DAP

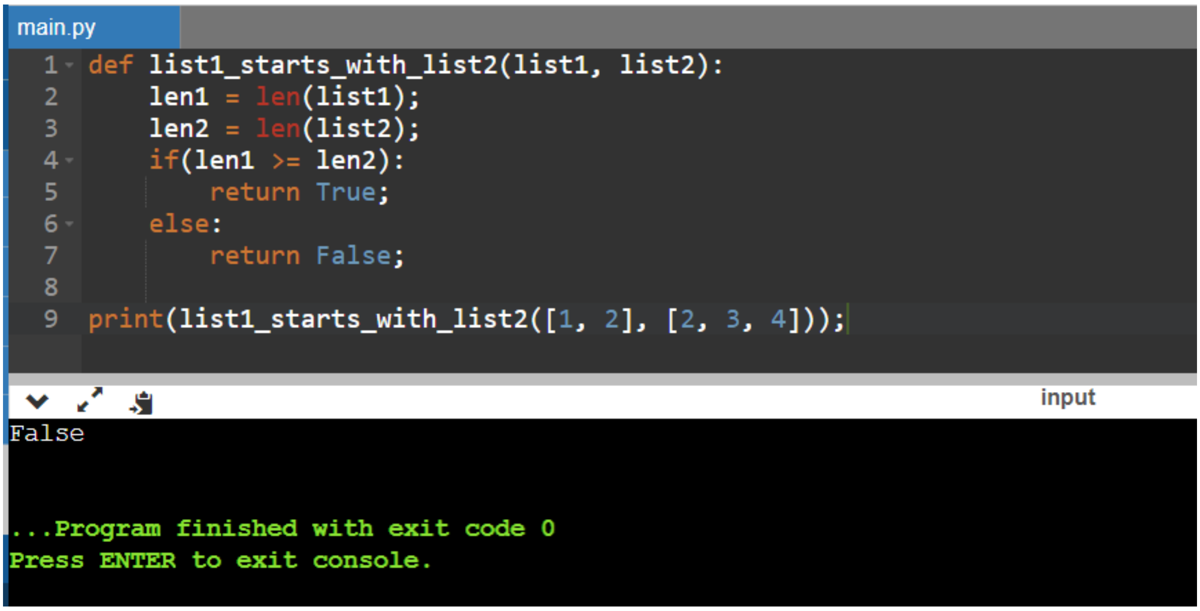
Assignment:01

Name:-**Chetan**

Roll no:-20001011502

Q1> Write a function with the signature def list1_starts_with_list2(list1, list2), which returns True iff list1 is at least as long as list2, and the first len(list2) elements of list1 are the same as list2. Note: len(lis) is the length of the list lis, i.e., the number of elements in lis. First write the function without using slicing ("slicing" means doing things like list1[2:5]), and using a loop.

Ans>



```
def list1_starts_with_list2(list1, list2):
    len1 = len(list1);
    len2 = len(list2);
    if(len1 >= len2):
        return True;
    else:
        return False;

print(list1_starts_with_list2([1, 2], [2, 3, 4]));
```

```
False


...Program finished with exit code 0
Press ENTER to exit console.
```

Q2> Write a function with the signature def match_pattern(list1, list2) which returns True iff the pattern list2 appears in list1. In other words, we return True iff there is an i such that 0≤i≤len(list1)-len(list2) and list1[i] = list2[0] list1[i + 1] =list2[1]

