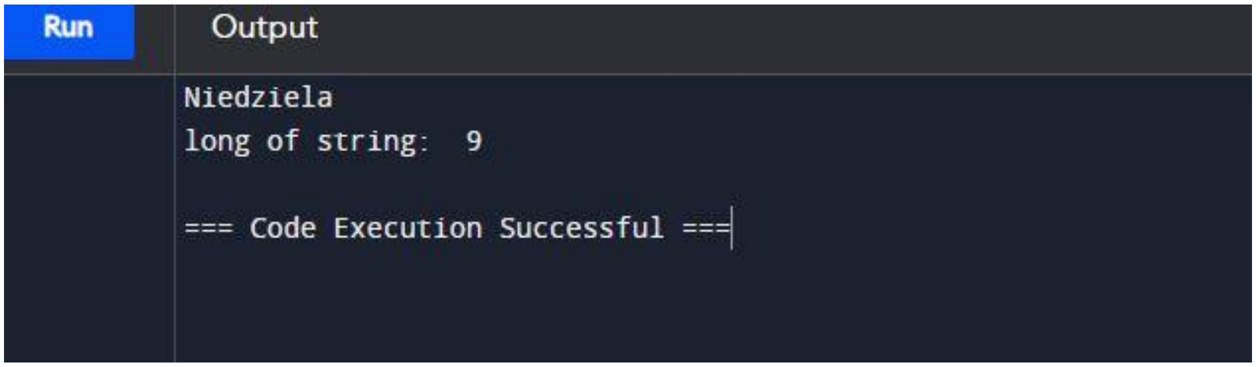
ZAD.1

1. Write a Python program to calculate the length of a string.

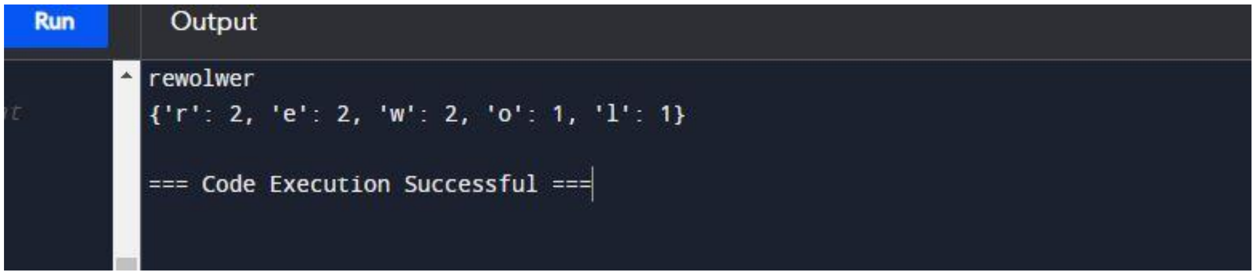
```
string ="Niedziela"
string_len=len(string)
print(string)
print("long of string: ", string_len)
```



```
Niedziela
long of string:  9

=== Code Execution Successful ===
```

2. Write a Python program to count the number of characters (character frequency) in a string.

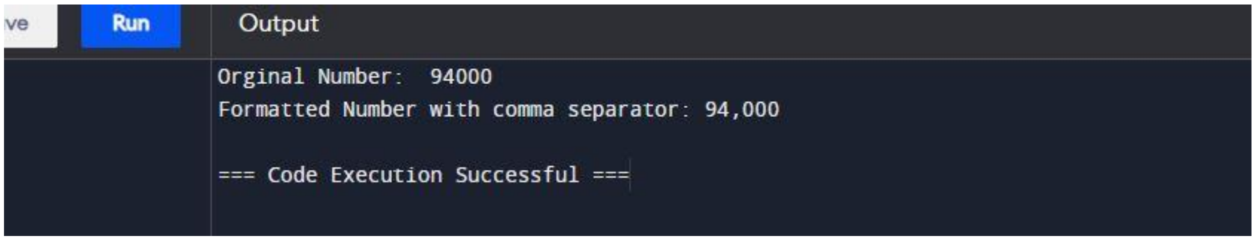
```
str1="rewolwer"
def char_frequency(str1):
   dict = {}
   for n in str1:
      keys = dict.keys()
      if n in keys:
         dict[n] += 1
      else:
         dict[n] = 1
   return dict
print(str1)
print(char_frequency(str1))
```



```
rewolwer
{'r': 2, 'e': 2, 'w': 2, 'o': 1, 'l': 1}

=== Code Execution Successful ===
```

3. Write a Python program to display a number with a comma separator.

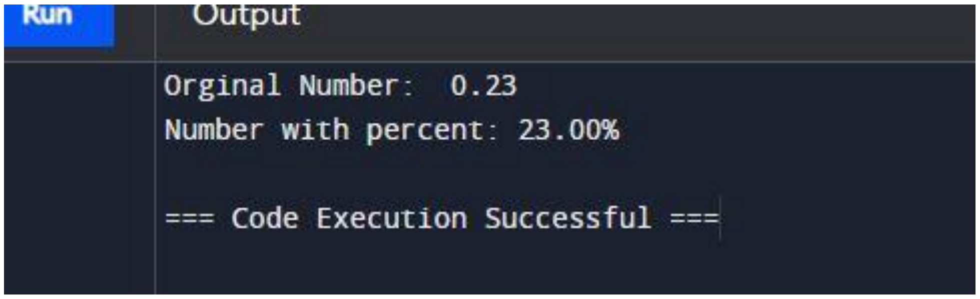
```
number= 94000
print("Orginal Number: ", number)
print("Formatted Number with comma separator: "+"{:,}".format(number))
```



```
Orginal Number:  94000
Formatted Number with comma separator: 94,000

=== Code Execution Successful ===
```

4. Write a Python program to format a number with a percentage.

```
number=0.23
print("Orginal Number: ", number)
print("Number with percent: "+"{:.2%}".format(number))
```

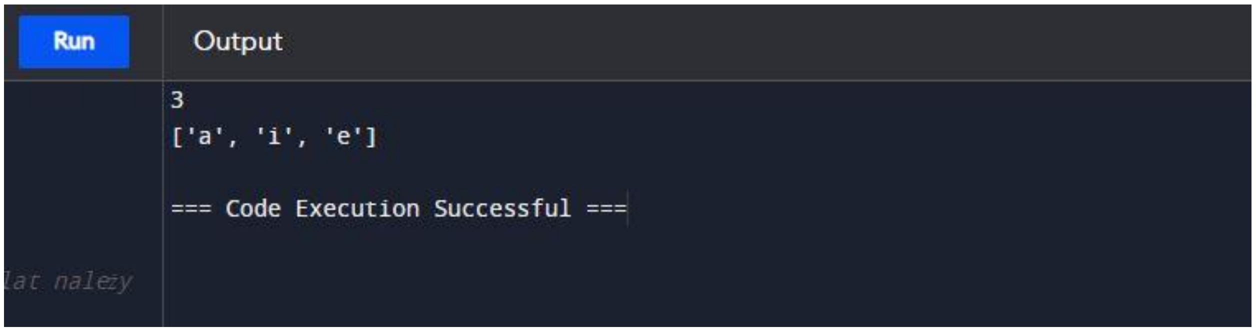


```
Orginal Number:  0.23
Number with percent: 23.00%

=== Code Execution Successful ===
```

5.Write a Python program to count and display vowels in text

```
def vowel(text):
   vowels = "aeiuoAEIOU"
   print(len([letter for letter in text if letter in vowels]))
   print([letter for letter in text if letter in vowels])
```

vowel('papier')

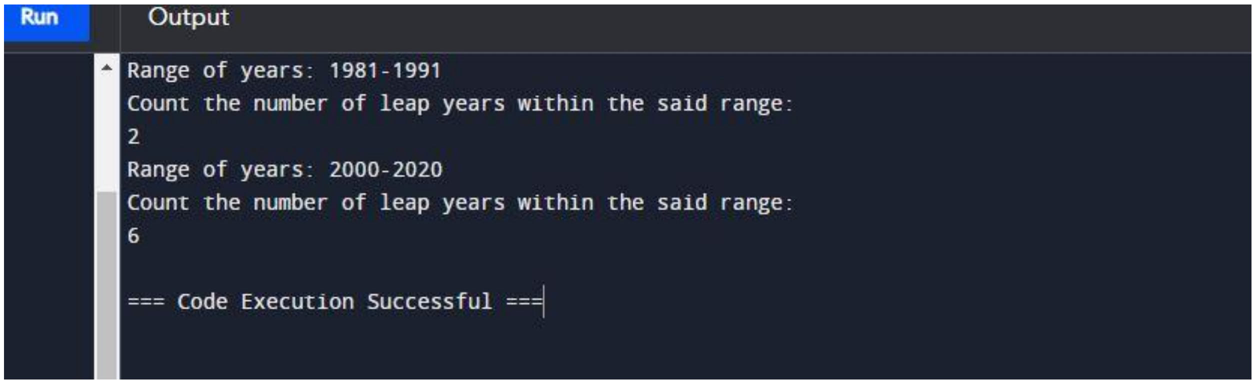


```
3
['a', 'i', 'e']

=== Code Execution Successful ===
```

6. Write a Python program that counts the number of leap years within the range of years. Ranges of years should be accepted as strings.

```
def test(r_years):
  start_year, end_year = map(int, r_years.split('-'))
  return sum(is_leap_year(year) for year in range(start_year, end_year+1))
def is_leap_year(y):
  if y % 400 == 0:
    return True
  if y % 100 == 0:
    return False
  if y % 4 == 0:
    return True
  else:
    return False
text = "1981-1991"


print("Range of years:", text)
print("Count the number of leap years within the said range:")
print(test(text))


text = "2000-2020"
```

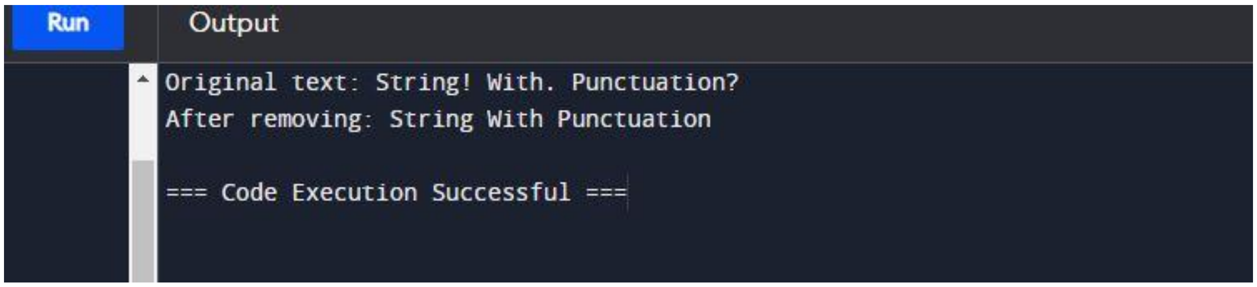
```
print("Range of years:", text)
print("Count the number of leap years within the said range:")
print(test(text))
```



```
Range of years: 1981-1991
Count the number of leap years within the said range:
2
Range of years: 2000-2020
Count the number of leap years within the said range:
6

=== Code Execution Successful ===
```

7.Write a Python program to remove punctuation from a given string.

```
def remove_punctuations(text):
   for c in string.punctuation:
      text = text.replace(c, "")
   return text
text = "String! With. Punctuation?"
print("Original text:", text)
result = remove_punctuations(text)
print("After removing:", result)
```

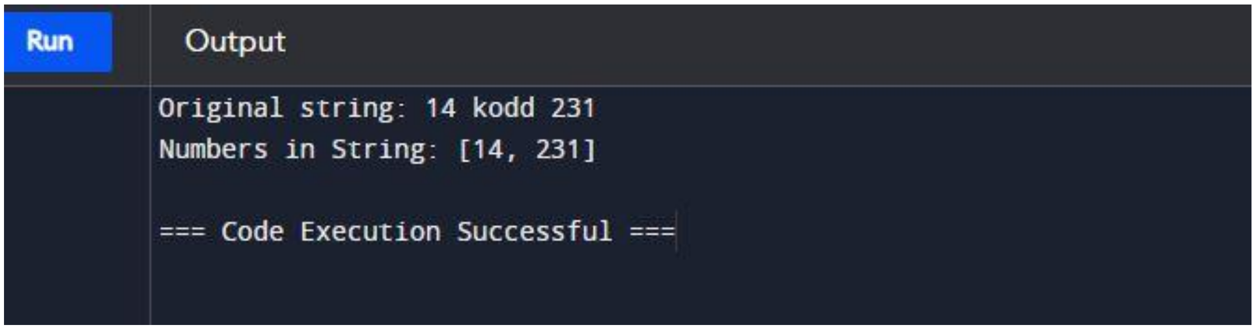


```
Original text: String! With. Punctuation?
After removing: String With Punctuation

=== Code Execution Successful ===
```

8.Write a Python program to extract numbers from a given string.

```
str1="14 kodd 231"
def only_num(str1):
   result = [int(str1) for str1 in str1.split() if str1.isdigit()]
```

```
    return result
print("Original string:", str1)
print("Numbers in String:", only_num(str1))
```

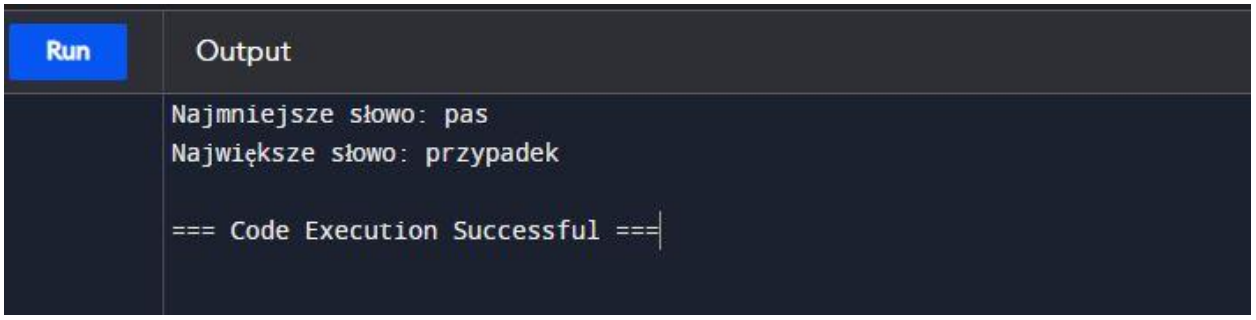

Run

Output

```
Original string: 14 kodd 231
Numbers in String: [14, 231]

=== Code Execution Successful ===
```

9.Write a Python program to find the smallest and largest words in a given string.

```
def znajdz_min_i_max_slowa(str1):
    slowa = str1.split()
    najmniejsze_slowo = None
    najwieksze_slowo = None
    for slowo in slowa:
        if najmniejsze_slowo is None or len(slowo) < len(najmniejsze_slowo):
            najmniejsze_slowo = slowo
        if najwieksze_slowo is None or len(slowo) > len(najwieksze_slowo):
            najwieksze_slowo = slowo

    return najmniejsze_slowo, najwieksze_slowo
str1 = "pas kanarek ryba przypadek"
min_slowo, max_slowo = znajdz_min_i_max_slowa(str1)
print("Najmniejsze słowo:", min_slowo)
print("Największe słowo:", max_slowo)
```

