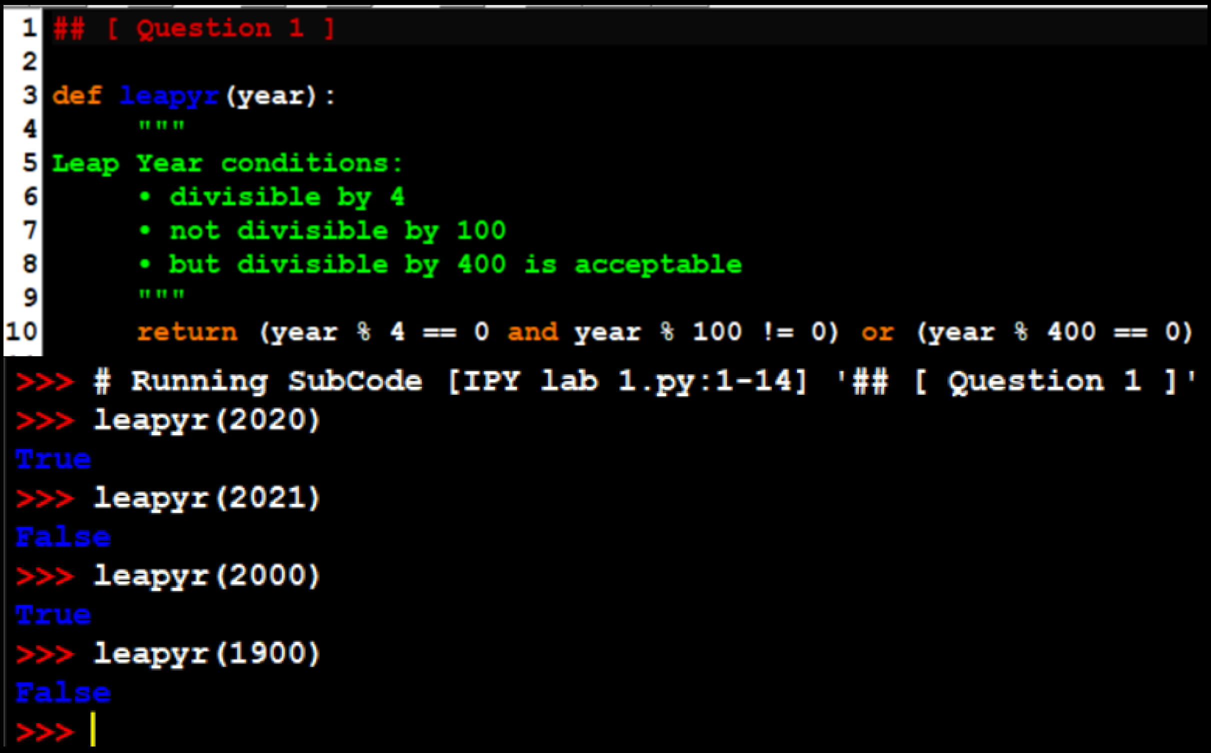
Name: Girish S Roll: AM.EN.U4AIE22044

Introduction to Python - 22AIE205

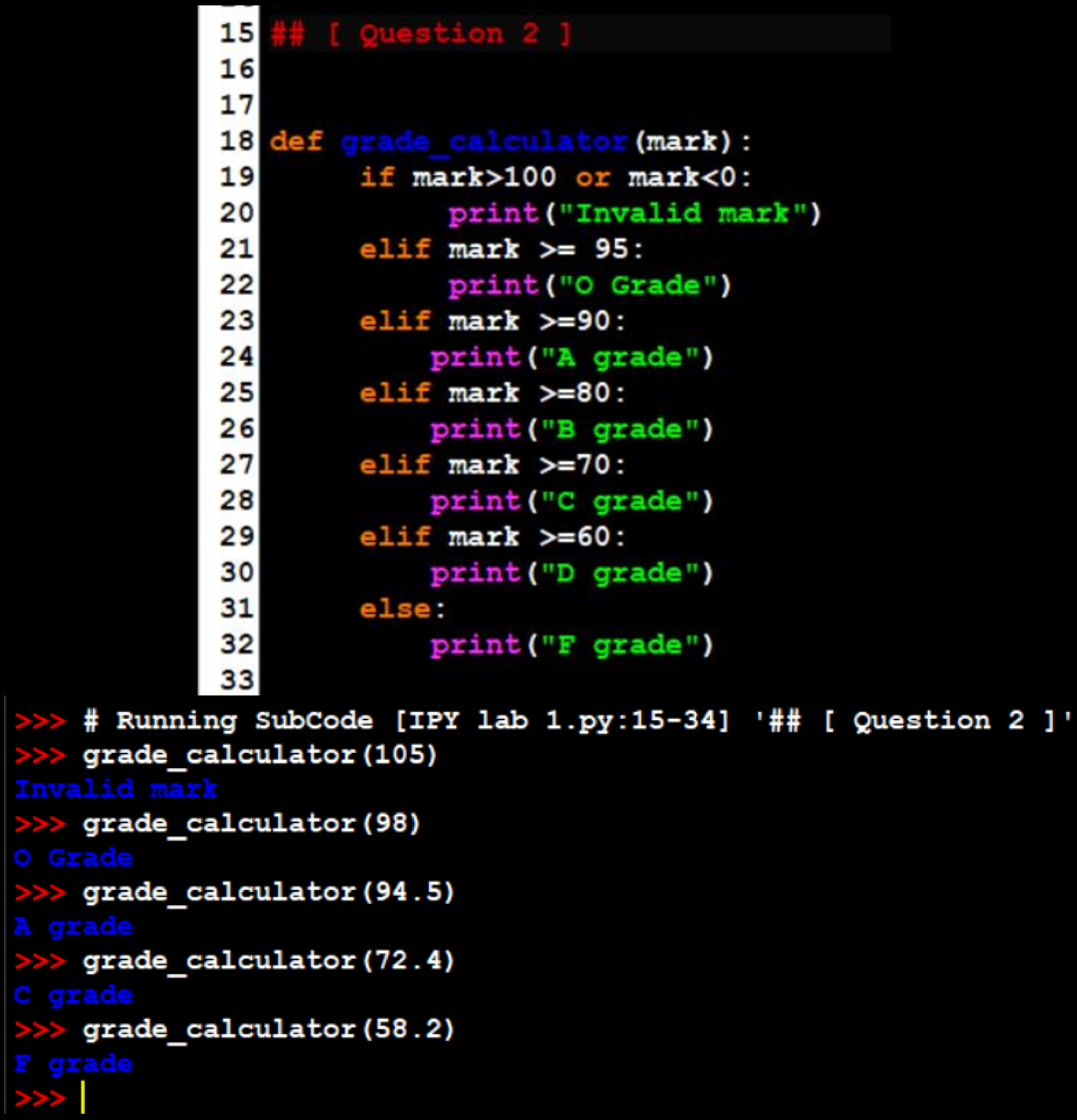
1. You are tasked with creating a program that determines whether a given year is a leap year or not. A leap year is a year that is exactly divisible by 4, except for years that are divisible by 100 but not by 400. Write a Python program that takes a year as input and prints whether it is a leap year or not.



```
## [ Question 1 ]

def leapyr(year):
    """
Leap Year conditions:
    • divisible by 4
    • not divisible by 100
    • but divisible by 400 is acceptable
    """
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)
```

```
>>> # Running SubCode [IPY lab 1.py:1-14] '## [ Question 1 ]'
>>> leapyr(2020)
True
>>> leapyr(2021)
False
>>> leapyr(2000)
True
>>> leapyr(1900)
False
>>> 
```

