year_user == 'done':
              print(2 * '\n ..................')
              print("\n Exiting the loop")
              break
          try:
            year = int(year_user)
            if ((year % 4 == 0) & (year % 100 != 0) or (year % 400 == 0)):
                    print(f"{year} is a leap-year")
            else:
               print(f"{year} is not a leap year")
          except ValueError:
                 print("Invalid")
```

```
Enter the Year you want to check or done to exit:  done
```

```
…
…

Exiting the loop
```

[2]:
```python
# Different methods of finding largest number
l= [2,45,67,98,56,4,5,36,365,23,675,34,678,978,908,4018]
print(max(l))
```

```
4018
```

[3]:
```python
# 2nd type to find out the largest number
num1 = float(input("Enter a Number: "))
num2 = float(input("Enter a Number: "))
num3 = float(input("Enter a Number: "))

if num1>num2 and num1>num3:
     largest = num1
elif num2>num1 and num2>num3:
     largest = num2
else:
    largest = num3
print(f"{largest} is the largest number in three")
```

```
Enter a Number:  345
Enter a Number:  454
Enter a Number:  3456

3456.0 is the largest number in three
```

[30]:
```python
# 3rd Method to find the largest

nums = [10, 45, 99, 23, 87]

nums_sorted = sorted(nums, reverse=True)
largest = nums_sorted[0]
second_largest = nums_sorted[1]
print("Largest:", largest)
print("Second Largest:", second_largest)
```

```
Largest: 99
Second Largest: 87
```

[31]:
```python
#4th method to find out
nums = [12,34,556,3,23,455,3432,23443,234,2,3]

largest = max(nums)
nums.remove(largest)
second_largest = max(nums)
```

```
print("Largest:", largest)
print("Second Largest:", second_largest)
```

```
Largest: 23443
Second Largest: 3432
```

[14]:
```
# to find  largest and 2nd largest

nums1 = [2,34,56,4,66,45,56,4,56,433,64,356,356,567]

r1= sorted(nums1,reverse=True)
print(r1)
print(r1[0])
print(r1[1])
```

```
[567, 433, 356, 356, 66, 64, 56, 56, 56, 45, 34, 4, 4, 2]
567
433
```

[2]:
```
# Find whether the word is palidrome or not
while True:
    user_input = input("Enter a Name: ")


    if user_input == user_input[::-1]:
        print(f"{user_input} is a Palidrome")
    elif user_input == 'done':
        print(2 * ('='))
        print("Exiting the loop")
        break
    else:
        print(f"{user_input} is not a palidrome")
```

```
Enter a Name:  racecar

racecar is a Palidrome

Enter a Name:  done

==
Exiting the loop
```

[24]:
```
# permutations of list or string

import itertools
data = ('Arj')

perms = list(itertools.permutations(data))
for p in perms:
    print(p)
```

```
('A', 'r', 'j')
('A', 'j', 'r')
('r', 'A', 'j')
('r', 'j', 'A')
('j', 'A', 'r')
('j', 'r', 'A')
```

[27]:
```
# Write a Python function to sum all the numbers in a list.
def sum(numbers):# function defination
    total=0 # local variable
    for x in numbers: #[1,2,3,4]
        total+=x # total=total+x
    return total

print(sum([1,2,3,4])) # function call

s=[1,2,3,4]
print(sum(s))
p=sum(s)
print(p)
```

```
10
10
10
```

[ ]:
```

```

[35]:
```
# Function to check whether number is "Prime or Not"

num= int(input("Enter a positive integer: "))

def is_prime(num):
    if num > 1:
        for n in range(2,num):
            if num % n == 0:
                return f"{num} is not a Prime Number"
        return f"{num} is a Prime Number"
    return f"{num} is not a prime number and positive integer"


print(is_prime(num))
```

```
Enter a positive integer:  3

3 is a Prime Number
```

[22]:
```
#find number is armstrong number or not
num = 153
digits =[int(d) for d in str(num)]
```

```
power = len(d)
total = sum(pow(d,power) for d in digits)

if total == num:
 print ("It is Arm strong Number")
else:
    print("Not an Arm strong")
```

```
It is Arm strong Number
```

2 Few Arm strong Numbers

- 153
- 407
- 9474
- 54748
- 548834
- 1741725,
- 4210818
- 115132219018763992565095597973971522401

[20]:
```
# checks a number whether it is armstrong or not



n = int(input("Enter a Number: "))

s = n
b = len(str(n))
sum1 = 0

while n != 0:
    r = n%10
    sum1 = sum1 + pow(r,b)
    n=n//10
if s == sum1:
    print(f"{s} is an armstrong Number")
else:
    print(f"{s} is not an Armstrong Number")
```