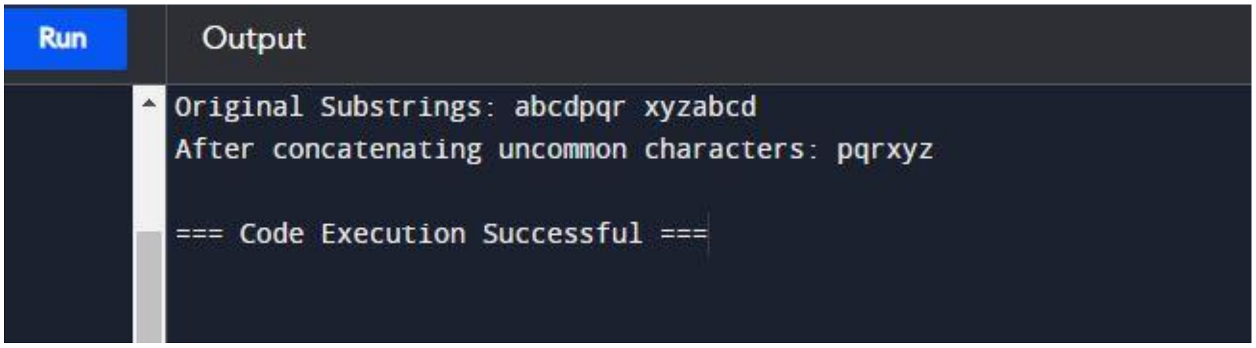

Run

Output

```
Najmniejsze słowo: pas
Największe słowo: przypadek

=== Code Execution Successful ===
```

10.Write a Python program that concatenates uncommon characters from two strings.

```
def uncommon_chars(s1, s2):


  set1 = set(s1)

  set2 = set(s2)


  common_chars = list(set1 & set2)

  result = [ch for ch in s1 if ch not in common_chars] + [ch for ch in s2 if ch not in common_chars]


  return(''.join(result))


s1 = 'abcdpqr'

s2 = 'xyzabcd'

print("Original Substrings:",s1+"",s2)

print("After concatenating uncommon characters:", uncommon_chars(s1, s2))
```

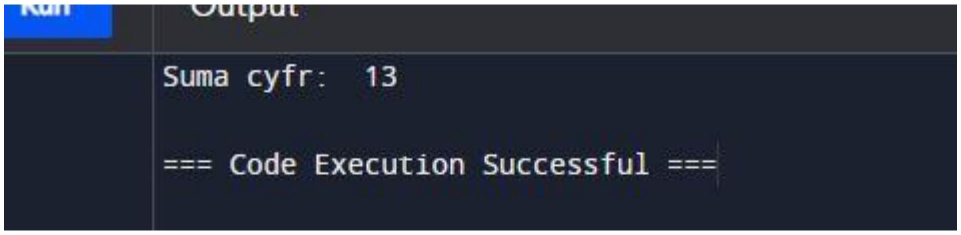

Run

Output

```
Original Substrings: abcdpqr xyzabcd
After concatenating uncommon characters: pqrxyz

=== Code Execution Successful ===
```

11.Write a Python program to compute the sum of the digits in a given string.

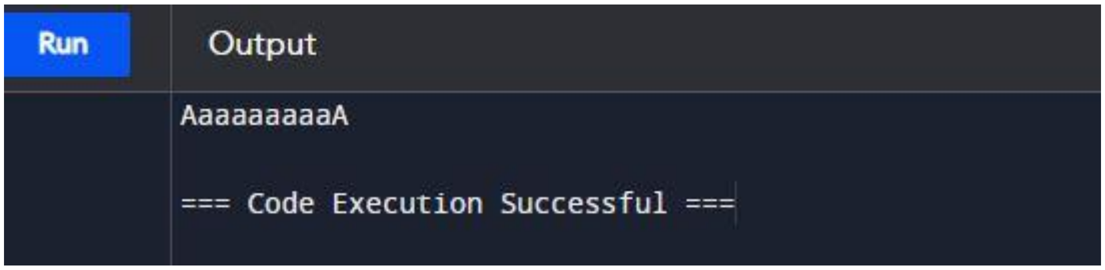
```
def sum_digits_string(str1):
    sum_digit = 0

    for char in str1:
        if char.isdigit():
            digit = int(char)
            sum_digit += digit
    return sum_digit
result1 = sum_digits_string("1kot43nic5")
print("Suma cyfr: ", result1)
```



```
Suma cyfr:  13

=== Code Execution Successful ===
```

12.Write a Python program to capitalize the first and last letters of each word in a given string.

```
def first_last_letters(str1):
    str1 = result = str1.title()
    result = ""

    for word in str1.split():
        result += word[:-1] + word[-1].upper() + " "
    return result[:-1]
print(first_last_letters("aaaaaaaaaa"))
```

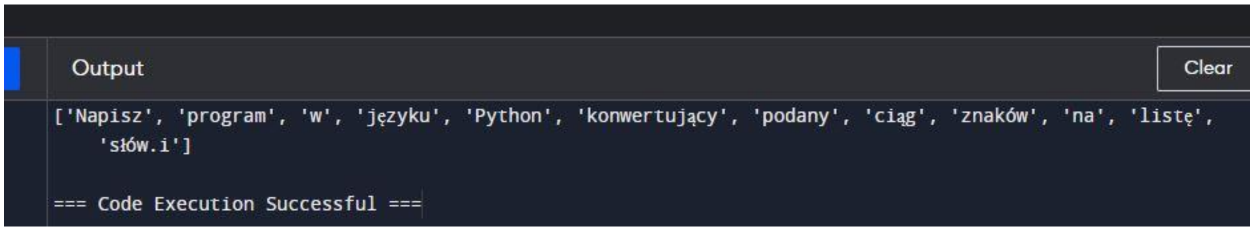


```
AaaaaaaaaA

=== Code Execution Successful ===
```

13.Write a Python program to convert a given string into a list of words.

str1 = "Napisz program w języku Python konwertujący podany ciąg znaków na listę słów.i"

print(str1.split(' '))



```
['Napisz', 'program', 'w', 'języku', 'Python', 'konwertujący', 'podany', 'ciąg', 'znaków', 'na', 'listę',
    'słów.i']

=== Code Execution Successful ===
```

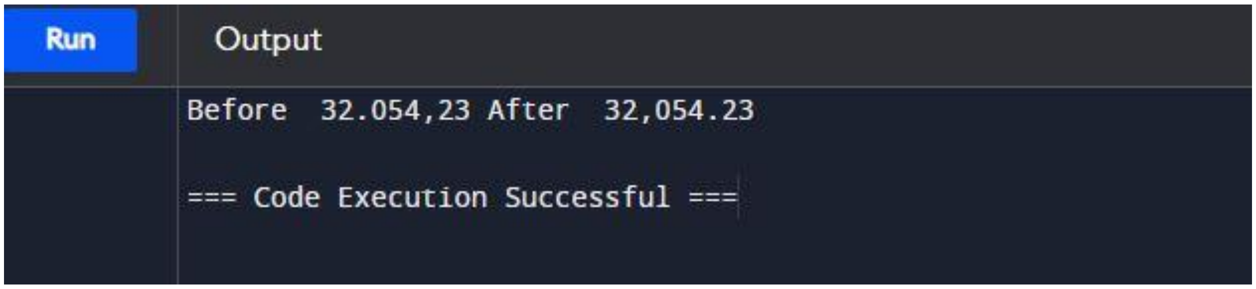
14.Write a Python program to swap commas and dots in a string.

amount = "32.054,23"

maketrans = amount.maketrans

new_amount = amount.translate(maketrans(',.', '.,'))

print("Before ",amount ,"After ", new_amount)



```
Before  32.054,23 After  32,054.23

=== Code Execution Successful ===
```

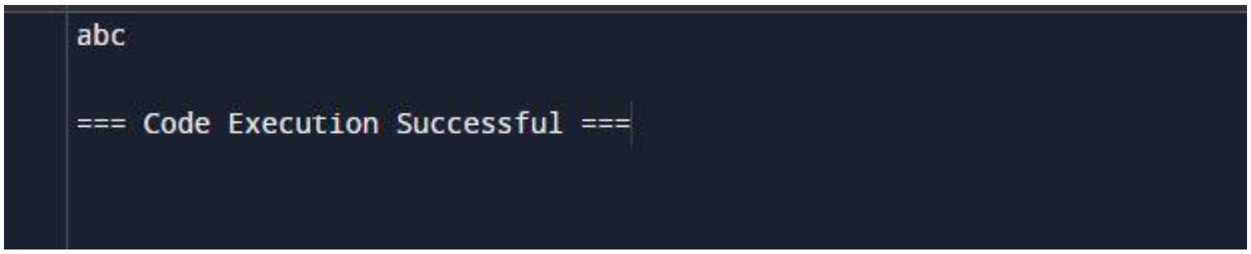
15.Write a Python program to remove spaces from a given string.

def remove_spaces(str1):

  str1 = str1.replace(' ', '')

  return str1

print(remove_spaces("a b c"))



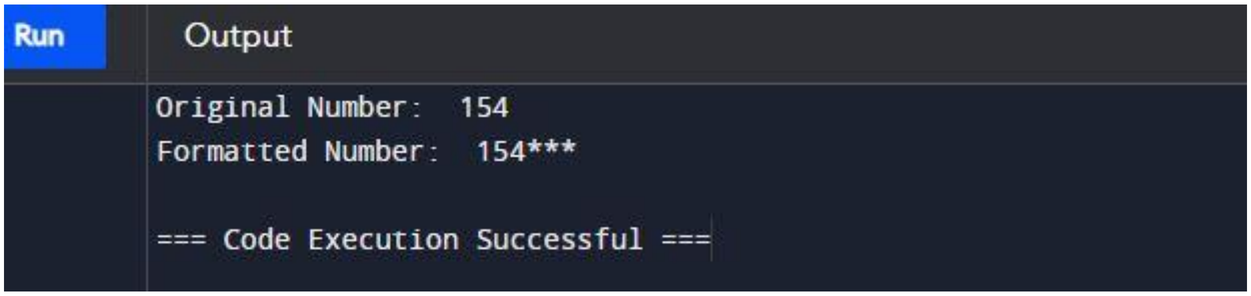
```
abc

=== Code Execution Successful ===
```

16.Write a Python program to print the following integers with '*' to the right of the specified width.

x = 154

print("Original Number: ", x)

print("Formatted Number: "+"{:*< 7d}".format(x))



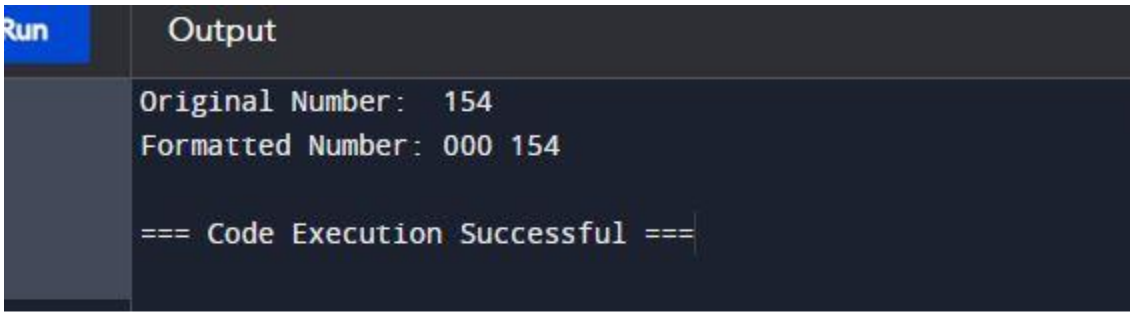
```
Original Number:  154
Formatted Number:  154***

=== Code Execution Successful ===
```

17.Write a Python program to print the following integers with zeros to the left of the specified width.

x = 154

print("Original Number: ", x)

print("Formatted Number: "+"{:0> 7d}".format(x))



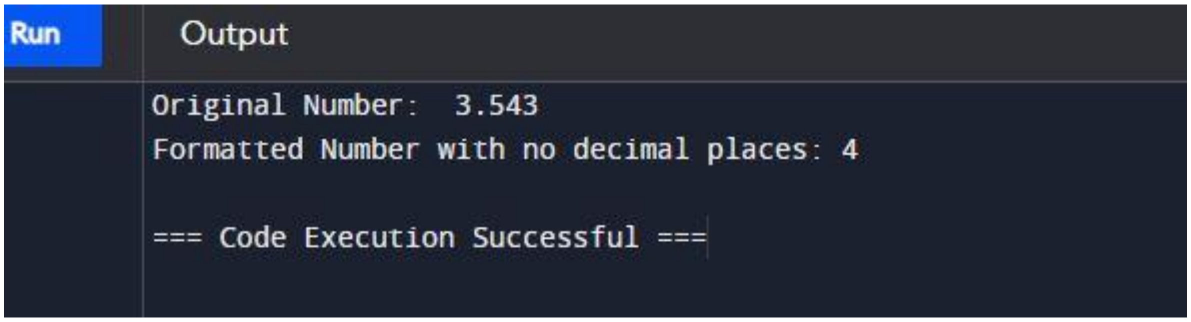
```
Original Number:  154
Formatted Number: 000 154

=== Code Execution Successful ===
```

18.Write a Python program to print the following positive and negative numbers with no decimal places.

x = 3.543

print("Original Number: ", x)

print("Formatted Number with no decimal places: "+"{:.0f}".format(x))



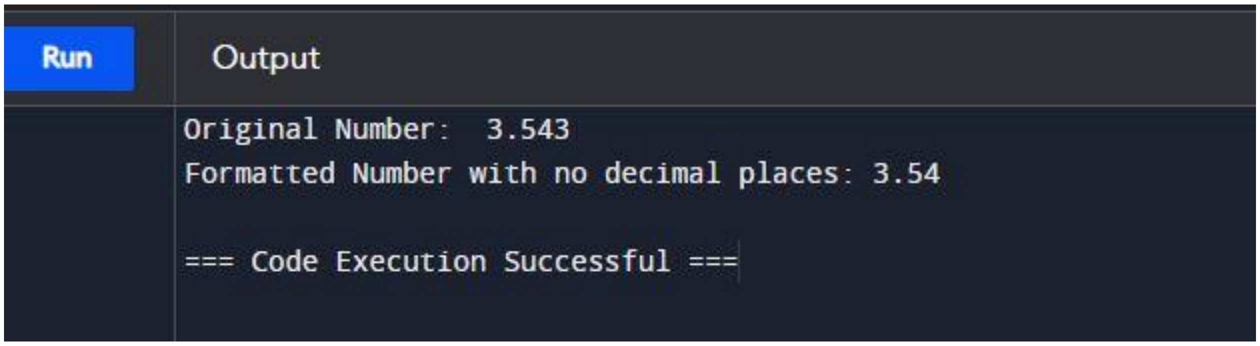
```
Original Number:  3.543
Formatted Number with no decimal places: 4

=== Code Execution Successful ===
```

19.Write a Python program to print the following numbers up to 2 decimal places.

x = 3.543

print("Original Number: ", x)

print("Formatted Number with no decimal places: "+"{:.2f}".format(x))



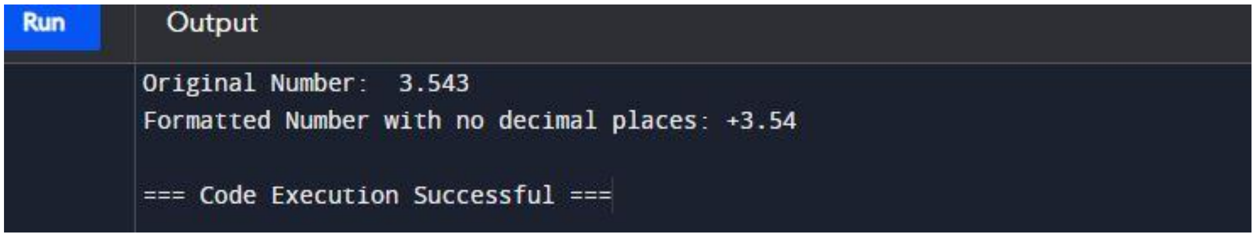
```
Original Number:  3.543
Formatted Number with no decimal places: 3.54

=== Code Execution Successful ===
```