2. You are responsible for grading the final exam of a computer science class. The grading scale is as follows:
   A: 90-100
   B: 80-89
   C: 70-79
   D: 60-69
   F: Below 60
   Write a Python program that takes a student's exam score as input and determines their grade using an if-else ladder. The program should display the grade earned by the student.



```python
## [ Question 2 ]


def grade_calculator(mark):
    if mark>100 or mark<0:
        print("Invalid mark")
    elif mark >= 95:
        print("O Grade")
    elif mark >=90:
       print("A grade")
    elif mark >=80:
       print("B grade")
    elif mark >=70:
       print("C grade")
    elif mark >=60:
       print("D grade")
    else:
       print("F grade")

```

```
>>> # Running SubCode [IPY lab 1.py:15-34] '## [ Question 2 ]'
>>> grade_calculator(105)
Invalid mark
>>> grade_calculator(98)
O Grade
>>> grade_calculator(94.5)
A grade
>>> grade_calculator(72.4)
C grade
>>> grade_calculator(58.2)
F grade
>>> 
```

3. You are building a program to calculate the cost of shipping a package. The cost depends on the weight of the package and the distance it needs to be shipped. Here are the rules:

• If the package weighs less than or equal to 2 pounds, the base cost is $5.00.

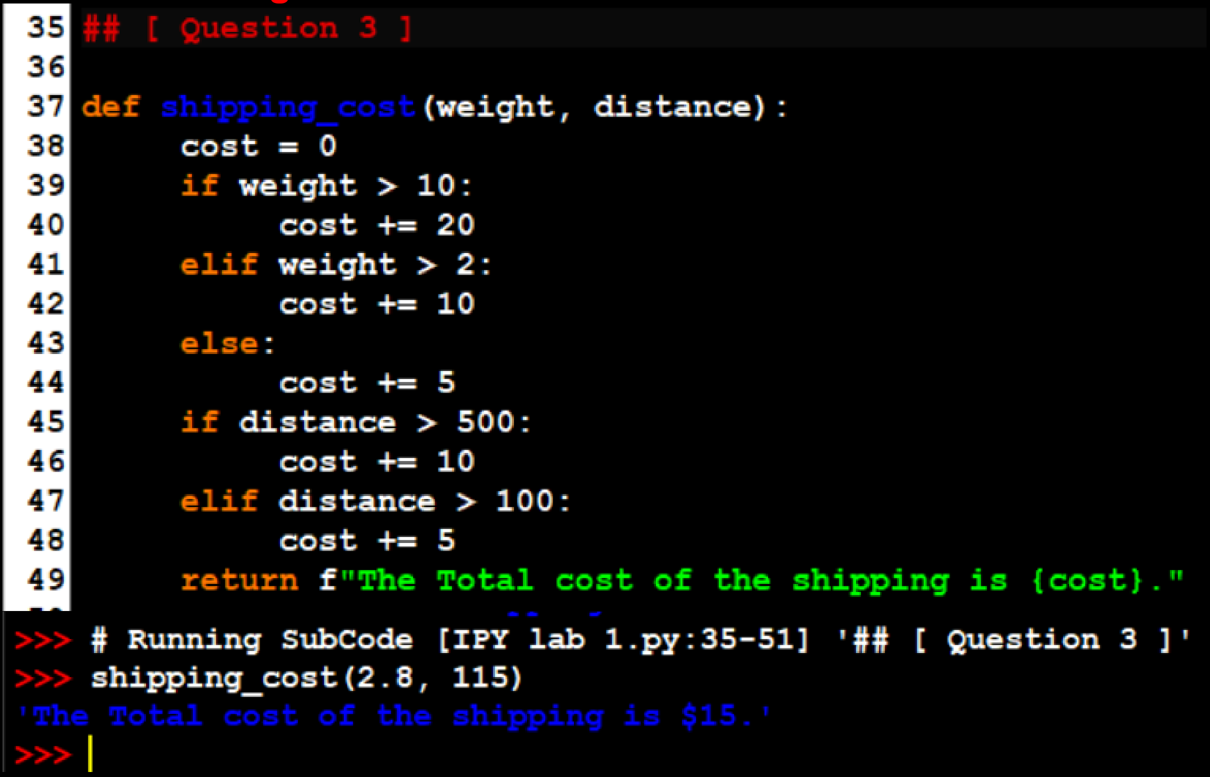
• If the package weighs more than 2 pounds but less than or equal to 10 pounds, the base cost is $10.00.

• If the package weighs more than 10 pounds, the base cost is $20.00.

• If the distance is less than or equal to 100 miles, there's no additional charge.

- If the distance is greater than 100 miles but less than or equal to 500 miles, there's a $5.00 additional charge.