```
Enter a Number:  153
```

```
153 is an armstrong Number
```

[19]:
```
# printing  Armstrong numbers from 1 to 100000
def is_armstrong(n):
    dig = [int(d) for d in str(n)]
    s = len(dig)
    total = sum(pow(d,s) for d in dig)
    return n == total

for i in range(1,1000):
    if is_armstrong(i):
        print(i)
```

```
1
2
3
4
5
6
7
8
9
153
370
371
407
```

[23]:
```
# sum of list of items
l= [34,56,33,43,444,345,332,453,34]
r= sum(l)
print(r)
```

```
1774
```

[32]:
```
# sum of list of items
nums= [34,56,33,43,444,345,332,453,34]

def sum_lst(nums):
    total = 0
    for  x in nums:
        total += x
    return total

print(sum_lst(nums))
```

```
1774
```

[39]:
```
l1 = [2,3,4,5]
l2 = [3,5,6,7]

result = map(lambda x,y:x+y,l1,l2)
print(list(result))
```

```
[5, 8, 10, 12]
```

```
[41]: # Adding two lists
      l1 = [2,3,4,5]
      l2 = [3,5,6,7]

      sum(l1+l2)
```

```
[41]: 35
```

```
[42]: # Adding list
      l1 = [2,3,4,5]
      l2 = [3,5,6,7]

      print(l1+l2)
```

```
[2, 3, 4, 5, 3, 5, 6, 7]
```

```
[98]: # Punctuations Removing

      punctuations = (r"""'',.-\/@#$%&*!?:;~`""")
      new_str= ""
      user_input = input("Enter the string: ")

      for c in user_input:
          if c not in punctuations:
              new_str += c
      print (new_str)
```

```
Enter the string:  hello, how , are you/.

hello how  are you
```

```
[2]: # reverse the list
     l2= [2, 3, 4, 5, 3, 5, 6, 7]
     print(sorted(l2, reverse = True)) # sorted reverse

     print(l2[::-1]) # general reverse as it is
```

```
[7, 6, 5, 5, 4, 3, 3, 2]
[7, 6, 5, 3, 5, 4, 3, 2]
```

```
[27]: def string_cnt(s):
          d = {"upper_letter": 0, "lower_letter": 0}
          for i in s:
              if i.isupper():
                  d['upper_letter'] += 1
              elif i.islower():
                  d['lower_letter'] += 1
          return d
```

```
print(string_cnt("This Is Dallas, Texas In USA And I Flew "))
```

```
{'upper_letter': 11, 'lower_letter': 19}
```

[15]:
```
# divisible by 7 but not by 5

l = []

for i in range(0,200):
    if i%7 == 0 and i % 5 != 0:
        l.append(i)

print(l)
```

```
[7, 14, 21, 28, 42, 49, 56, 63, 77, 84, 91, 98, 112, 119, 126, 133, 147, 154,
161, 168, 182, 189, 196]
```

[21]:
```
# Trip all numbers in a list using map

lst = (2,3,4,5,6,67)

result = map(lambda x:x+x+x,lst)

print(list(result))
```

```
[6, 9, 12, 15, 18, 201]
```

[22]:
```
#getting last element of list using negative indexing
lst1 = [2, 3, 4, 5, 3, 5, 6, 7]
print(lst1[-1])
```

```
7
```

[25]:
```
# febonacci numbers

def feb(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        return feb(n-1)+feb(n-2)

for i in range(10):
    print (feb(i))
```

```
0
1
1
2
```

```
3
5
8
13
21
34
```

[36]:
```
# program to calculate recursive sum of Lists

def recur(lst):
    total = 0
    for e in lst:
        if type(e) == type([]):
            total = total + recur(e)
        else:
            total = total + e
    return total
print(recur([2,1,9,10,[2,6,7,9],3]))
```

```
49
```

[37]:
```
# to check whether the key is already in the dict or not

my_dict = {1:2,2:3,3:4,5:10}
if 2 in my_dict:
    print("in there")
else:
    print("Not There")
```

```
in there
```

[40]:
```
# count the each word in a given sentense

def count(str):
    counts = dict()
    str=str.split()
    for word in str:
        if word in counts:
            counts[word] += 1
        else:
            counts[word] = 1
    return counts

print(count("this is a test case test for is checking  for how for this is test␣
 ↪case test"))
```

```
{'this': 2, 'is': 3, 'a': 1, 'test': 4, 'case': 2, 'for': 3, 'checking': 1,
'how': 1}
```

[47]:
```
# calculate the length of the string
user_input= input("Enter a string: ")

try:
    float(user_input)
    print("Enter a String")
except ValueError:
    print(len(user_input
```