20. Write a Python program to print the following numbers up to 2 decimal places with a sign.

x = 3.543

print("Original Number: ", x)

print("Formatted Number with no decimal places: "+"{:+.2f}".format(x))



```
Original Number:  3.543
Formatted Number with no decimal places: +3.54

=== Code Execution Successful ===
```

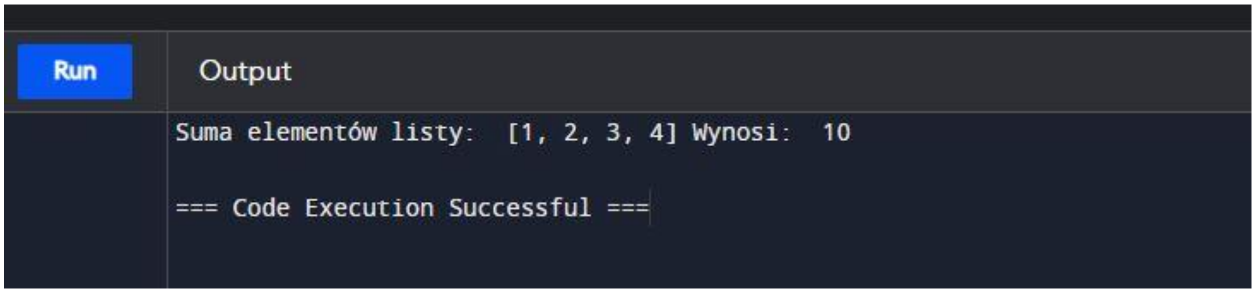
ZAD.2

1. Write a Python program to sum all the items in a list.

lis1=[1,2,3,4]

suma_lis=sum(lis1)

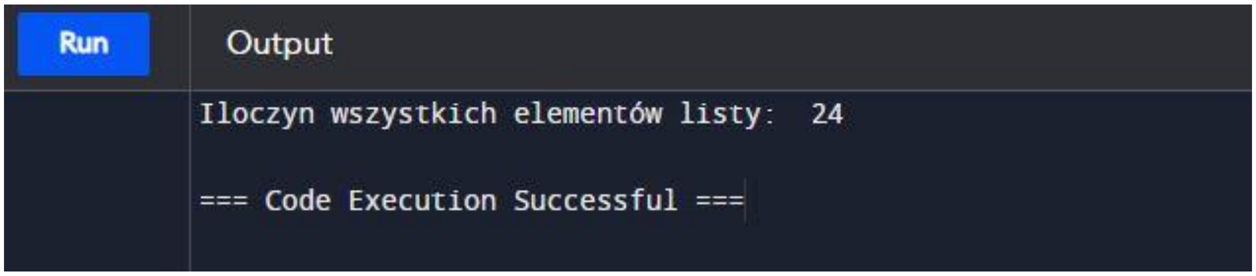
print("Suma elementów listy: ", lis1, "Wynosi: ", suma_lis)



```
Suma elementów listy:  [1, 2, 3, 4] Wynosi:  10

=== Code Execution Successful ===
```

2. Write a Python program to multiply all the items in a list.

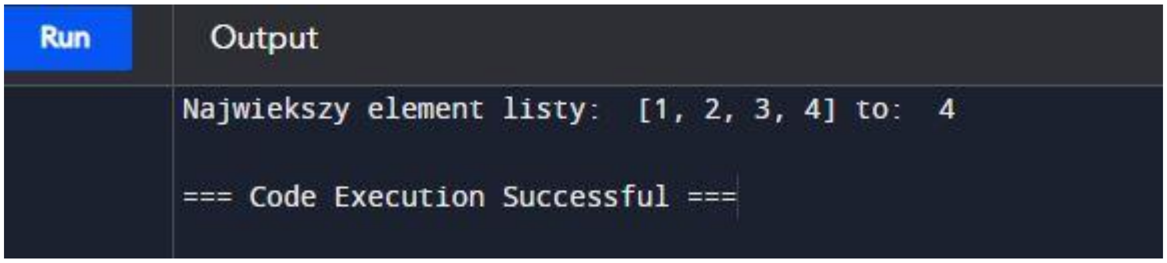
```
def multiply_list(lis1):
   tot = 1
   for x in lis1:
      tot *= x
   return tot
print("Iloczyn wszystkich elementów listy: ", multiply_list([1, 2, 3, 4]))
```



```
Iloczyn wszystkich elementów listy:  24

=== Code Execution Successful ===
```

3. Write a Python program to get the largest number from a list.

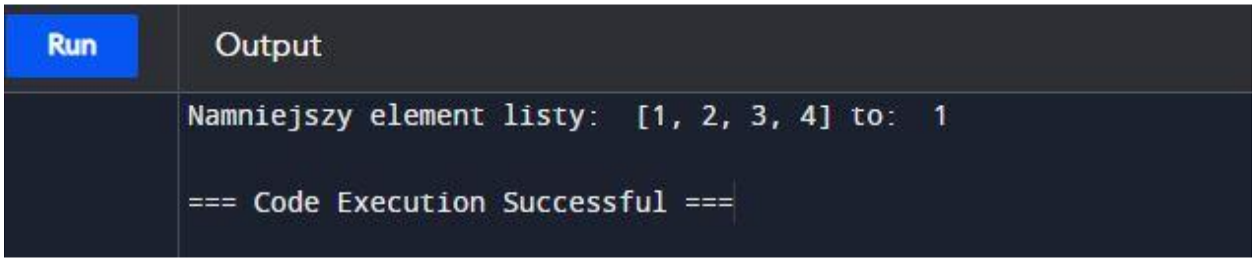
```
lis1=[1,2,3,4]
print("Najwiekszy element listy: ", lis1, "to: ",max(lis1))
```



```
Najwiekszy element listy:  [1, 2, 3, 4] to:  4

=== Code Execution Successful ===
```

4. Write a Python program to get the smallest number from a list.

```
lis1=[1,2,3,4]
print("Namniejszy element listy: ", lis1, "to: ",min(lis1))
```



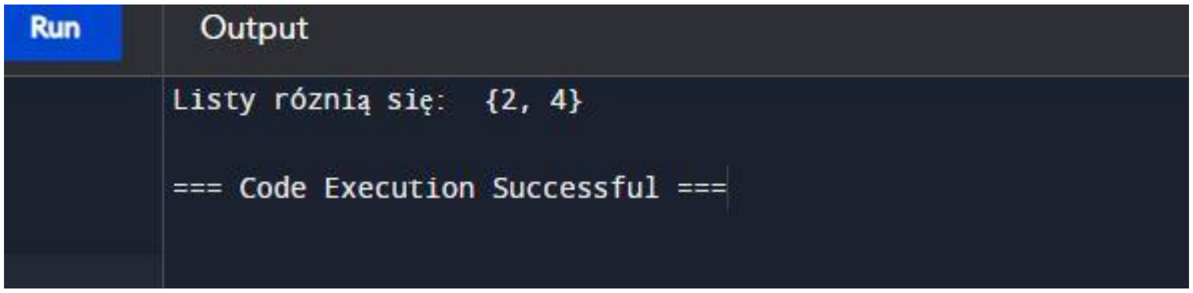
```
Namniejszy element listy:  [1, 2, 3, 4] to:  1

=== Code Execution Successful ===
```

5. Write a Python program to calculate the difference between the two lists.

lis1=[1,2,3,4]

lis2=[1,1,3,1]

print("Listy róznią się: ",set(lis1) - set(lis2))



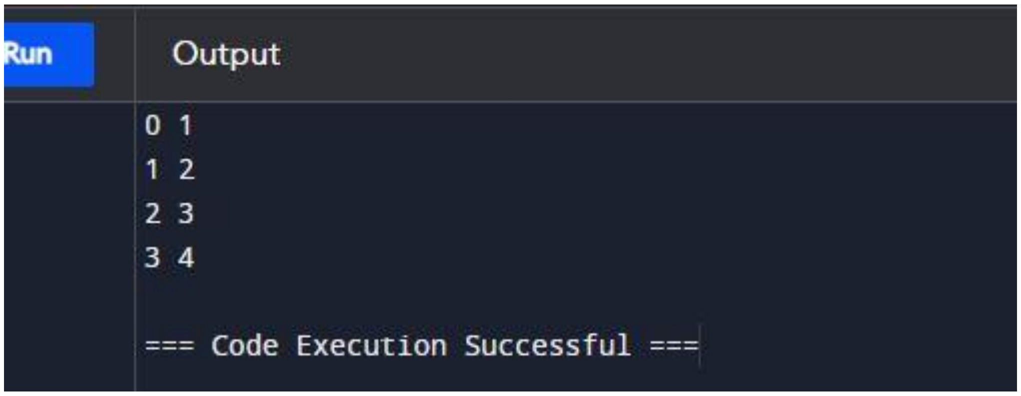
```
Listy róznią się:  {2, 4}

=== Code Execution Successful ===
```

6. Write a Python program to access the index of a list.

lis1=[1,2,3,4]

for lis1_index, lis1_val in enumerate(lis1):

    print(lis1_index, lis1_val)



```
0 1
1 2
2 3
3 4

=== Code Execution Successful ===
```

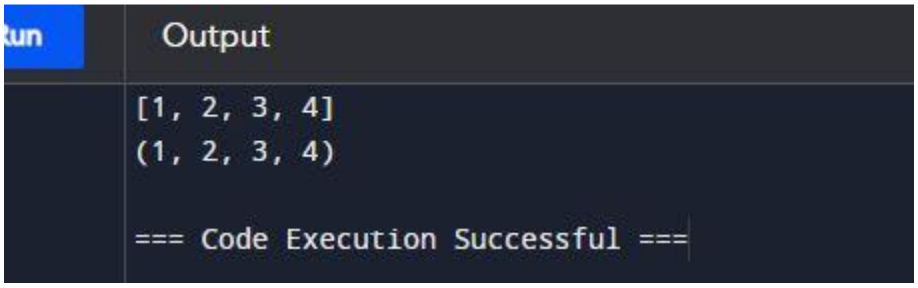
7. Write a Python program to convert a list of characters into a string.

lis1=[1,2,3,4]

print(lis1)

str1=tuple(lis1)

print(str1)



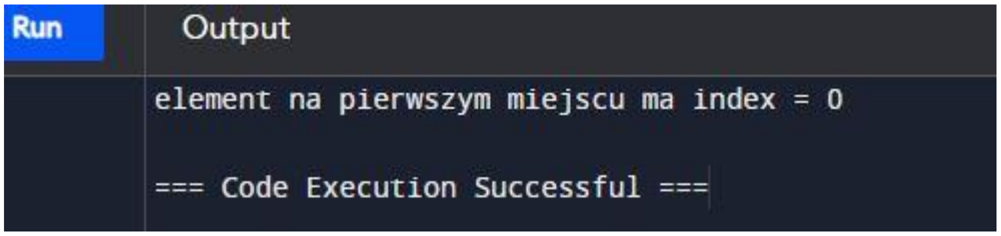
```
[1, 2, 3, 4]
(1, 2, 3, 4)

=== Code Execution Successful ===
```

8. Write a Python program to find the index of an item in a specified list.

lis1=[1,2,3,4]

index=lis1.index(1)

print("element na pierwszym miejscu ma index =", index)


Run

Output

```
element na pierwszym miejscu ma index = 0

=== Code Execution Successful ===
```

9. Write a Python program to flatten a shallow list.

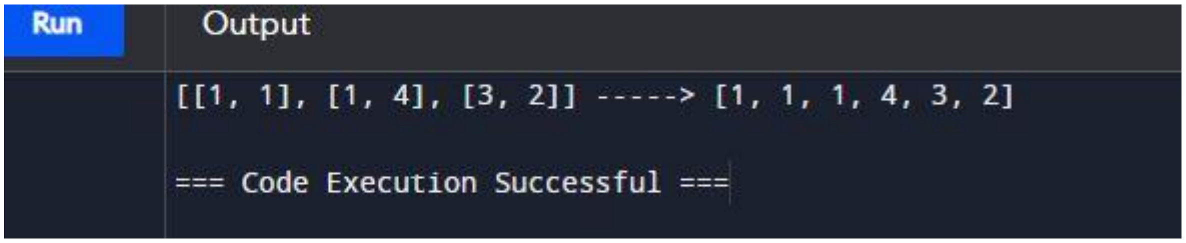
import itertools

import itertools

lis1 = [[1,1],[1,4],[3,2]]

flat_list = list(itertools.chain(*lis1))

print(lis1, "----->", flat_list)


Run

Output

```
[[1, 1], [1, 4], [3, 2]] -----> [1, 1, 1, 4, 3, 2]

=== Code Execution Successful ===
```

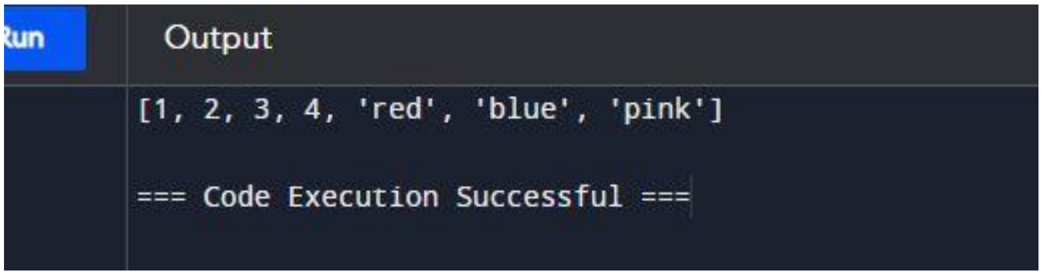
10. Write a Python program to append a list to the second list.

lis1 = [1, 2, 3, 4]

lis2 = ['red', 'blue', 'pink']

final_list=lis1 + lis2

print(final_list)

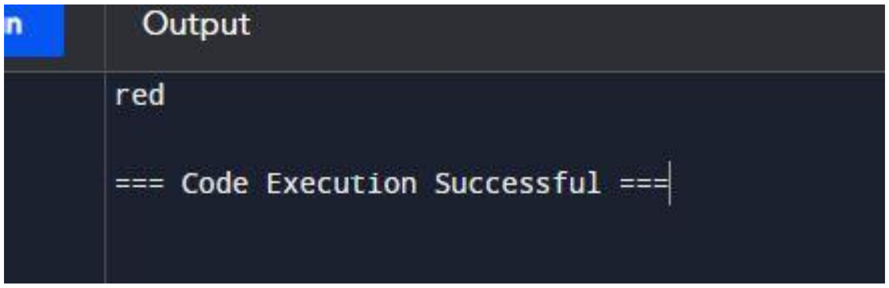

Run

Output

```
[1, 2, 3, 4, 'red', 'blue', 'pink']

=== Code Execution Successful ===
```