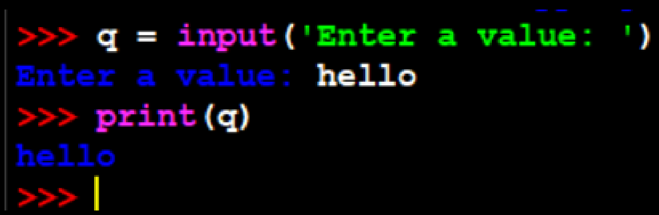
- If the distance is greater than 500 miles, there's a $10.00 additional charge.



```
## [ Question 3 ]

def shipping_cost(weight, distance):
    cost = 0
    if weight > 10:
        cost += 20
    elif weight > 2:
        cost += 10
    else:
        cost += 5
    if distance > 500:
        cost += 10
    elif distance > 100:
        cost += 5
    return f"The Total cost of the shipping is {cost}."
```

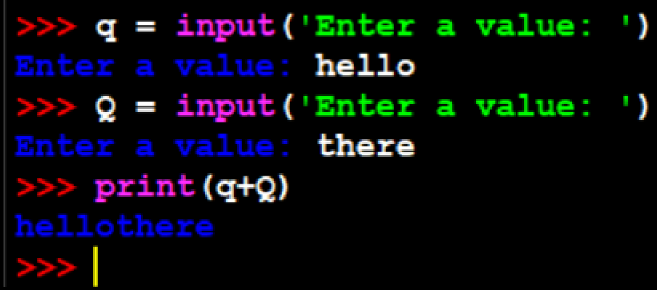
```
>>> # Running SubCode [IPY lab 1.py:35-51] '## [ Question 3 ]'
>>> shipping_cost(2.8, 115)
'The Total cost of the shipping is $15.'
>>> |
```

4. Accepting user input. Write your observations of the output of (a) to (d)
    a.



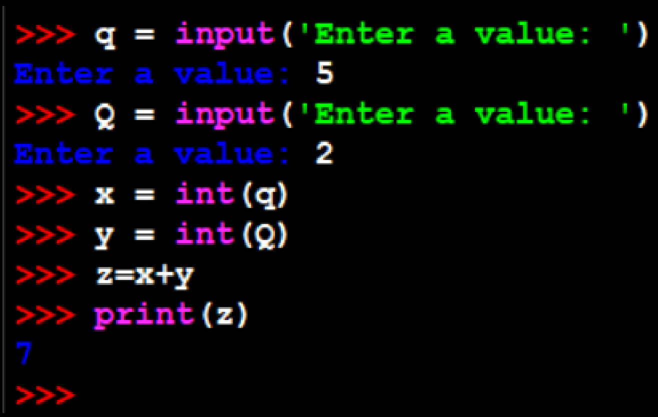
```
>>> q = input('Enter a value: ')
Enter a value: hello
>>> print(q)
hello
>>> |
```

b.



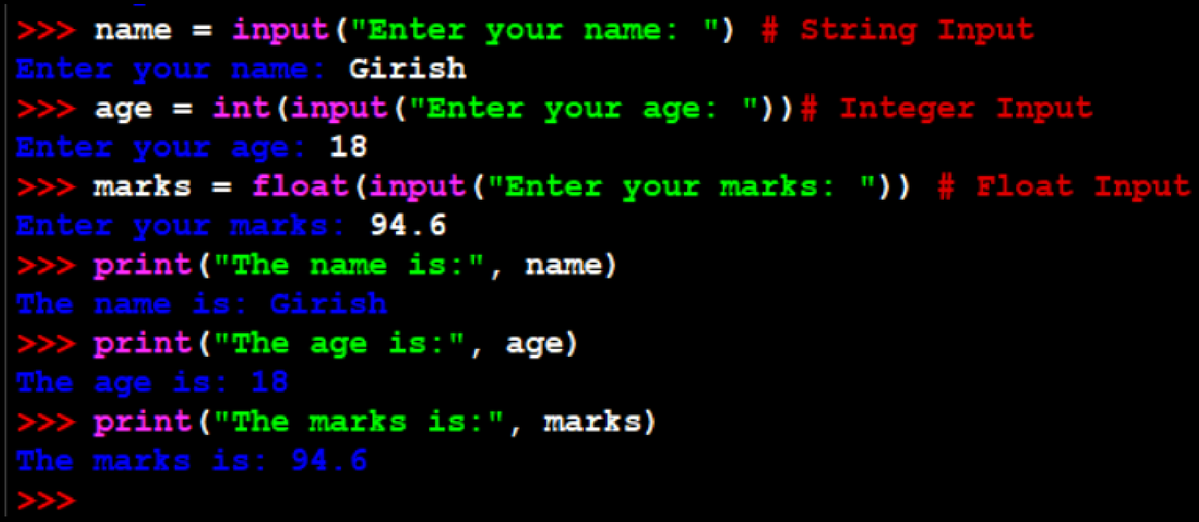
```
>>> q = input('Enter a value: ')
Enter a value: hello
>>> Q = input('Enter a value: ')
Enter a value: there
>>> print(q+Q)
hellothere
>>> |
```

c.



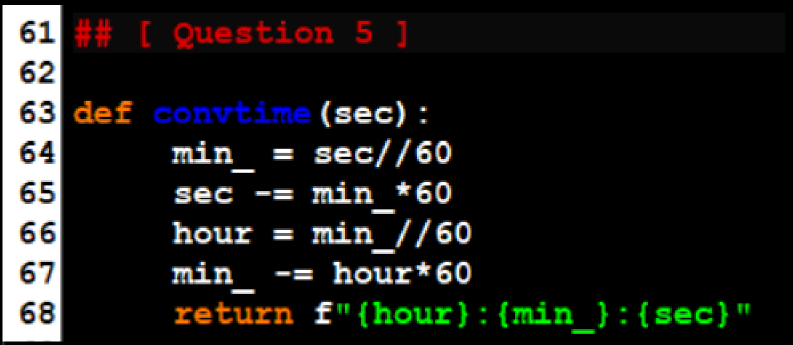
```
>>> q = input('Enter a value: ')
Enter a value: 5
>>> Q = input('Enter a value: ')
Enter a value: 2
>>> x = int(q)
>>> y = int(Q)
>>> z=x+y
>>> print(z)
7
>>>
```

d.