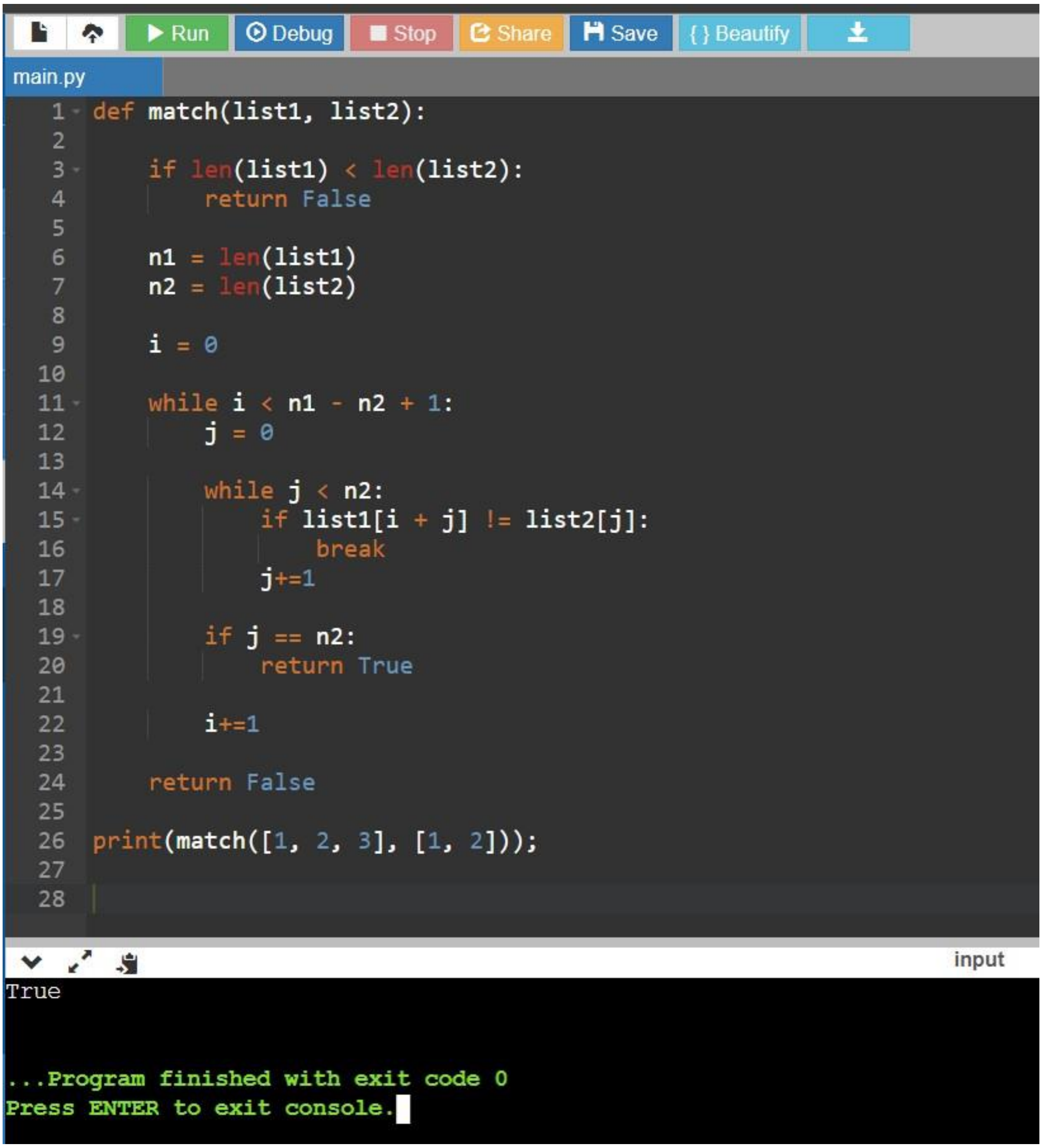
.

.

.

list1[i + len(list2) – 1] = list2[-1] For example, if list1 is [4, 10, 2, 3, 50, 100] and list2 is [2, 3, 50], match_pattern(list1, list2) returns True since the pattern [2, 3, 50] appears in list1