11. Write a Python program to select an item randomly from a list.

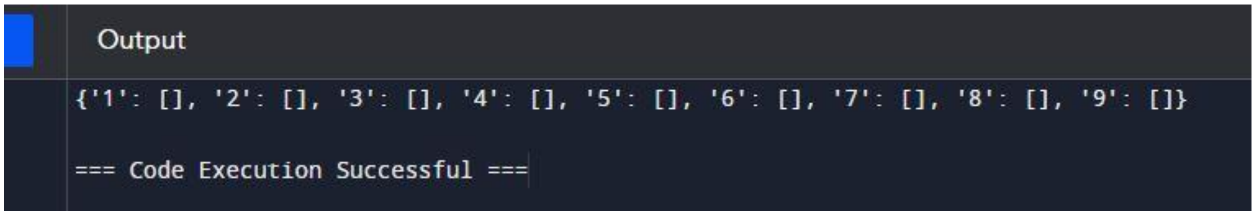
```
import random

lis2 = ['red', 'blue', 'pink']

print(random.choice(lis2))
```



```
red

=== Code Execution Successful ===
```

12. Write a Python program to create multiple lists.

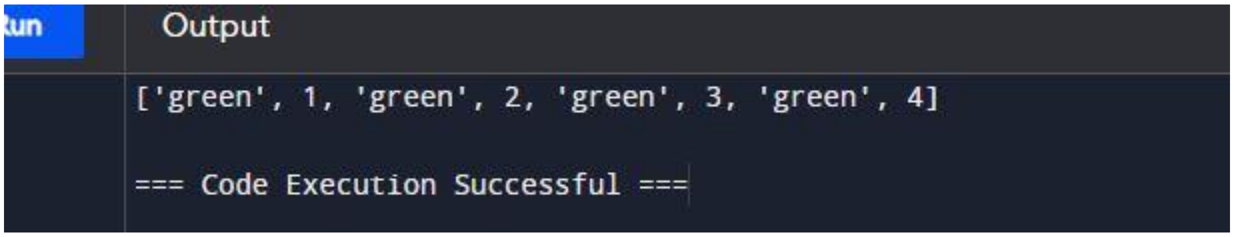
```
lis1 = {}

for i in range(1, 10):

   lis1[str(i)] = []

print(lis1)
```



```
{'1': [], '2': [], '3': [], '4': [], '5': [], '6': [], '7': [], '8': [], '9': []}

=== Code Execution Successful ===
```

13. Write a Python program to insert an element before each element of a list.

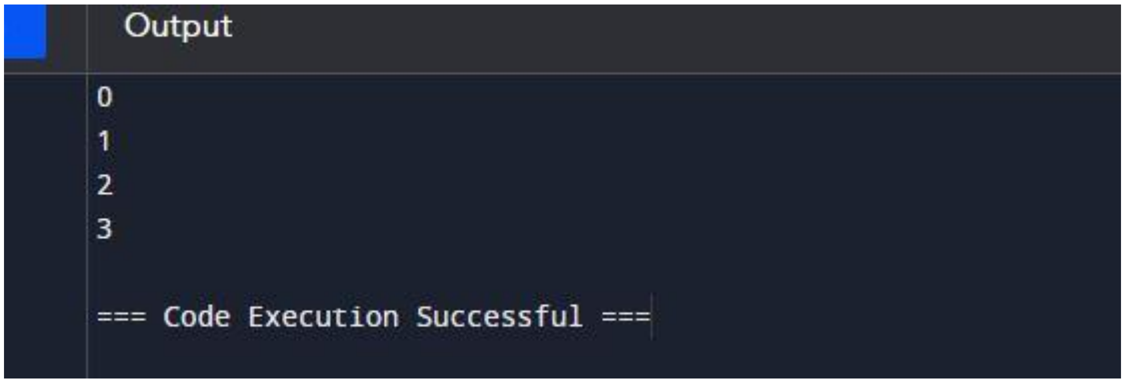
```
lis1 = [1,2,3,4]

lis1=[v for elt in lis1 for v in ('green', elt)]

print(lis1)
```



```
['green', 1, 'green', 2, 'green', 3, 'green', 4]

=== Code Execution Successful ===
```

14. Write a Python program to create a list with infinite elements.

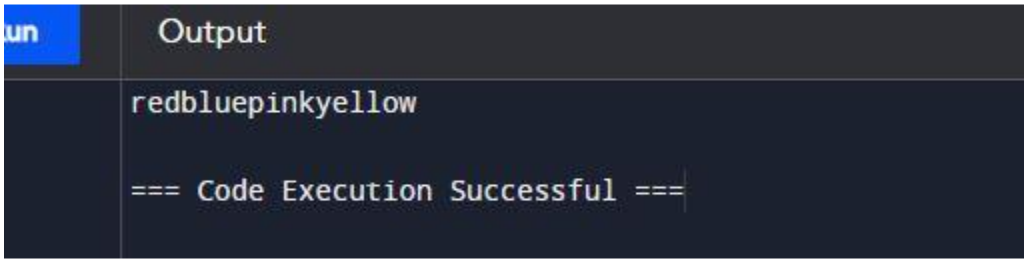
```
import itertools


lis1 = itertools.count()


print(next(lis1))
print(next(lis1))
print(next(lis1))
print(next(lis1))
```


Output

```
0
1
2
3

=== Code Execution Successful ===
```

15. Write a Python program to concatenate elements of a list.

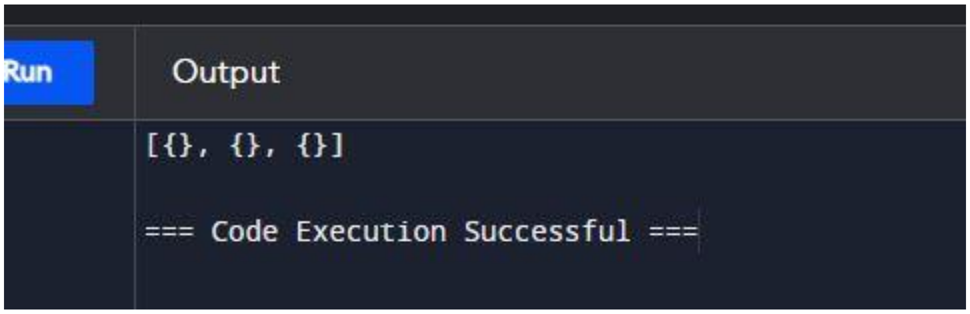
```
lis1=('red', 'blue', 'pink', 'yellow')
print('' .join(lis1))
```


Output

```
redbluepinkyellow

=== Code Execution Successful ===
```

16. Write a Python program to create a list of empty dictionaries.

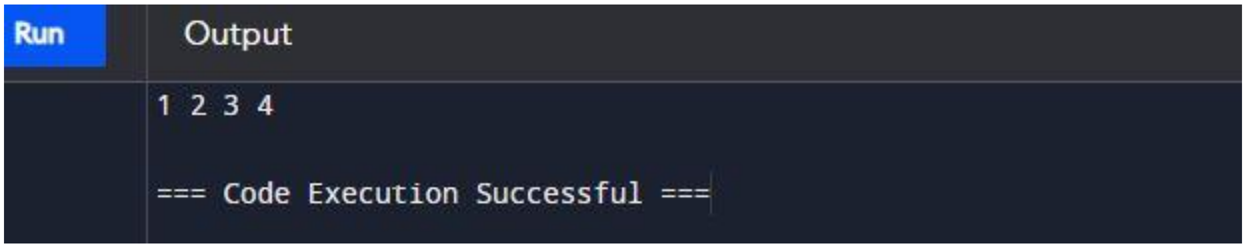
```
n = 3
lis1 = [{} for _ in range(n)]
print(lis1)
```


Run
Output

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

=== Code Execution Successful ===
```

17. Write a Python program to print a list of space-separated elements.

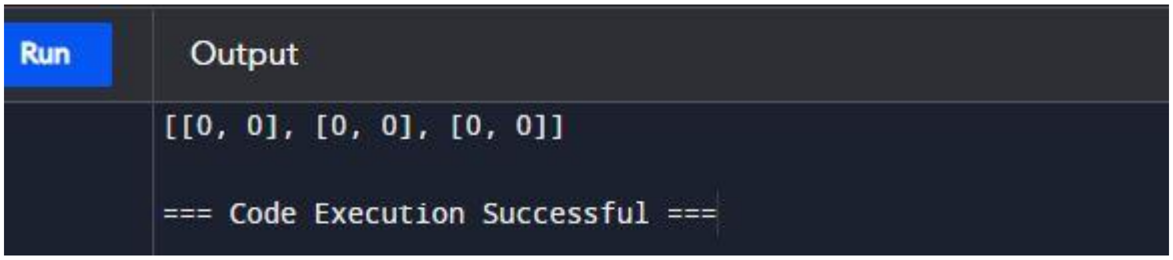
```
lis1=(1, 2, 3, 4)

print(*lis1)
```


Run
Output

```
1 2 3 4

=== Code Execution Successful ===
```

18. Write a Python program to create a multidimensional list (lists of lists) with zeros.

```
lis1 = []

for i in range(3):

    lis1.append([])

    for j in range(2):

        lis1[i].append(0)

print(lis1)
```

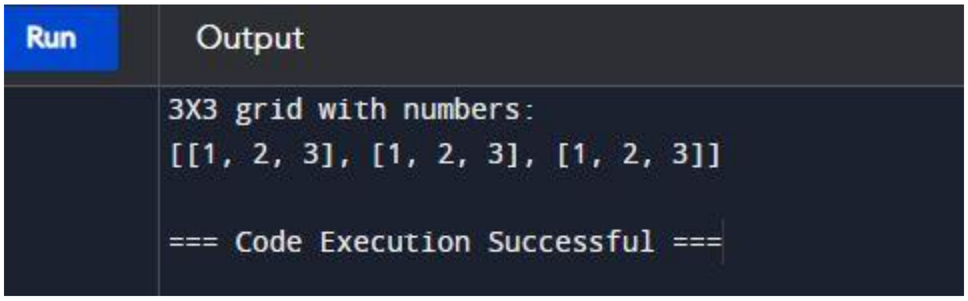

Run
Output

```
[[0, 0], [0, 0], [0, 0]]

=== Code Execution Successful ===
```

19. Write a Python program to create a 3X3 grid with numbers.

```
nums = []
```

```
for i in range(3):
    nums.append([])
    for j in range(1, 4):
        nums[i].append(j)
print(nums)
```



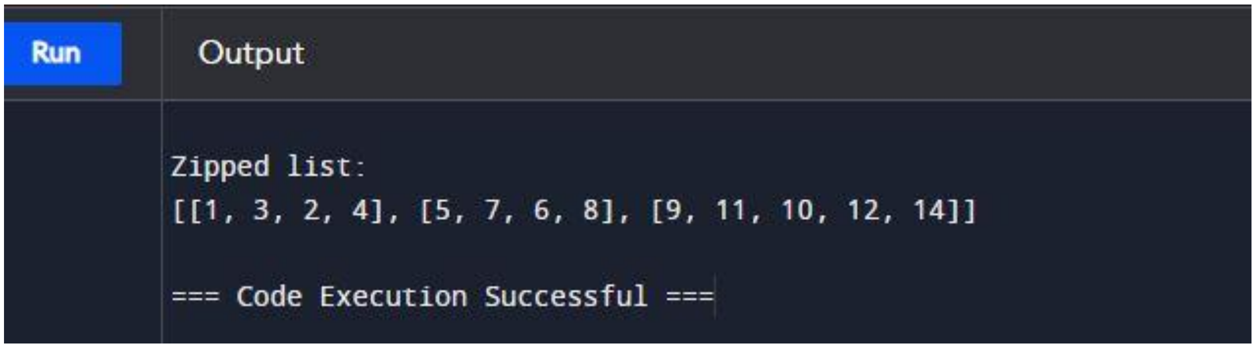
Run

Output

```
3X3 grid with numbers:
[[1, 2, 3], [1, 2, 3], [1, 2, 3]]

=== Code Execution Successful ===
```

20. Write a Python program to Zip two given lists of lists.

```
lis1=[[1, 3], [5, 7], [9, 11]]
lis2=[[2, 4], [6, 8], [10, 12, 14]]
result = list(map(list.__add__, lis1, lis2))
print("\nZipped list:\n" +  str(result))
```



Run

Output

```

Zipped list:
[[1, 3, 2, 4], [5, 7, 6, 8], [9, 11, 10, 12, 14]]

=== Code Execution Successful ===
```

ZAD.3

1. Write a Python script to sort (ascending and descending) a dictionary by value.

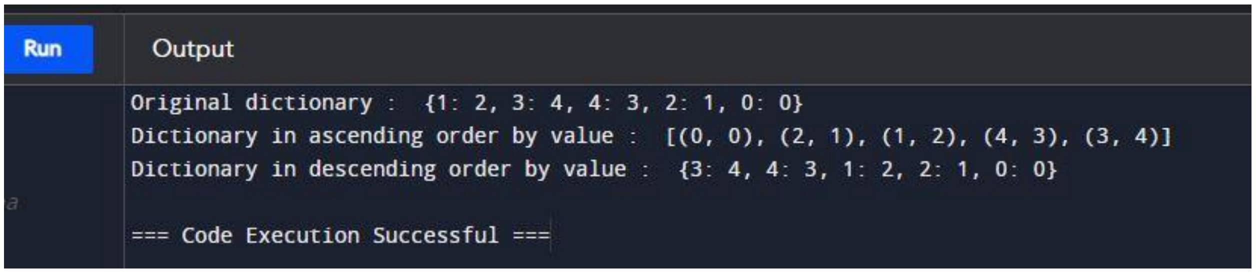
```
import operator
d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}


print('Original : ',d)
```

sorted_d = sorted(d.items(), key=operator.itemgetter(1))

print('Dictionary in ascending order by value : ',sorted_d)

sorted_d = dict(sorted(d.items(), key=operator.itemgetter(1), reverse=True))

print('Dictionary in descending order by value : ',sorted_d)



```
Original dictionary :  {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
Dictionary in ascending order by value :  [(0, 0), (2, 1), (1, 2), (4, 3), (3, 4)]
Dictionary in descending order by value :  {3: 4, 4: 3, 1: 2, 2: 1, 0: 0}

=== Code Execution Successful ===
```

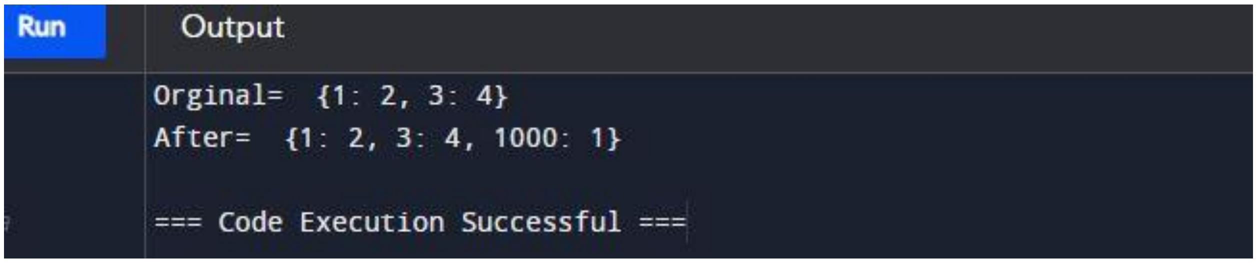
2. Write a Python script to add a key to a dictionary.

d = {1: 2, 3: 4,}

print("Orginal= ", d)

d.update({1000:1})

print("After= ", d)