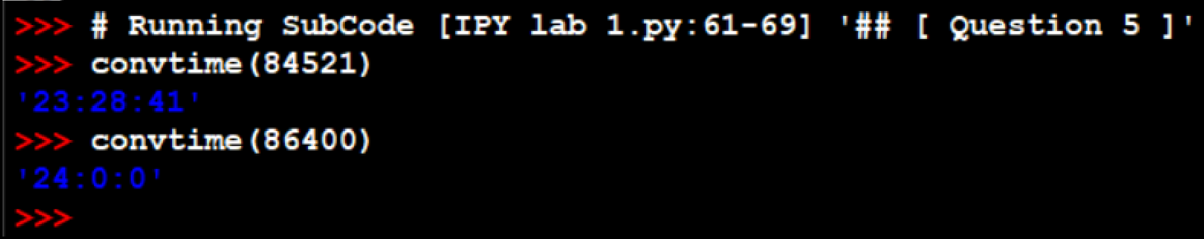


```
>>> name = input("Enter your name: ") # String Input
Enter your name: Girish
>>> age = int(input("Enter your age: "))# Integer Input
Enter your age: 18
>>> marks = float(input("Enter your marks: ")) # Float Input
Enter your marks: 94.6
>>> print("The name is:", name)
The name is: Girish
>>> print("The age is:", age)
The age is: 18
>>> print("The marks is:", marks)
The marks is: 94.6
>>>
```

5. Write a program to read the number of seconds and print it in the form hr:min:sec.



```
## [ Question 5 ]

def convtime(sec):
     min_ = sec//60
     sec -= min_*60
     hour = min_//60
     min_ -= hour*60
     return f"{hour}:{min_}:{sec}"
```

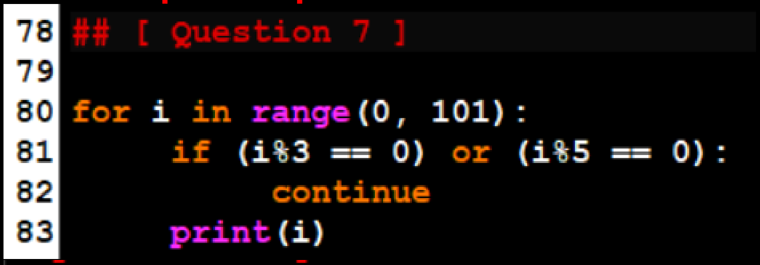


```
>>> # Running SubCode [IPY lab 1.py:61-69] '## [ Question 5 ]'
>>> convtime(84521)
'23:28:41'
>>> convtime(86400)
'24:0:0'
>>>
```

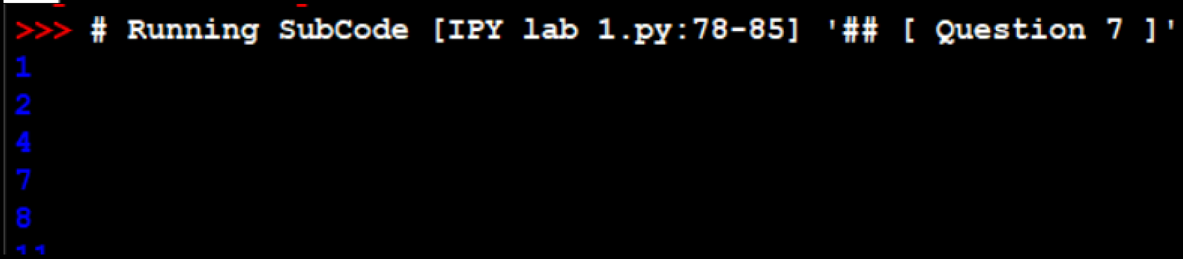
6. Which out of the code snippets below, print the numbers from 1 to 10. Give the reason for the error in the code snippets below which does not print from 1 to 10.

- **All the snippets will not run due to Indentation Error. [After correcting the Indentation Error]**
- **Snippet ‘a’ will run and print until 10 since the while condition is i<10 and not i<=10.**
- **Snippet ‘c’ will run but will print 3 5 7 9, due to the initial condition being i=3 and increment update being i+=2.**
- **Snippet ‘d’ and ‘e’ will have no output on stdout since while loop’s condition is not satisfied on the 0th iteration.**
- **Snippet ‘e’ has no increment updation resulting in a possible non terminating loop**

7. Write a Python program that prints all the numbers from 0 to 100 except multiples of 3 or 5.

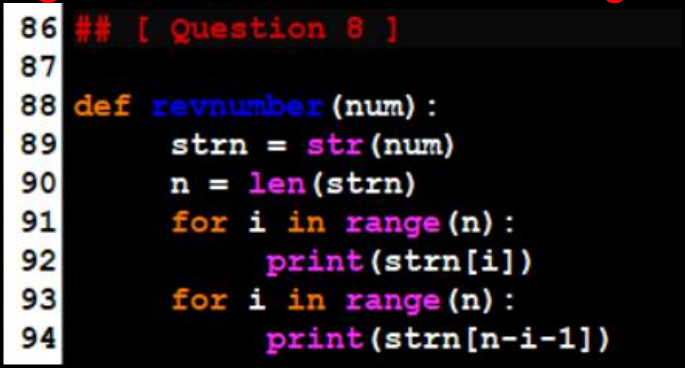


```python
## [ Question 7 ]

for i in range(0, 101):
    if (i%3 == 0) or (i%5 == 0):
        continue
    print(i)
```

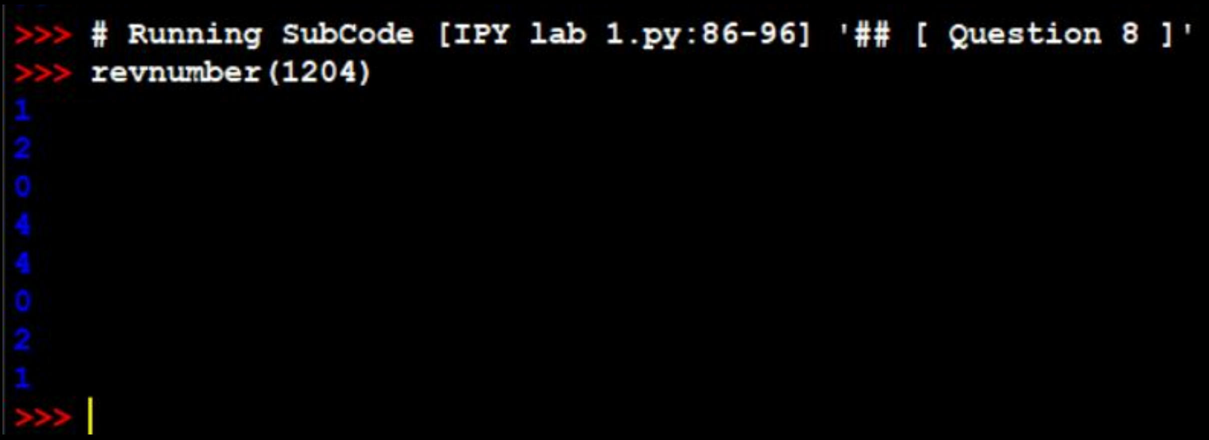


```
>>> # Running SubCode [IPY lab 1.py:78-85] '## [ Question 7 ]'
1
2
4
7
8
```

8. Write a Python program to take an n-digit integer and print the digits of the number from left to right and right to left.