Ans

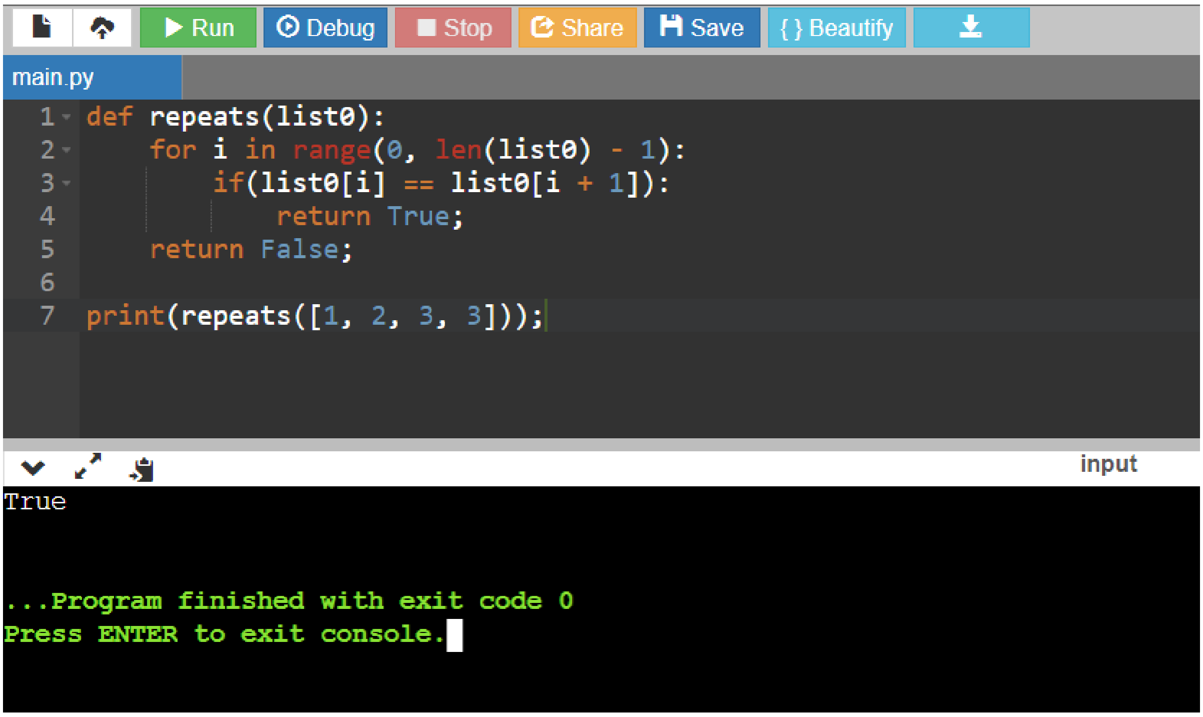


```python
def match(list1, list2):

    if len(list1) < len(list2):
        return False

    n1 = len(list1)
    n2 = len(list2)

    i = 0

    while i < n1 - n2 + 1:
        j = 0

        while j < n2:
            if list1[i + j] != list2[j]:
                break
            j+=1

        if j == n2:
            return True

        i+=1

    return False

print(match([1, 2, 3], [1, 2]));

```

```
True


...Program finished with exit code 0
Press ENTER to exit console.
```

Q3 > Write a function with the signature def repeats(list0), which returns True if list0 contains at least two adjacent elements with the same value.

Ans>

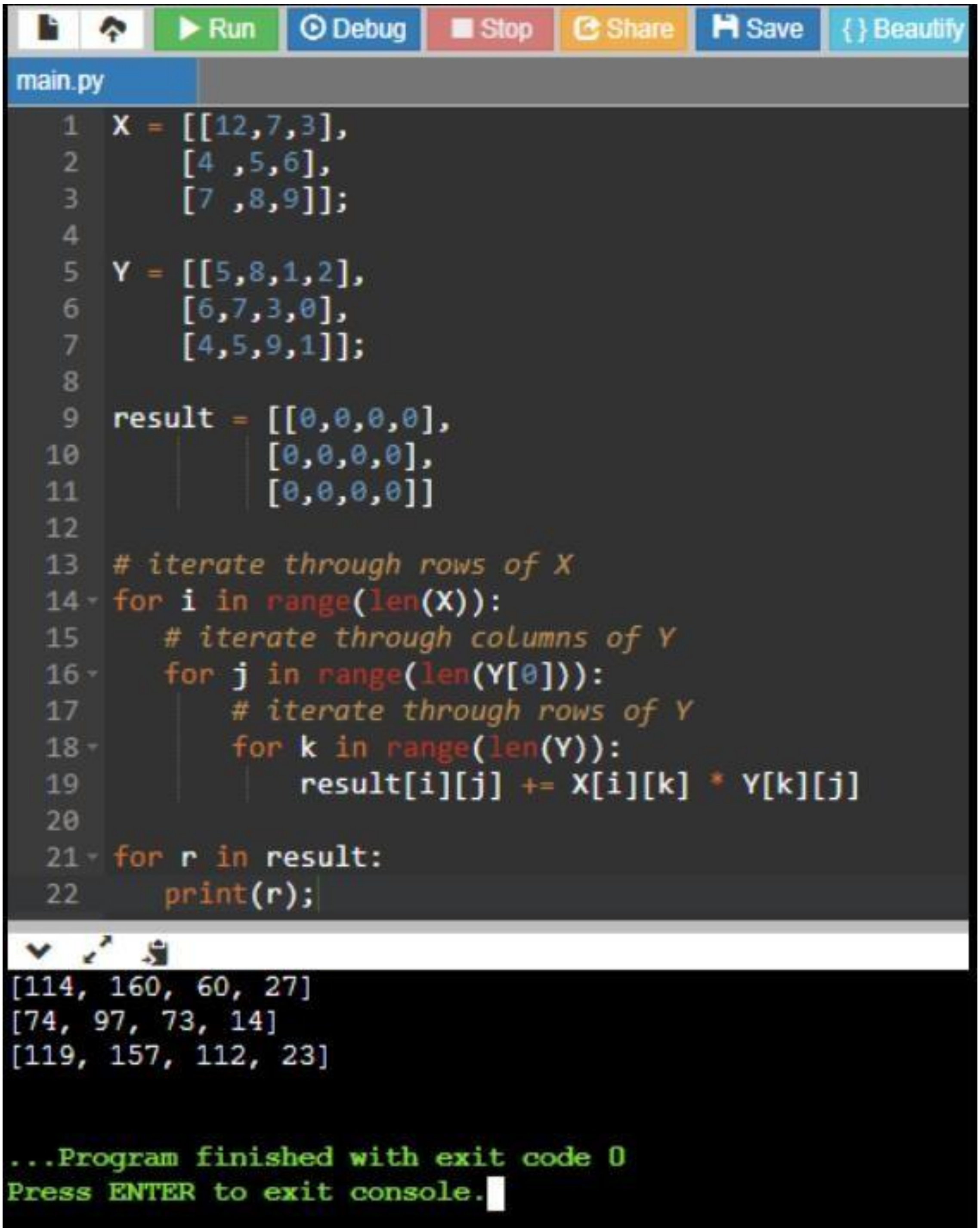


```python
def repeats(list0):
    for i in range(0, len(list0) - 1):
        if(list0[i] == list0[i + 1]):
            return True;
    return False;

print(repeats([1, 2, 3, 3]));
```

```
True


...Program finished with exit code 0
Press ENTER to exit console.
```