```
Orginal=  {1: 2, 3: 4}
After=  {1: 2, 3: 4, 1000: 1}

=== Code Execution Successful ===
```

3. Write a Python script to concatenate the following dictionaries to create a new one.

d1 = {1: 2, 3: 4,}

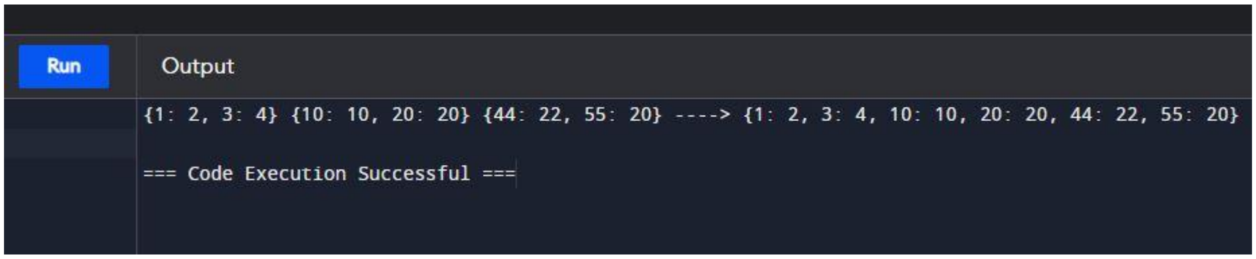
d2 = {10:10,20:20}

d3 = {44:22, 55:20}

d4={}

for d in (d1, d2, d3): d4.update(d)

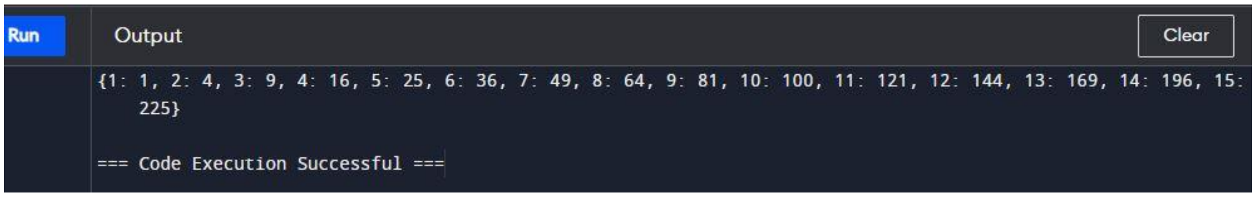
print(d1, d2, d3, "---->",d4)



```
{1: 2, 3: 4} {10: 10, 20: 20} {44: 22, 55: 20} ----> {1: 2, 3: 4, 10: 10, 20: 20, 44: 22, 55: 20}

=== Code Execution Successful ===
```

4. Write a Python script to print a dictionary where the keys are numbers between 1 and 15 (both included) and the values are the square of the keys.

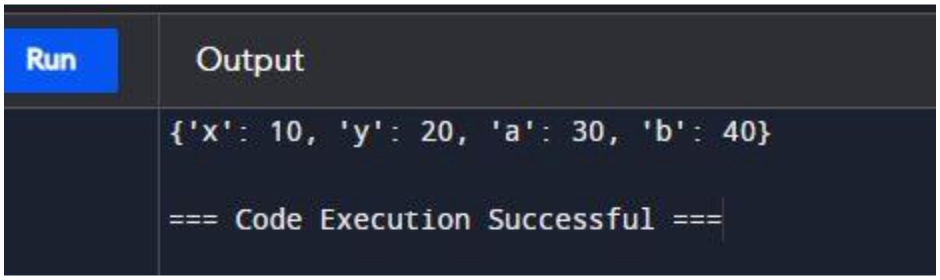
```
d1= {}

for x in range(1, 16):

   d1[x] = x**2

print(d1)
```



```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121, 12: 144, 13: 169, 14: 196, 15:
   225}

=== Code Execution Successful ===
```

5. Write a Python script to merge two Python dictionaries.

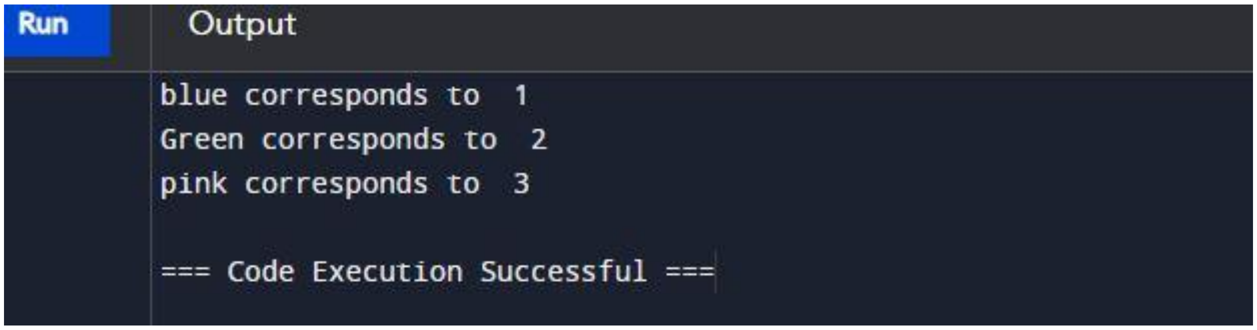
```
d1 = {'x':10, 'y':20}

d2 = {'a':30, 'b':40}

d3=d1.copy()

d3.update(d2)

print(d3)
```



```
{'x': 10, 'y': 20, 'a': 30, 'b': 40}

=== Code Execution Successful ===
```

6. Write a Python program to iterate over dictionaries using for loops.

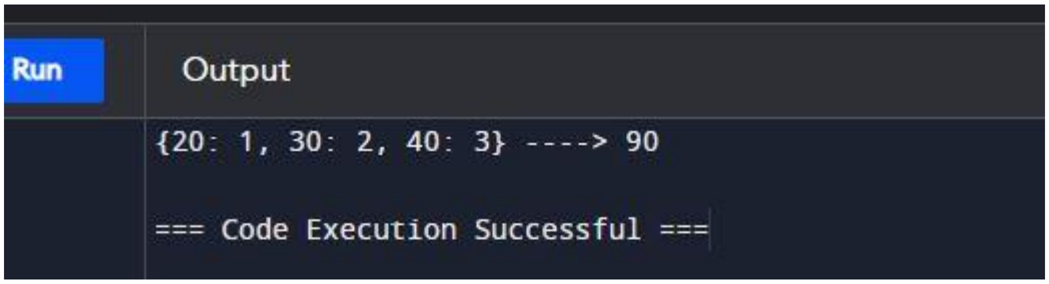
```
d = {'blue': 1, 'Green': 2, 'pink': 3}
for color_key, value in d.items():
    print(color_key, 'corresponds to ', d[color_key])
```



```
blue corresponds to  1
Green corresponds to  2
pink corresponds to  3

=== Code Execution Successful ===
```

7. Write a Python program to sum all the items in a dictionary.

```
d1={20:1, 30:2, 40:3}
print(d1,"---->", sum(d1))
```



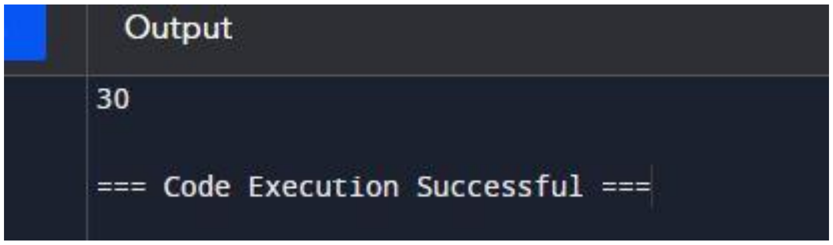
```
{20: 1, 30: 2, 40: 3} ----> 90

=== Code Execution Successful ===
```

8. Write a Python program to multiply all the items in a dictionary.

```
d1 = {'data1': 5, 'data2': 2, 'data3': 3}
result = 1
```

for key in d1:

    result = result * d1[key]



```
30

=== Code Execution Successful ===
```

9. Write a Python program to remove a key from a dictionary.

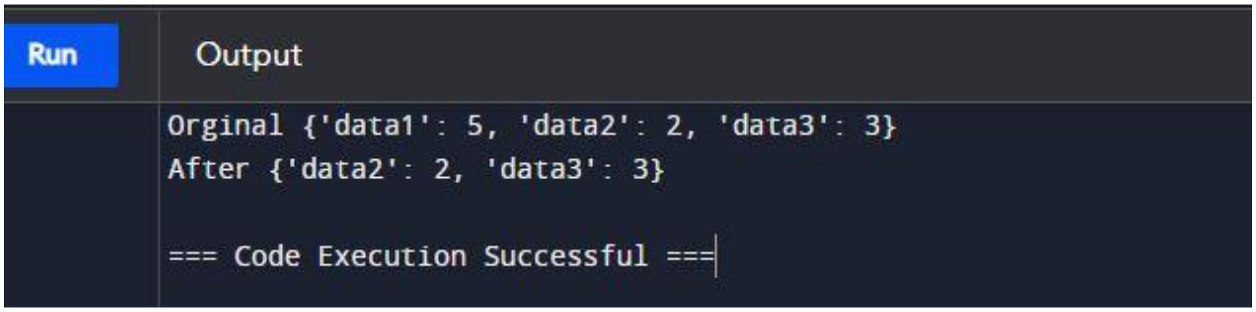
d1 = {'data1': 5, 'data2': 2, 'data3': 3}

print("Orginal",d1)

del d1['data1']

print("After", d1)

print(result)



```
Orginal {'data1': 5, 'data2': 2, 'data3': 3}
After {'data2': 2, 'data3': 3}

=== Code Execution Successful ===
```

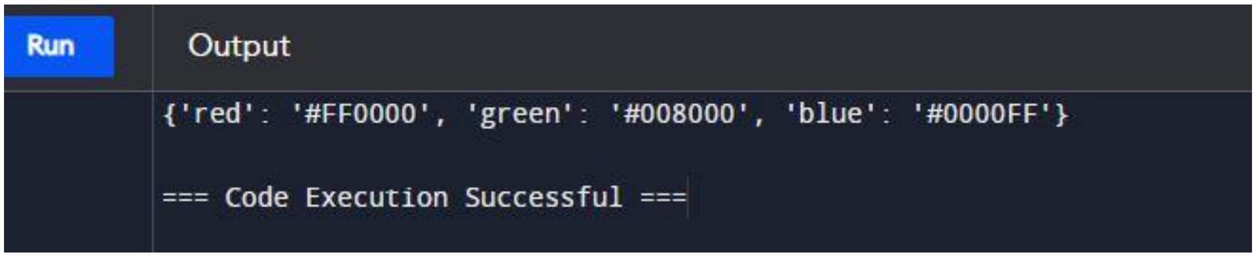
10. Write a Python program to map two lists into a dictionary.

keys = ['red', 'green', 'blue']

values = ['#FF0000', '#008000', '#0000FF']

color = dict(zip(keys, values)

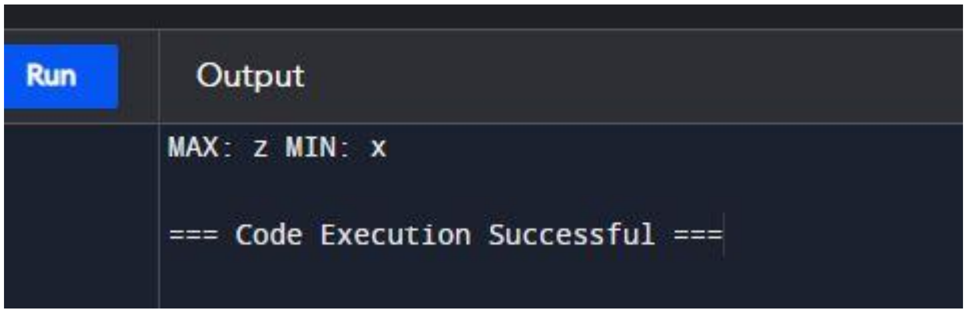
print(color)



```
{'red': '#FF0000', 'green': '#008000', 'blue': '#0000FF'}

=== Code Execution Successful ===
```

11. Write a Python program to get the maximum and minimum values of a dictionary.

```
d1={'x':10, 'y':20, 'z':30}
max_d1 = max(d1.keys())
min_d1 = min(d1.keys())

print("MAX:", max_d1, "MIN:", min_d1 )
```

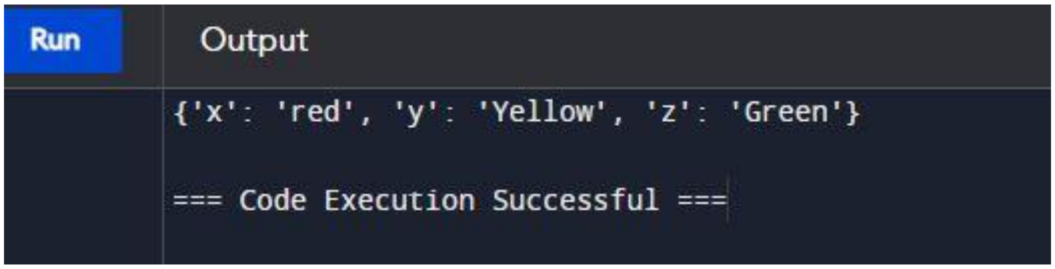


```
MAX: z MIN: x

=== Code Execution Successful ===
```

12. Write a Python program to get a dictionary from an object's fields.

```
class dictObj(object):
  def __init__(self):
    self.x = 'red'
    self.y = 'Yellow'
    self.z = 'Green'

  def do_nothing(self):
    pass

test = dictObj()
```

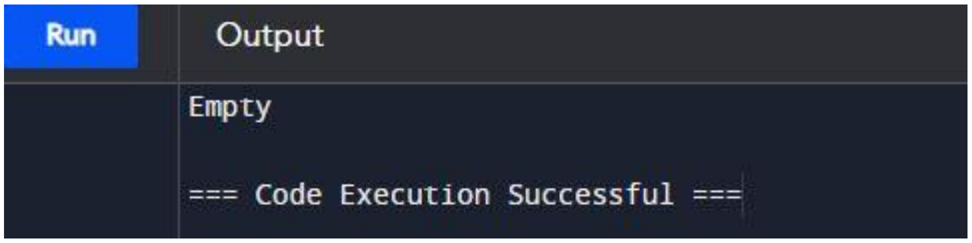
print(test.__dict__)


Run Output