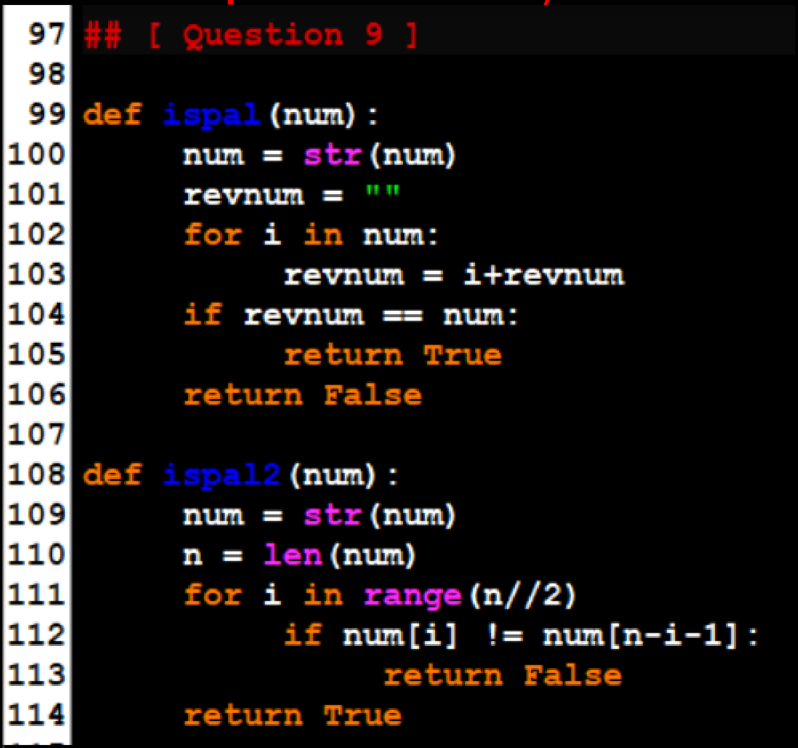


```python
## [ Question 8 ]

def revnumber(num):
    strn = str(num)
    n = len(strn)
    for i in range(n):
        print(strn[i])
    for i in range(n):
        print(strn[n-i-1])
```

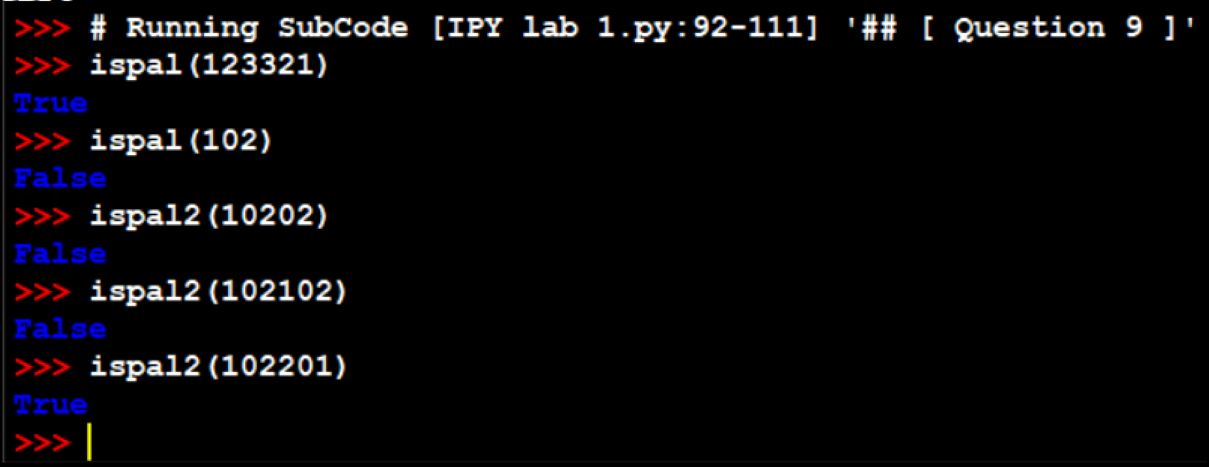


```
>>> # Running SubCode [IPY lab 1.py:86-96] '## [ Question 8 ]'
>>> revnumber(1204)
1
2
0
4
4
0
2
1
>>> 
```

9. Write a python program to check if a number given by the user is a palindrome. (Hint: A number is a palindrome if the number is equal to its reverse.)



```python
## [ Question 9 ]

def ispal(num):
    num = str(num)
    revnum = ""
    for i in num:
        revnum = i+revnum
    if revnum == num:
        return True
    return False

def ispal2(num):
    num = str(num)
    n = len(num)
    for i in range(n//2):
        if num[i] != num[n-i-1]:
            return False
    return True
```