```
Enter a string:  hello how are you
```

```
17
```

[48]:
```
# calculate the length of the string
str= "hello how are you"

print(len(str))
```

```
17
```

[49]:
```
# calculate the length of the string
def str_length(str):
    count = 0
    for char in str:
        count += 1
    return count

print(str_length("hello how are you"))
```

```
17
```

[50]:
```
# empty dicts inside a list

s= [{} for _ in range (10)]
print(s)

print([{} for _ in range (10)])
```

```
[{}, {}, {}, {}, {}, {}, {}, {}, {}, {}]
[{}, {}, {}, {}, {}, {}, {}, {}, {}, {}]
```

[58]:
```
# list extension without append
l1=[1,2,3,4]
l2=[5,6,7,8]
l1.append(l2)
print(l1)

l1=[1,2,3,4]
l1.extend(l2)
print(l1)
```

```
l1=[1,2,3,4]
l1[:0]=l2
print(l1)
```

```
[1, 2, 3, 4, [5, 6, 7, 8]]
[1, 2, 3, 4, 5, 6, 7, 8]
[5, 6, 7, 8, 1, 2, 3, 4]
```

[63]:
```
# check if first and last element of list is same

lst = [1, 2, 3, 4, 5, 6, 7, 8,1]

if lst[0] == lst[-1]:
    print(f"'{lst[0]}',1st and '{lst[-1]}', last element is same")
else:
    print(f" '{lst[0]}' ,1st and '{lst[-1]}', last element is not same")



# same but with the function
def check_num(lst):
    first = lst[0]
    last = lst[-1]
    if first == last:
        return ("Both are same ")
    else:
        return("Not Same")

print(check_num([1, 2, 3, 4, 5, 6, 7, 8,1,]))
```

```
'1',1st and '1', last element is same
Both are same
```

[72]:
```
#splitting text data with multiple delimiters
import re
text = "hello!, how are you, are you doing?, ok:"
d= re.split(r'[.,{}/\|\n:;?!]' , text)
print(d)

# with spaces
e = "How are you and where are you from"
print(e.split())

# with single delimiter
f= "How,are,you,and,where,are,you,from"
print(f.split(","))

# a text which splits on the bases of any delimeter will be taken as list
```

```
['hello', '', ' how are you', ' are you doing', '', ' ok', '']
['How', 'are', 'you', 'and', 'where', 'are', 'you', 'from']
['How', 'are', 'you', 'and', 'where', 'are', 'you', 'from']
```

[84]:
```
# all the white spaces was removed and only single white space left

import re

text = "hello,  how,  are,    you,  are,  you,  doing, ok:"
r = (text.split(","))
r1= list(r)
print(re.sub(r'\s+',' ',text))
r = (text.split(","))
print(text)
```

```
hello, how, are, you, are, you, doing, ok:
hello,  how,  are,    you,  are,  you,  doing, ok:
```

[87]:
```
#accessing Index as we needed

e3 = [1, 2, 3, 4, 5, 6, 7, 8]

for i, vals in enumerate(e3, start =1):
    print (i,vals)
```

```
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
```

[97]:
```
#get current time and  date
import pytz
import datetime

time_TX = pytz.timezone('America/chicago')
print(datetime.datetime.now(time_TX).strftime('%I:%M:%S %p'))
```

```
04:13:11 PM
```

[4]:
```
#python command to add two metrices using nested loop
met1=[[7,6,8],
      [4,5,6],
      [7,9,3]]

met2 = [[4,3,2],
        [5,7,12],
        [6,9,10]]

x = [[0,0,0],
     [0,0,0],
     [0,0,0]]


for i in range (len(x)):
    for j in range (len(x[0])):
        x[i][j]= met1[i][j] + met2[i][j]

for r in x:
    print(r)
```

```
[11, 9, 10]
[9, 12, 18]
[13, 18, 13]
```

[6]:
```
#multiplication Table in python
N = int(input("Enter a Number: "))

for i in range (1,11):
    print(N, "x",i,"=", N*i)
```

```
Enter a Number:  9

9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
```

[10]:
```
#Access both keys and values using items() function from dictionary

details= {'Name':'Kiran', 'roll': 2, 'from':'Guntur'}

for i,j in details.items():
    print(i,j)
```

```
Name Kiran
roll 2
from Guntur
```

```
[17]: # sorting the string values seperated by "-"
      # input = "green-orange-purple-pink-oxyzen-white-black"

      items =[n for n in input("Enter a string seperated by '-': ").split("-")]
      items.sort()
      print("-".join(items))
```

```
Enter a string seperated by '-':  green-orange-purple-pink-oxyzen-white-black

black-green-orange-oxyzen-pink-purple-white
```

```
[ ]:
```

```
[18]: # program to detect no of local variables in a function
      def vinay():
          x=3
          y=6
          u=10
          f="hello"
          z=24

      print(vinay.__code__.co_nlocals)
```

```
5
```

```
[27]: #5 user inputs using list comprehension
      num = [int(input("Enter a number: ")) for i in range (0,5)]
      print("Users_List = ", num)
```

```
Enter a number:  3
Enter a number:  4
Enter a number:  5
Enter a number:  6
Enter a number:  10

Users_List =  [3, 4, 5, 6, 10]
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```