```
{'x': 'red', 'y': 'Yellow', 'z': 'Green'}

=== Code Execution Successful ===
```

13. Write a Python program to check if a dictionary is empty or not.

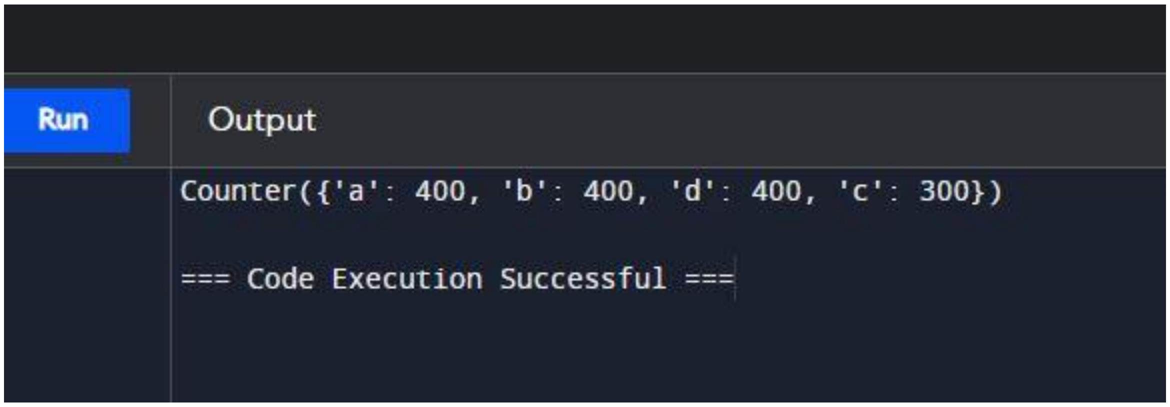
```
d1 = {}


if not bool(d1):
   print("Empty")
```


Run Output

```
Empty

=== Code Execution Successful ===
```

14. Write a Python program to combine two dictionary by adding values for common keys.

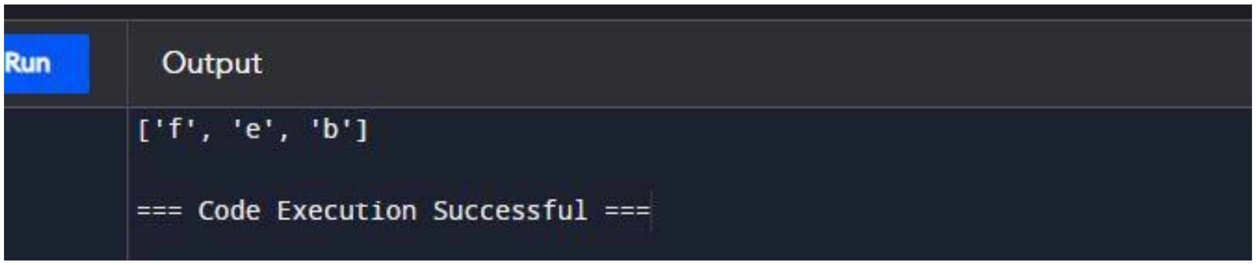
```
from collections import Counter


d1 = {'a': 100, 'b': 200, 'c': 300}
d2 = {'a': 300, 'b': 200, 'd': 400}
d = Counter(d1) + Counter(d2)
print(d)
```



```
Counter({'a': 400, 'b': 400, 'd': 400, 'c': 300})

=== Code Execution Successful ===
```

15. Write a Python program to find the highest 3 values of corresponding keys in a dictionary.

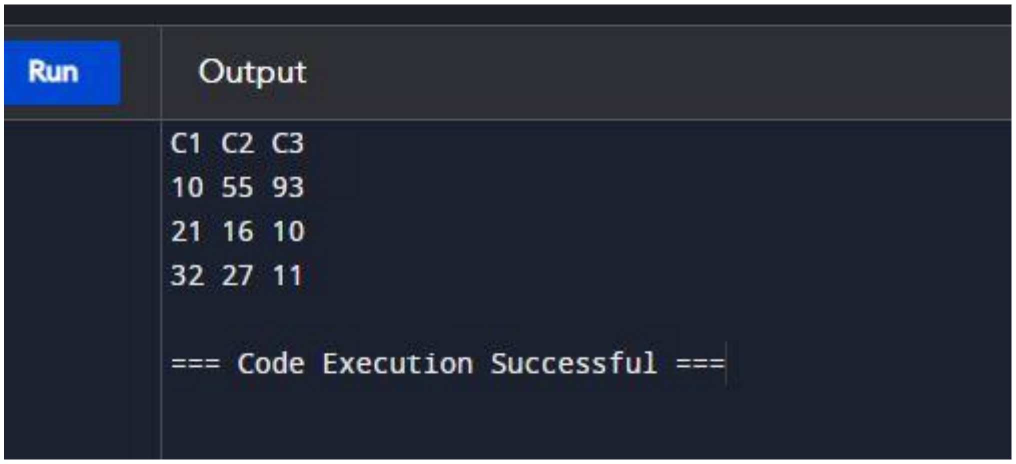
```
from heapq import nlargest
d1 = {'a': 1, 'b': 100, 'c': 2, 'd': 3, 'e': 400, 'f': 460}
three_largest = nlargest(3, d1, key=d1.get)
print(three_largest)
```



```
['f', 'e', 'b']

=== Code Execution Successful ===
```

16. Write a Python program to print a dictionary in table format.

```
my_dict = {'C1': [10, 21, 32], 'C2': [55, 16, 27], 'C3': [93, 10, 11]}


for row in zip(*([key] + (value) for key, value in sorted(my_dict.items()))):
    print(*row)
```


Run

Output

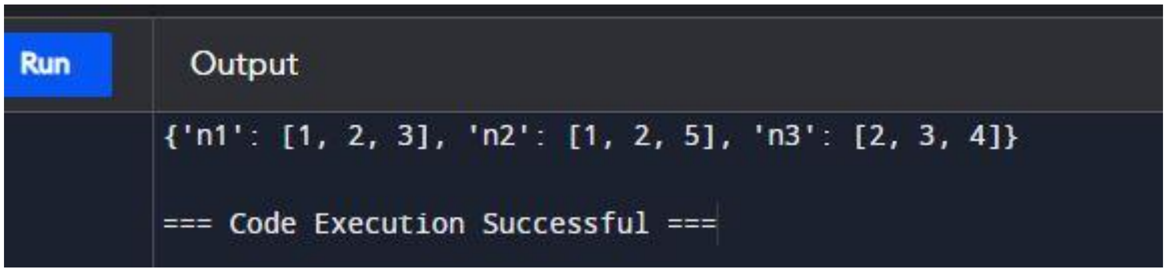
```
C1 C2 C3
10 55 93
21 16 10
32 27 11

=== Code Execution Successful ===
```

17. Write a Python program to sort a list alphabetically in a dictionary.

d1 = {'n1': [2, 3, 1], 'n2': [5, 1, 2], 'n3': [3, 2, 4]}

sorted_dict = {x: sorted(y) for x, y in d1.items()}

print(sorted_dict)


Run

Output

```
{'n1': [1, 2, 3], 'n2': [1, 2, 5], 'n3': [2, 3, 4]}

=== Code Execution Successful ===
```

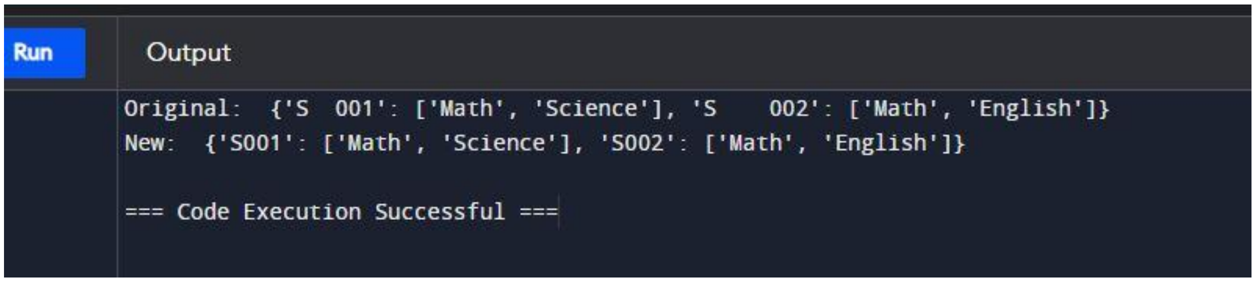
18. Write a Python program to remove spaces from dictionary keys.

student_list = {'S 001': ['Math', 'Science'], 'S   002': ['Math', 'English']}

print("Original: ", student_list)

student_dict = {x.translate({32: None}): y for x, y in student_list.items()}

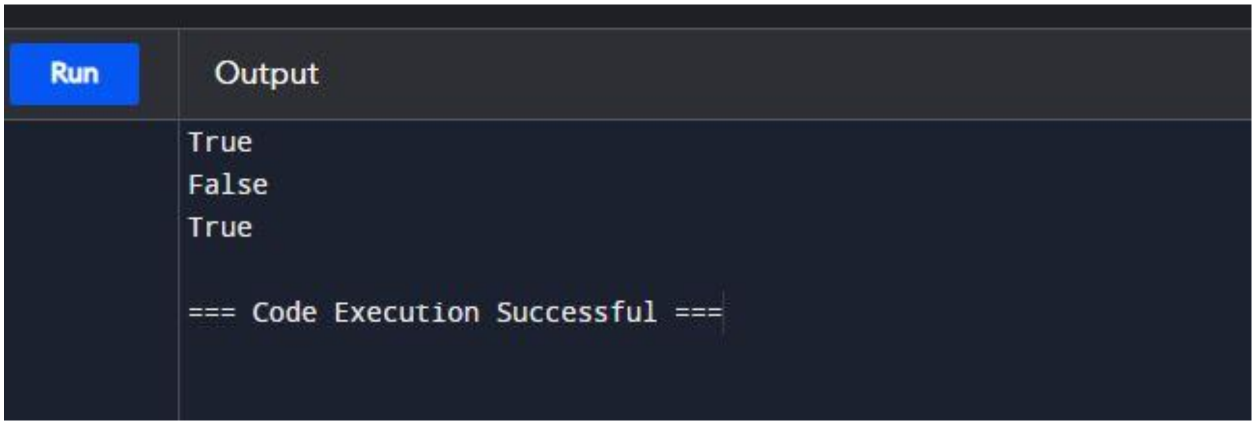
print("New: ", student_dict)



```
Original:  {'S  001': ['Math', 'Science'], 'S    002': ['Math', 'English']}
New:  {'S001': ['Math', 'Science'], 'S002': ['Math', 'English']}

=== Code Execution Successful ===
```

19. Write a Python program to check if multiple keys exist in a dictionary.

```
student = {
  'name': 'Alex',
  'class': 'V',
  'roll_id': '2'
}

print(student.keys() >= {'class', 'name'})
print(student.keys() >= {'name', 'Alex'})
print(student.keys() >= {'roll_id', 'name'})
```

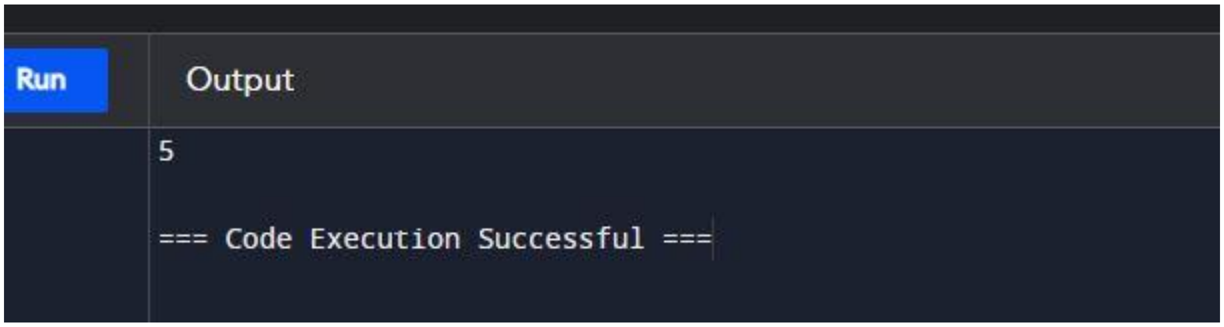


```
True
False
True

=== Code Execution Successful ===
```

20. Write a Python program to count the number of items in a dictionary value that is a list.

```
dict =  {'Alex': ['subj1', 'subj2', 'subj3'], 'David': ['subj1', 'subj2']}
ctr = sum(map(len, dict.values()))
```

```
print(ctr)
```

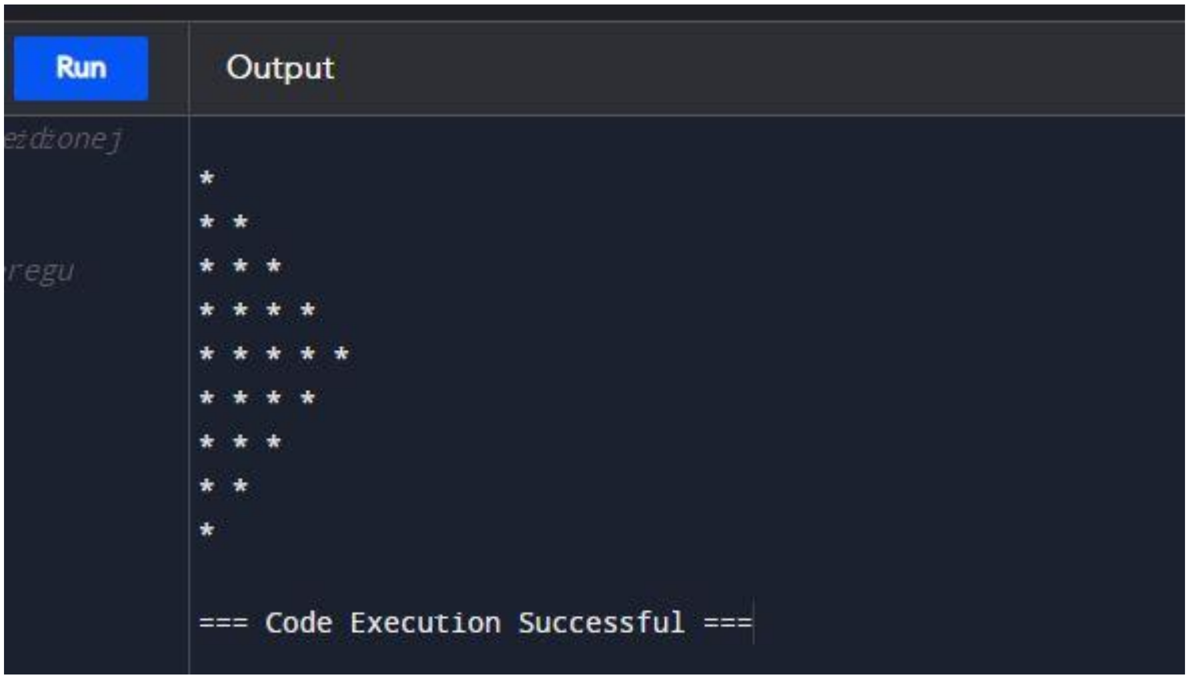


```
5

=== Code Execution Successful ===
```

ZAD.4

1. Write a Python program to construct the following pattern, using a nested for loop.

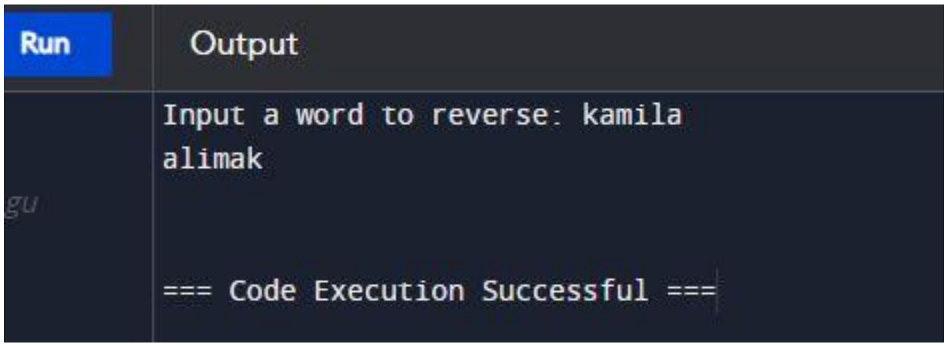
```
n = 5
for i in range(n):
    for j in range(i):
        print('* ', end="")
    print('')
for i in range(n, 0, -1):
    for j in range(i):
        print('* ', end="")
    print('')
```


Run Output

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*

=== Code Execution Successful ===
```

2. Write a Python program that accepts a word from the user and reverses it.

```
word = input("Input a word to reverse: ")
for char in range(len(word) - 1, -1, -1):
  print(word[char], end="")
print("\n")
```