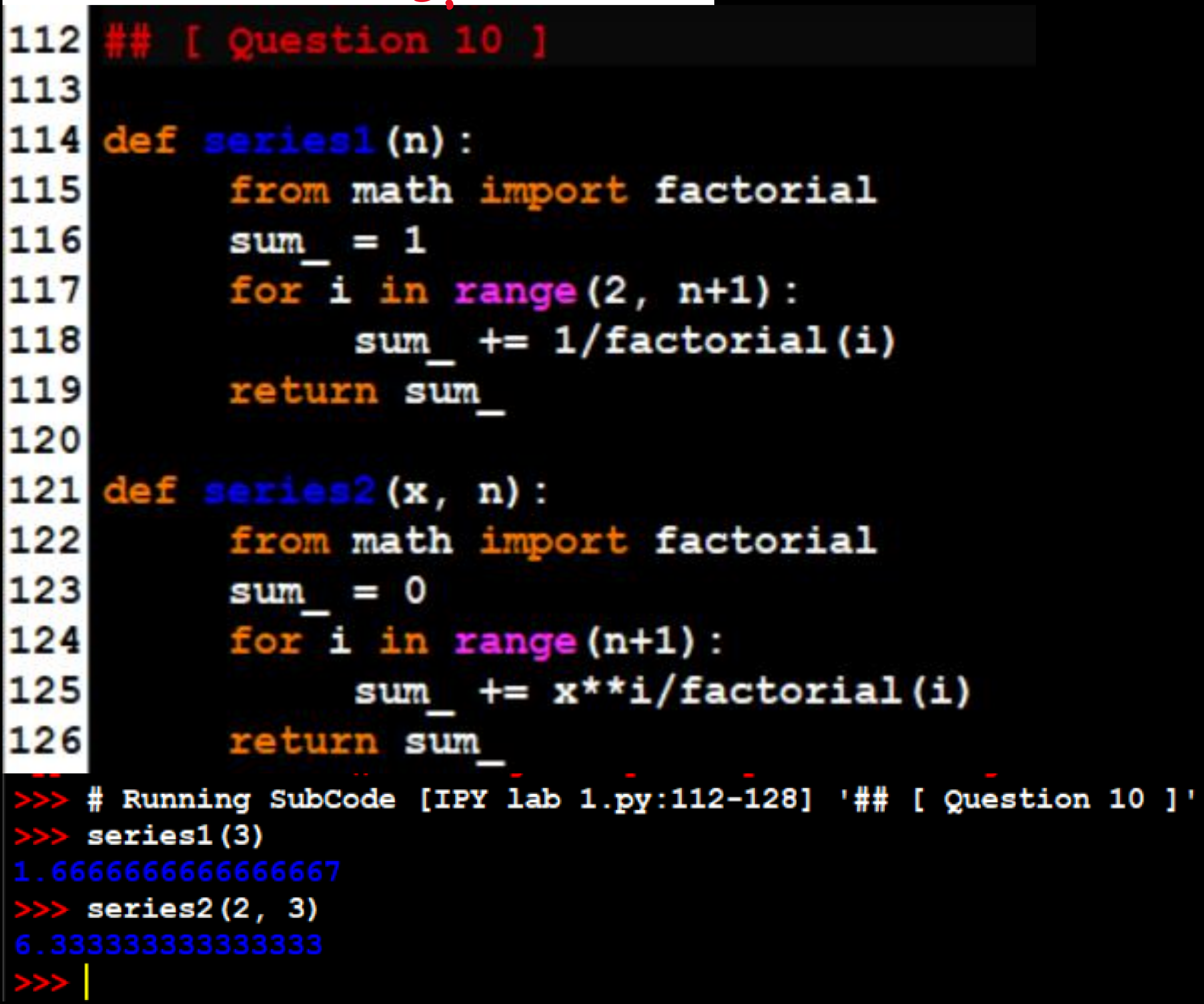
```
>>> # Running SubCode [IPY lab 1.py:92-111] '## [ Question 9 ]'
>>> ispal(123321)
True
>>> ispal(102)
False
>>> ispal2(10202)
False
>>> ispal2(102102)
False
>>> ispal2(102201)
True
>>> 
```

10. Write a Python program to find the sum of the below series provided n is a number given by the user.

$$1 + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{n!};$$

$$x + \frac{x^2}{2!} + \frac{x^3}{2!} + \ldots + \frac{x^n}{n!},$$

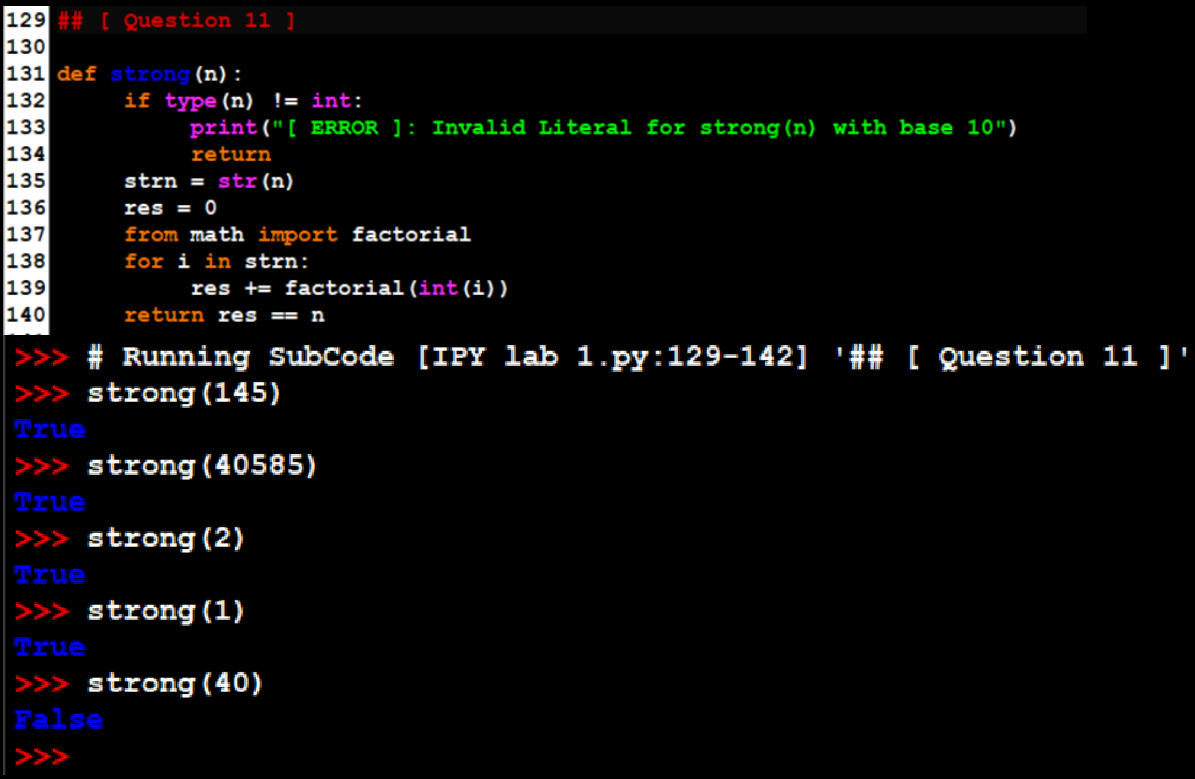
3!



```python
## [ Question 10 ]

def series1(n):
        from math import factorial
        sum_ = 1
        for i in range(2, n+1):
                sum_ += 1/factorial(i)
        return sum_

def series2(x, n):
        from math import factorial
        sum_ = 0
        for i in range(n+1):
                sum_ += x**i/factorial(i)
        return sum_
```

```
>>> # Running SubCode [IPY lab 1.py:112-128] '## [ Question 10 ]'
>>> series1(3)
1.6666666666666667
>>> series2(2, 3)
6.333333333333333
>>> 
```