Q4 > Write a Python function to perform matrix multiplication.

Ans>



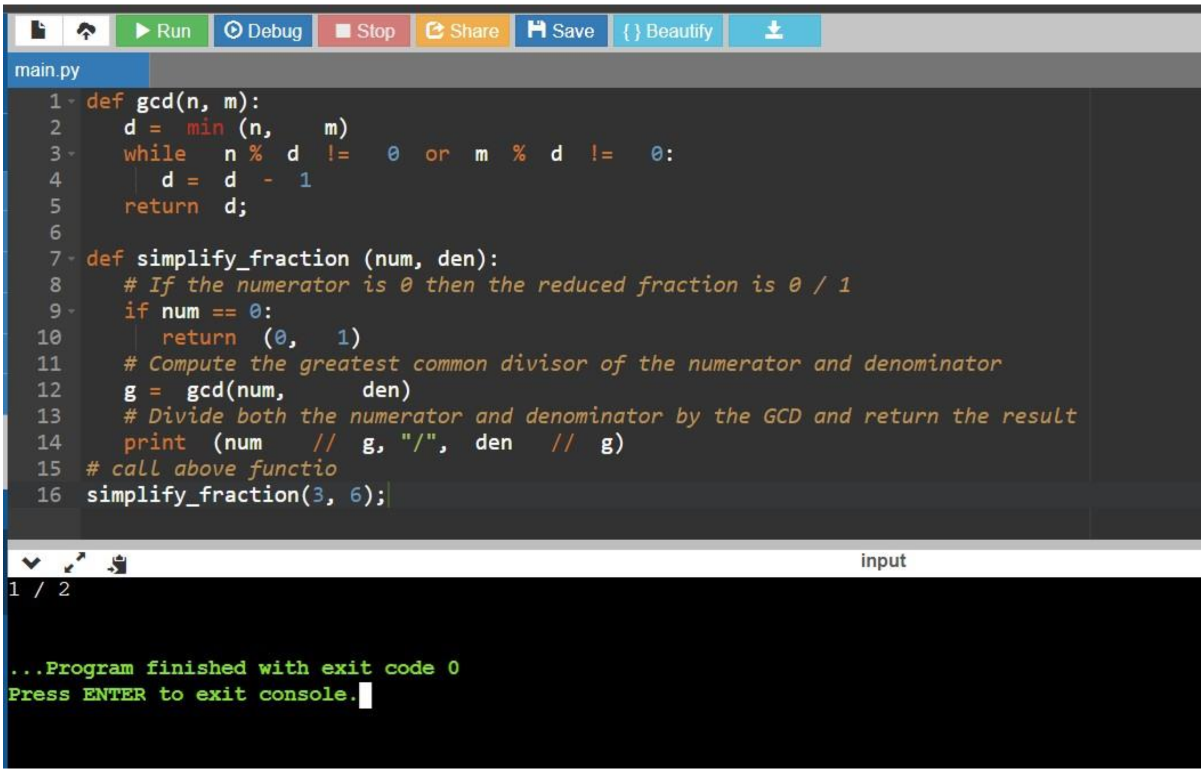
```
X = [[12,7,3],
     [4 ,5,6],
     [7 ,8,9]];

Y = [[5,8,1,2],
     [6,7,3,0],
     [4,5,9,1]];

result = [[0,0,0,0],
          [0,0,0,0],
          [0,0,0,0]]

# iterate through rows of X
for i in range(len(X)):
    # iterate through columns of Y
    for j in range(len(Y[0])):
        # iterate through rows of Y
        for k in range(len(Y)):
            result[i][j] += X[i][k] * Y[k][j]

for r in result:
    print(r);
```

```
[114, 160, 60, 27]
[74, 97, 73, 14]
[119, 157, 112, 23]


...Program finished with exit code 0
Press ENTER to exit console.
```