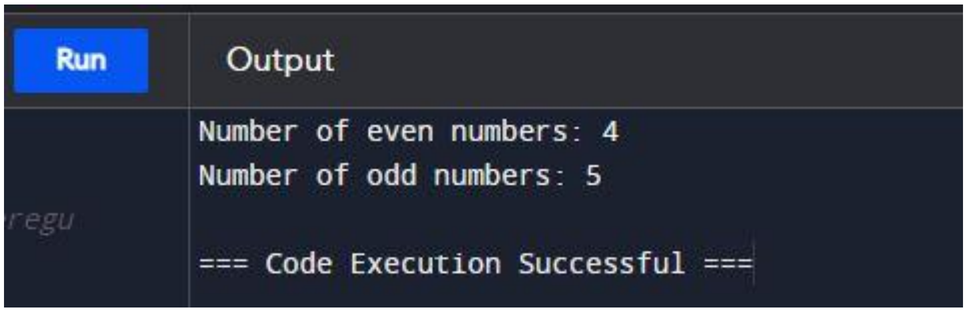

Run Output

```
Input a word to reverse: kamila
alimak


=== Code Execution Successful ===
```

3. Write a Python program to count the number of even and odd numbers in a series of numbers

```
numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9)
count_odd = 0
count_even = 0
```

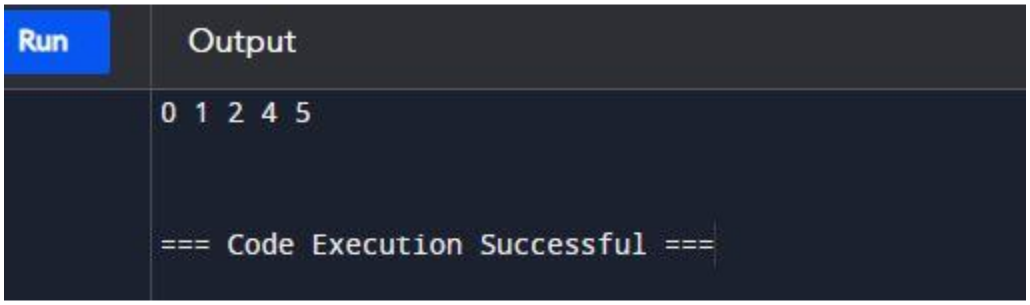
```
for x in numbers:
    if not x % 2:
        count_even += 1
    else:
        count_odd += 1
print("Number of even numbers:", count_even)
print("Number of odd numbers:", count_odd)
```



```
Number of even numbers: 4
Number of odd numbers: 5

=== Code Execution Successful ===
```

4. Write a Python program that prints all the numbers from 0 to 6 except 3 and 6.

```
for x in range(6):
    if (x == 3 or x == 6):
        continue
    print(x, end=' ')
print("\n")
```

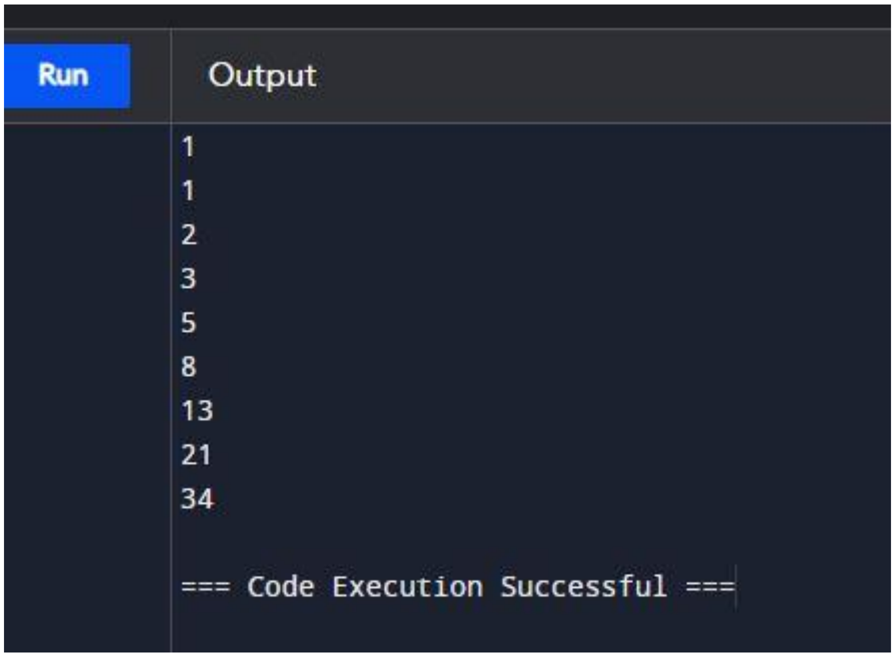


```
0 1 2 4 5


=== Code Execution Successful ===
```

5. Write a Python program to get the Fibonacci series between 0 and 50.

```
x, y = 0, 1
```

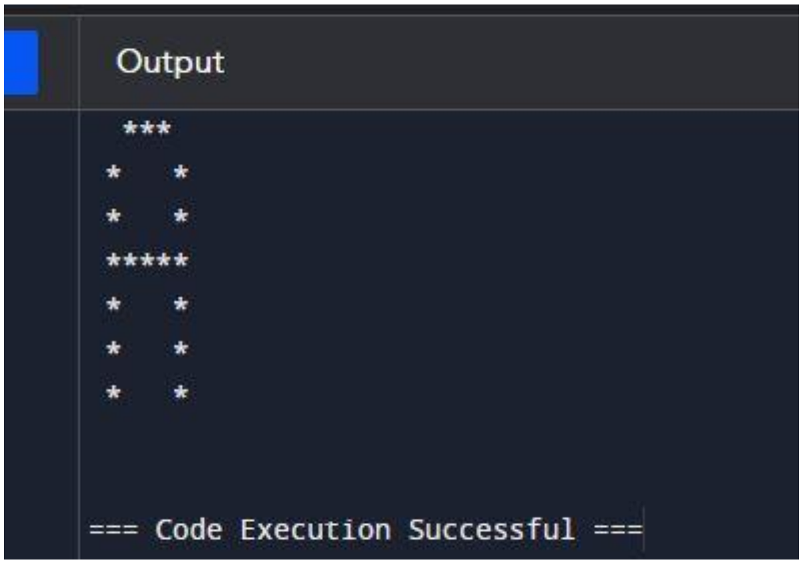
```
while y < 50:
    print(y)

    x, y = y, x + y
```



```
1
1
2
3
5
8
13
21
34

=== Code Execution Successful ===
```

6. Write a Python program to print the alphabet pattern 'A'.

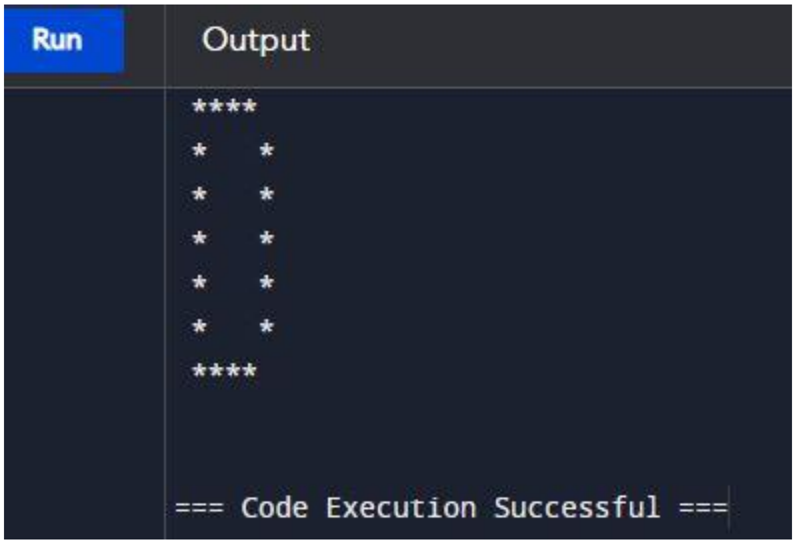
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (((column == 1 or column == 5) and row != 0) or ((row == 0 or row == 3) and (column > 1 and
column < 5))):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
  ***
 *   *
 *   *
 *****
 *   *
 *   *
 *   *



=== Code Execution Successful ===
```

7. Write a Python program to print the alphabet pattern 'D'.

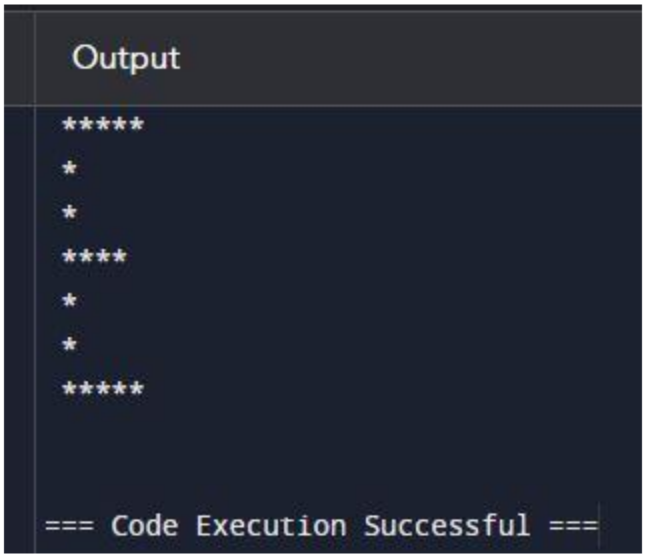
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 1 or ((row == 0 or row == 6) and (column > 1 and column < 5)) or (column == 5 and row != 0 and row != 6)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
****
*   *
*   *
*   *
*   *
*   *
****


=== Code Execution Successful ===
```

8. Write a Python program to print the alphabet pattern 'E'.

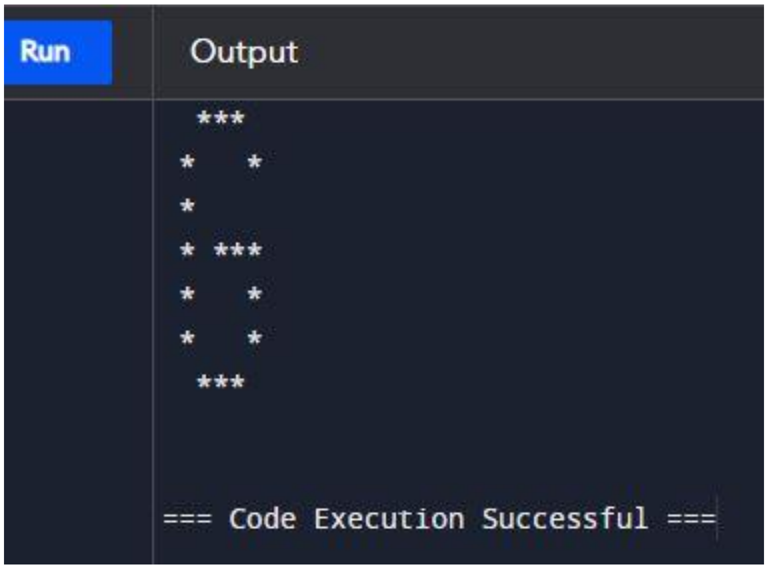
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 1 or ((row == 0 or row == 6) and (column > 1 and column < 6)) or (row == 3 and column > 1 and column < 5)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
*****
*
*
****
*
*
*****


=== Code Execution Successful ===
```

9. Write a Python program to print the alphabet pattern 'G'.

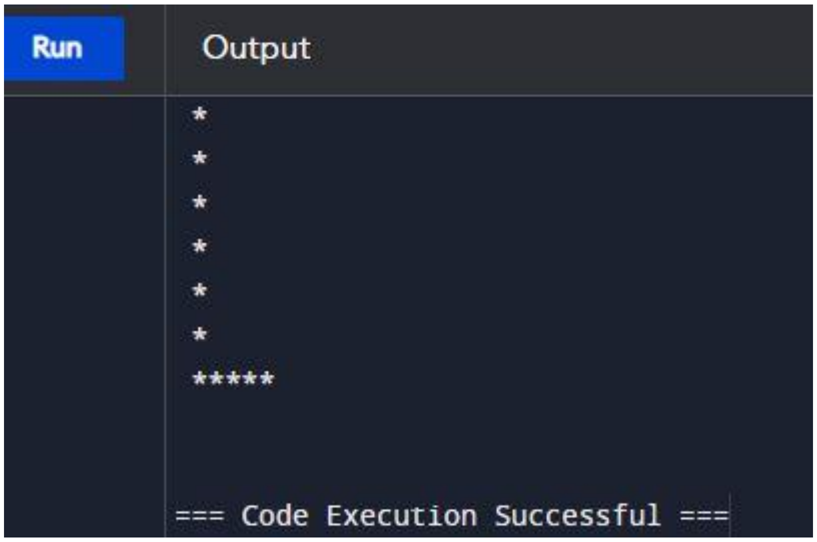
```
result_str = ""
for row in range(0, 7):
   for column in range(0, 7):
      if ((column == 1 and row != 0 and row != 6) or ((row == 0 or row == 6) and column > 1 and column < 5) or (row == 3 and column > 2 and column < 6) or (column == 5 and row != 0 and row != 2 and row != 6)):
         result_str = result_str + "*"
      else:
         result_str = result_str + " "
   result_str = result_str + "\n"
print(result_str)
```



```
 ***
*   *
*
* ***
*   *
*   *
 ***


=== Code Execution Successful ===
```

10. Write a Python program to print the alphabet pattern 'L'.

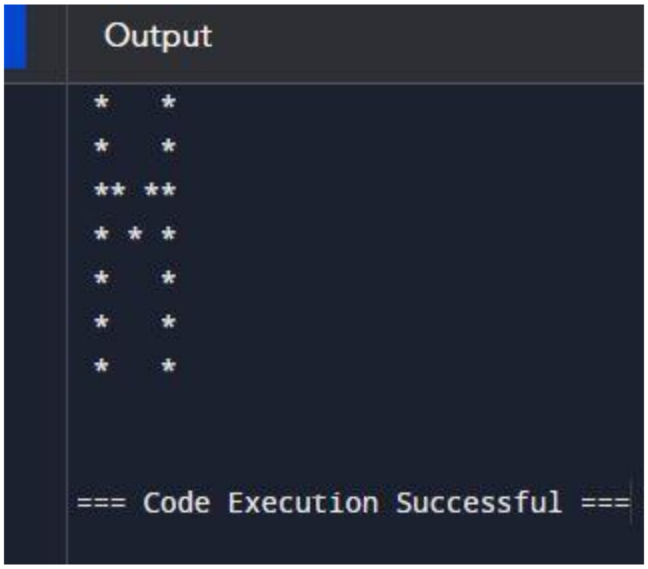
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 1 or (row == 6 and column != 0 and column < 6)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
*
*
*
*
*
*
*****


=== Code Execution Successful ===
```

11. Write a Python program to print the alphabet pattern 'M'.

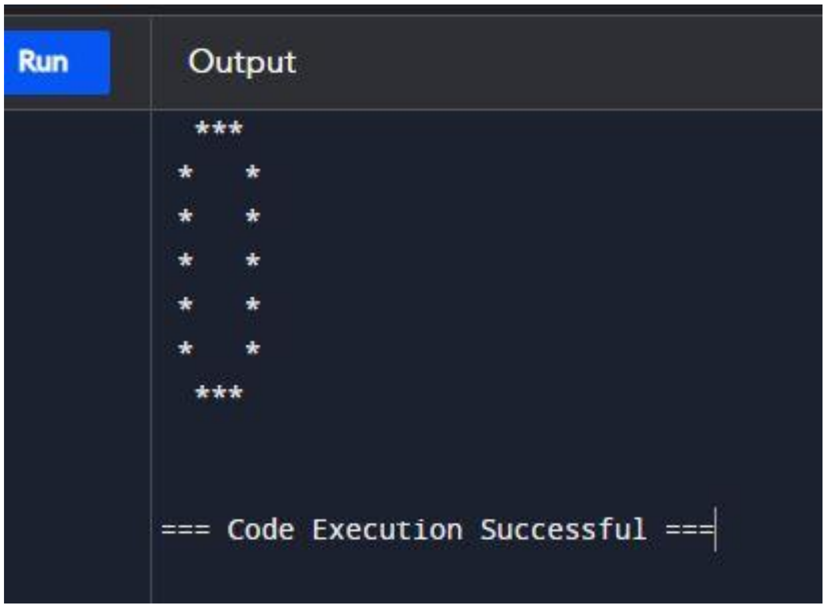
```
result_str = ""
for row in range(0, 7):
   for column in range(0, 7):
      if (column == 1 or column == 5 or (row == 2 and (column == 2 or column == 4)) or (row == 3 and column == 3)):
         result_str = result_str + "*"
      else:
         result_str = result_str + " "
   result_str = result_str + "\n"
print(result_str)
```


Output

```
*   *
*   *
** **
* * *
*   *
*   *
*   *


=== Code Execution Successful ===
```

12.Write a Python program to print the alphabet pattern 'O'.