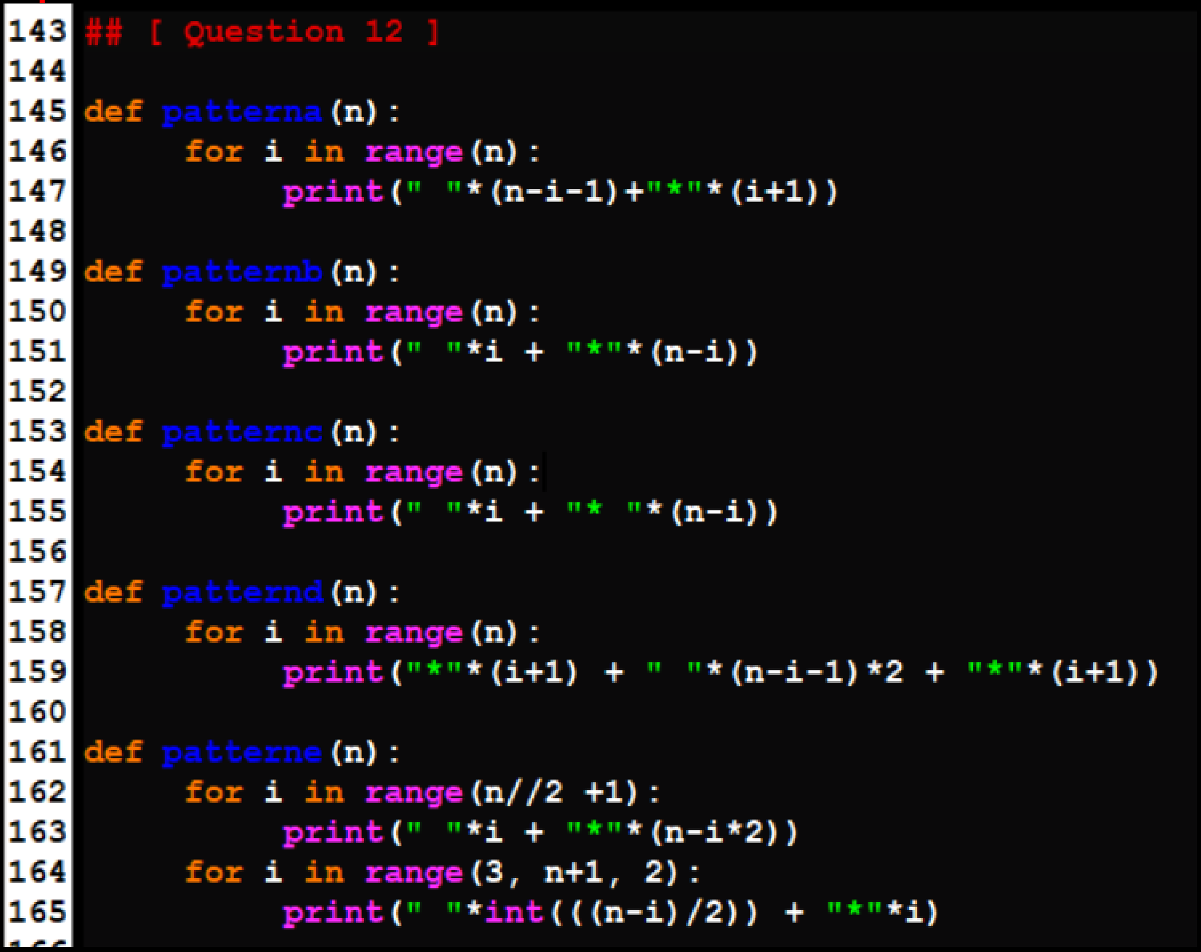
11. Write a program to check whether a number is strong number or not. *Strong number* is a special number whose sum of factorial of digits is equal to the original number.



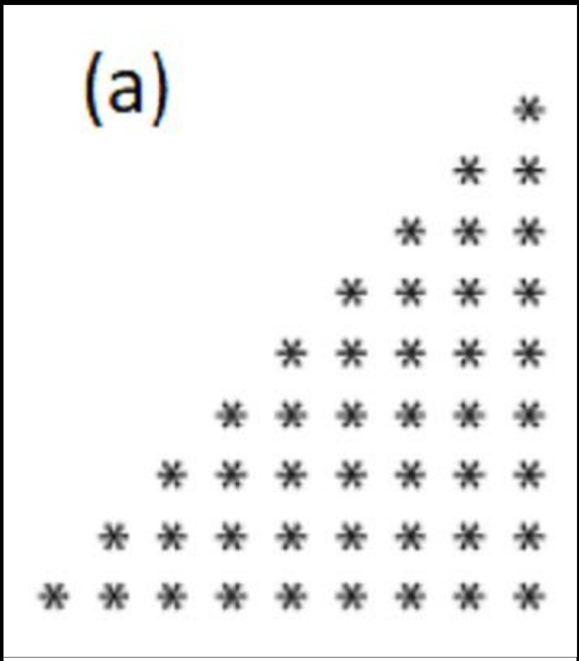
```
## [ Question 11 ]

def strong(n):
    if type(n) != int:
        print("[ ERROR ]: Invalid Literal for strong(n) with base 10")
        return
    strn = str(n)
    res = 0
    from math import factorial
    for i in strn:
        res += factorial(int(i))
    return res == n
```

```
>>> # Running SubCode [IPY lab 1.py:129-142] '## [ Question 11 ]'
>>> strong(145)
True
>>> strong(40585)
True
>>> strong(2)
True
>>> strong(1)
True
>>> strong(40)
False
>>>
```

12. Write python program to print the below patterns. Take as input no. of rows

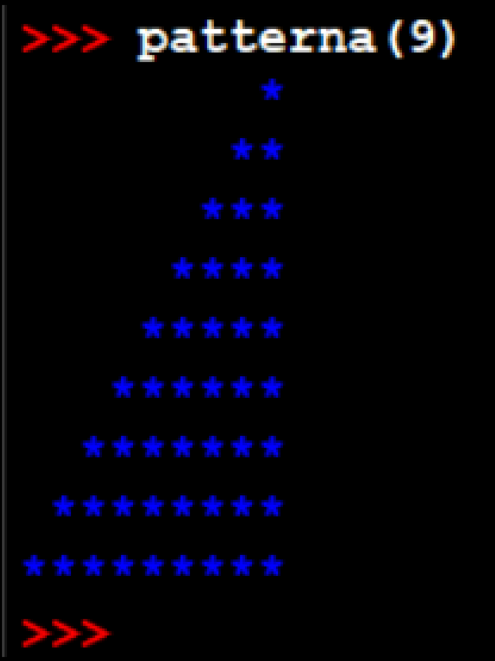


```
## [ Question 12 ]

def patterna(n):
    for i in range(n):
        print(" "*(n-i-1)+"*"*(i+1))

def patternb(n):
    for i in range(n):
        print(" "*i + "*"*(n-i))

def patternc(n):
    for i in range(n):
        print(" "*i + "* "*(n-i))

def patternd(n):
    for i in range(n):
        print("*"*(i+1) + " "*(n-i-1)*2 + "*"*(i+1))

def patterne(n):
    for i in range(n//2 +1):
        print(" "*i + "*"*(n-i*2))
    for i in range(3, n+1, 2):
        print(" "*int(((n-i)/2)) + "*"*i)
```