Q5 > Write a function with the signature simpify_fraction(n, m) which prints the simplified version of the fraction n/m.

For example, simplify_fraction(3,6) should print 1/2 and simplify_fraction(8, 4) should print 2/1 Hint: use a similar technique to the one we used when determining whether a number is prime. That is, try dividing both the numerator and the denominator by every possible divisor in turn. For example, if you are simplifying 16/12, you can try dividing both by 16, 15, 14, ..., 1.

Ans>

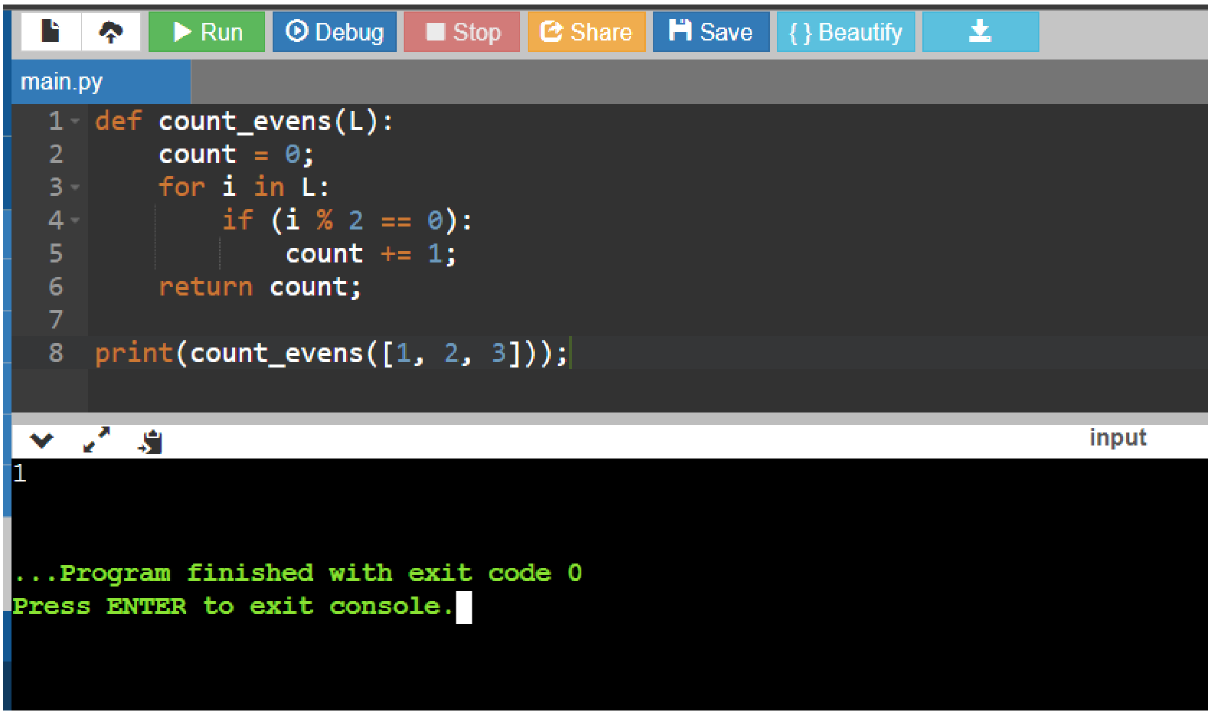


```
def gcd(n, m):
   d =  min (n,    m)
   while   n %  d  !=   0  or  m  %  d  !=   0:
      d =  d  -  1
   return  d;

def simplify_fraction (num, den):
   # If the numerator is 0 then the reduced fraction is 0 / 1
   if num == 0:
      return  (0,   1)
   # Compute the greatest common divisor of the numerator and denominator
   g =  gcd(num,      den)
   # Divide both the numerator and denominator by the GCD and return the result
   print  (num    //  g, "/",  den   //  g)
# call above functio
simplify_fraction(3, 6);
```

```
1 / 2


...Program finished with exit code 0
Press ENTER to exit console.
```