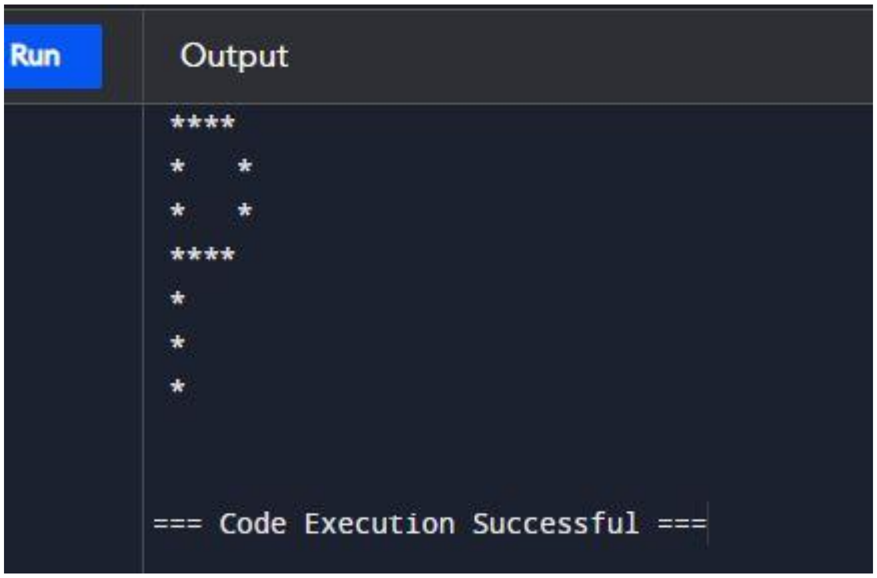
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (((column == 1 or column == 5) and row != 0 and row != 6) or ((row == 0 or row == 6) and column > 1 and column < 5)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
  ***
 *   *
 *   *
 *   *
 *   *
 *   *
  ***


=== Code Execution Successful ===
```

13. Write a Python program to print the alphabet pattern 'P'.

```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 1 or ((row == 0 or row == 3) and column > 0 and column < 5) or ((column == 5 or
column == 1) and (row == 1 or row == 2))):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```

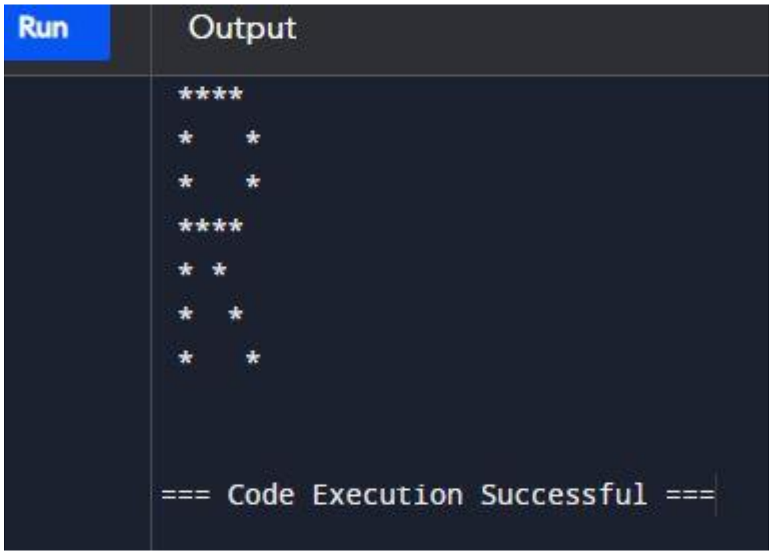

Run

Output

```
****
*   *
*   *
****
*
*
*


=== Code Execution Successful ===
```

14. Write a Python program to print the alphabet pattern 'R'.

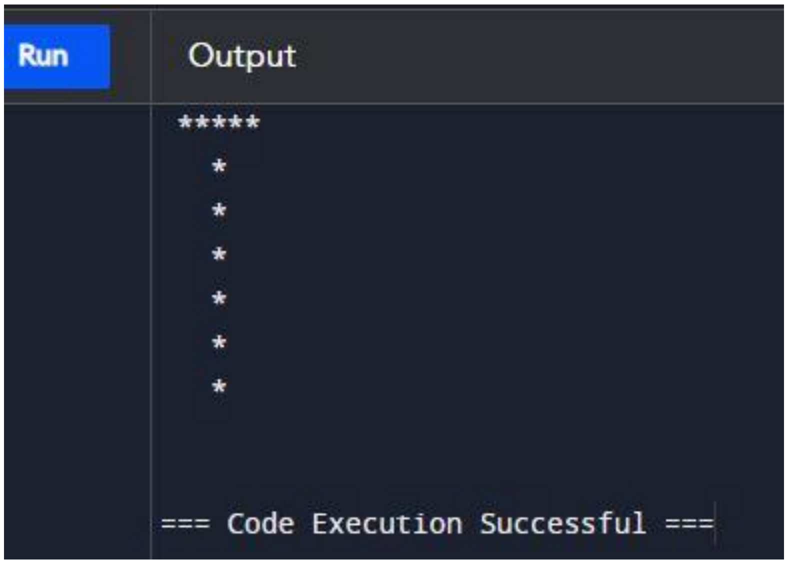
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 1 or ((row == 0 or row == 3) and column > 1 and column < 5) or (column == 5 and row != 0 and row < 3) or (column == row - 1 and row > 2)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
****
*   *
*   *
****
* *
*  *
*   *


=== Code Execution Successful ===
```

15. Write a Python program to print the alphabet pattern 'T'.

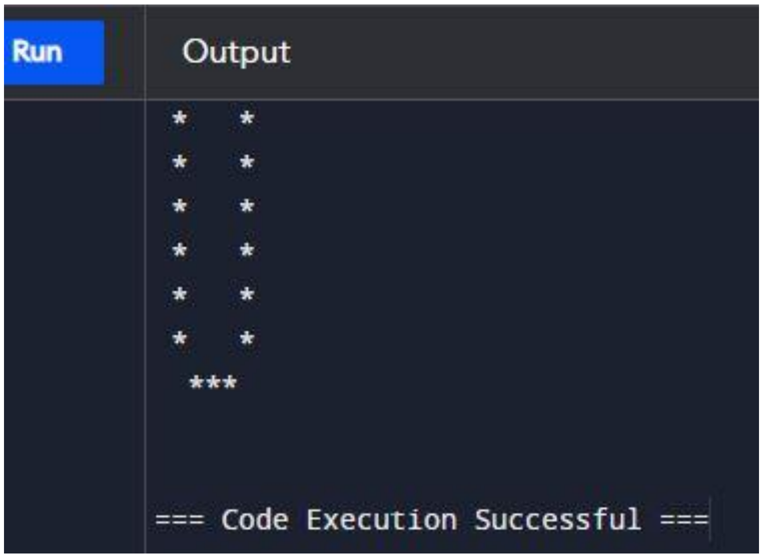
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (column == 3 or (row == 0 and column > 0 and column <6)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
*****
  *
  *
  *
  *
  *
  *


=== Code Execution Successful ===
```

16. Write a Python program to print the alphabet pattern 'U'.

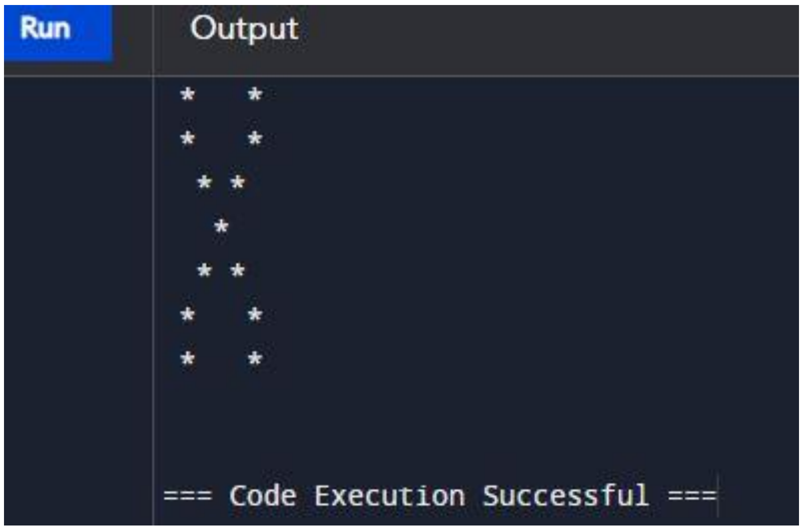
```
result_str = ""
for row in range(0, 7):
  for column in range(0, 7):
    if (((column == 1 or column == 5) and row != 6) or (row == 6 and column > 1 and column < 5)):
      result_str = result_str + "*"
    else:
      result_str = result_str + " "
  result_str = result_str + "\n"
print(result_str)
```



```
 *   *
 *   *
 *   *
 *   *
 *   *
 *   *
  ***


=== Code Execution Successful ===
```

17. Write a Python program to print the alphabet pattern 'X'.

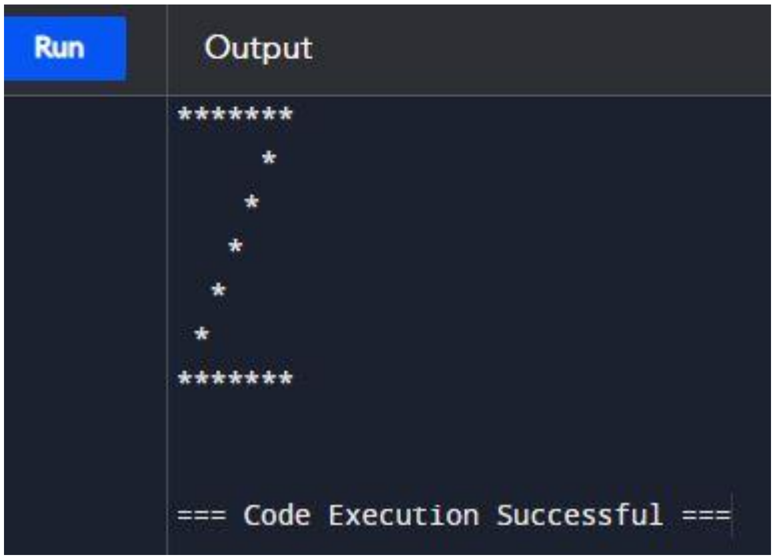
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (((column == 1 or column == 5) and (row > 4 or row < 2)) or
            row == column and column > 0 and column < 6 or
            (column == 2 and row == 4) or
            (column == 4 and row == 2)):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
 *   *
 *   *
  * *
   *
  * *
 *   *
 *   *


=== Code Execution Successful ===
```

18. Write a Python program to print the alphabet pattern 'Z'.

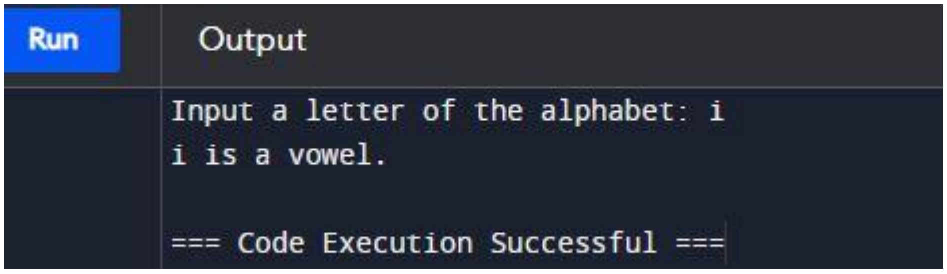
```
result_str = ""
for row in range(0, 7):
    for column in range(0, 7):
        if (((row == 0 or row == 6) and column >= 0 and column <= 6) or row + column == 6):
            result_str = result_str + "*"
        else:
            result_str = result_str + " "
    result_str = result_str + "\n"
print(result_str)
```



```
*******
     *
    *
   *
  *
 *
*******


=== Code Execution Successful ===
```

19. Write a Python program to check whether an alphabet is a vowel or consonant.

```
l = input("Input a letter of the alphabet: ")
if l in ('a', 'e', 'i', 'o', 'u'):
  print("%s is a vowel." % l)
elif l == 'y':
  print("Sometimes the letter y stands for a vowel, sometimes for a consonant.")
else:
  print("%s is a consonant." % l)
```

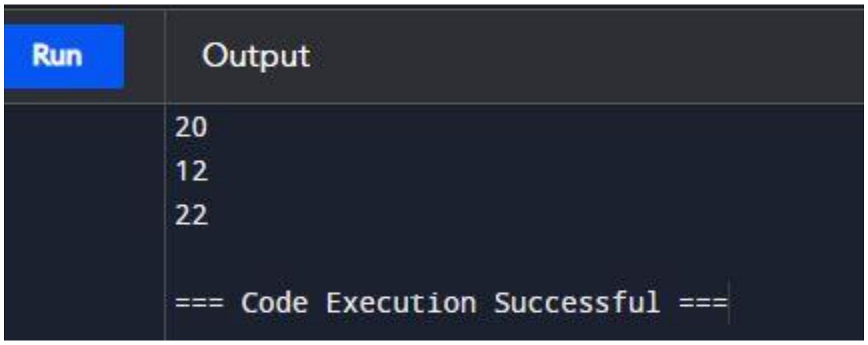


```
Input a letter of the alphabet: i
i is a vowel.

=== Code Execution Successful ===
```

20. Write a Python program to sum two integers. However, if the sum is between 15 and 20 it will return 20.

```
def sum(x, y):
  # Calculate the sum of 'x' and 'y' and assign it to the variable 'sum'
  sum = x + y
```

```
    # Check if the calculated sum falls within the range of 15 to 19 (inclusive)
    if sum in range(15, 20):
        return 20  # If the sum falls within the specified range, return 20
    else:
        return sum  # If the sum doesn't fall within the specified range, return the calculated sum

# Call the 'sum' function with different arguments and print the results
print(sum(10, 6))   # Call the function 'sum' with arguments 10 and 6, then print the result
print(sum(10, 2))   # Call the function 'sum' with arguments 10 and 2, then print the result
print(sum(10, 12))  # Call the function 'sum' with arguments 10 and 12, then print the result
```



```
20
12
22

=== Code Execution Successful ===
```