Q6> Write a function with the signature def count_evens(L) that returns the number of even integers in the list L. Assume L only contains integers.

Ans>



```python
def count_evens(L):
    count = 0;
    for i in L:
        if (i % 2 == 0):
            count += 1;
    return count;

print(count_evens([1, 2, 3]));
```

```
1


...Program finished with exit code 0
Press ENTER to exit console.
```

Q7> Recall that you can compute the sum 1+2+....+n using total = 0

for i in range(1, n+1):

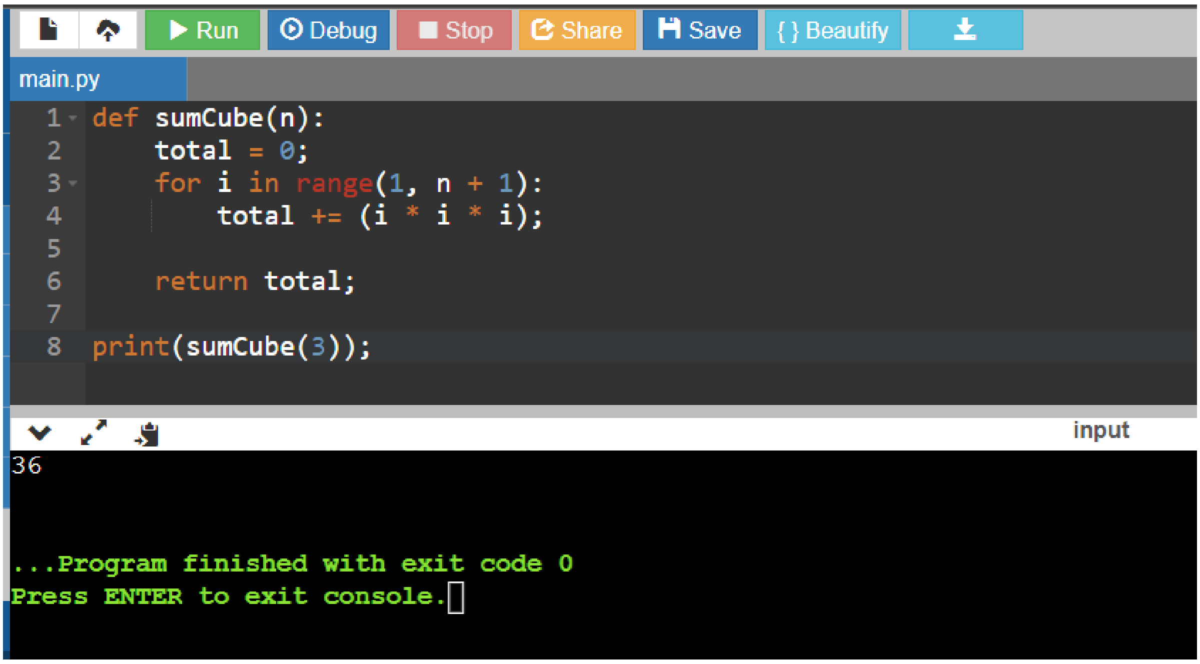
total += i

Write a function that computes 1 3+2 3+...+n 3 using a loop. Write another function that computes the sum using the formula here.

Now write a function with the signature def check_sum(n) that computes the sum of consecutive cubes using a loop and using the formula, and return True iff the results match. (Recall: in this class, iff implies that False is returned if True is not returned.)

Now, write a function with the signature def check_sums_up_to_n(N) that checks that the formula works for every n≤N and returns True iff for every n≤N, the formula works

Ans>



```python
def sumCube(n):
    total = 0;
    for i in range(1, n + 1):
        total += (i * i * i);

    return total;

print(sumCube(3));
```

```
36


...Program finished with exit code 0
Press ENTER to exit console.
```