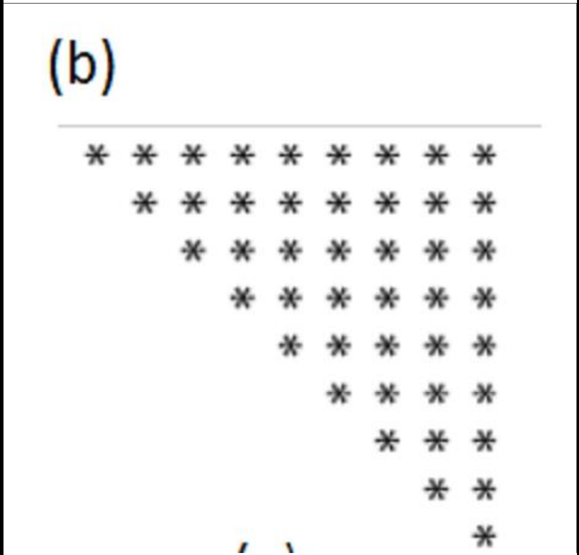


(a)

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

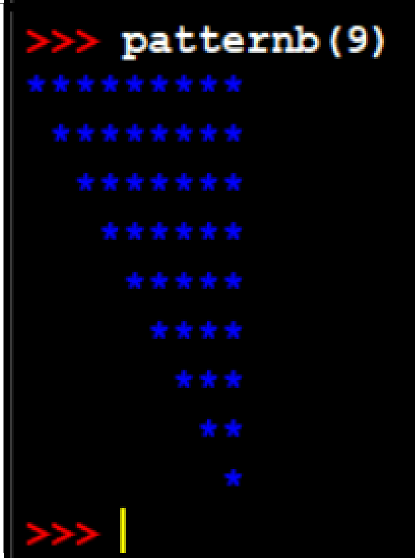


```
>>> patterna(9)
        *
       **
      ***
     ****
    *****
   ******
  *******
 ********
*********
>>>
```

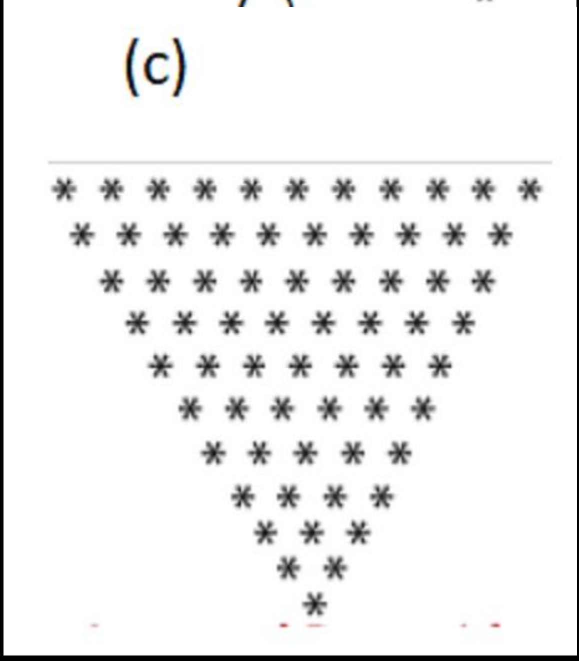


(b)

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

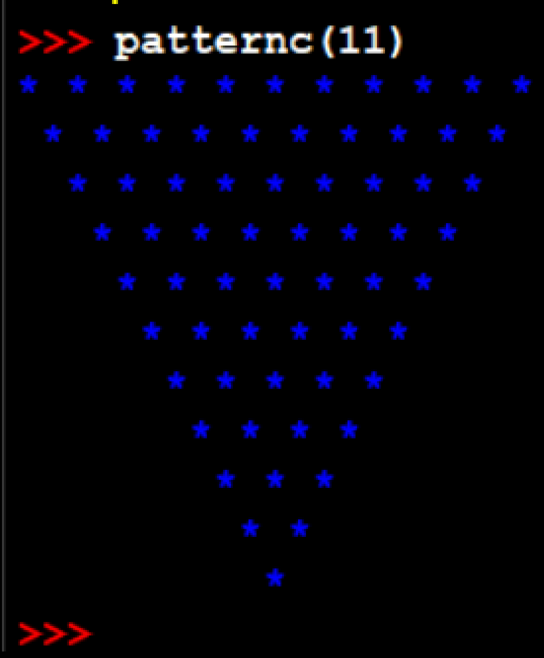


```
>>> patternb(9)
*********
 ********
  *******
   ******
    *****
     ****
      ***
       **
        *
>>> |
```

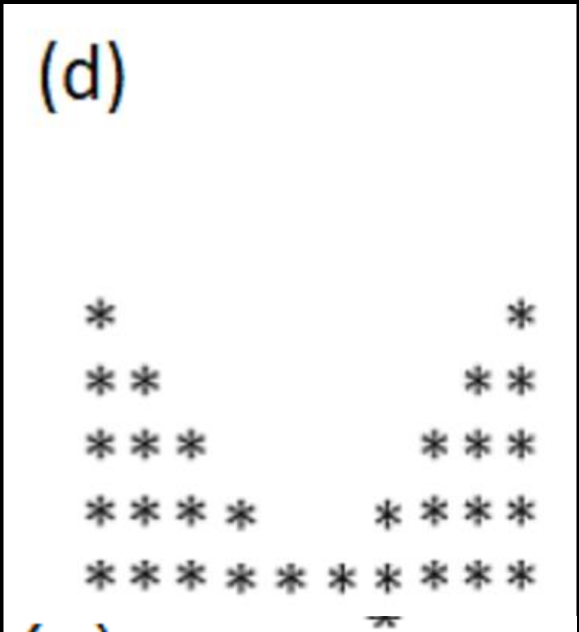


(c)

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

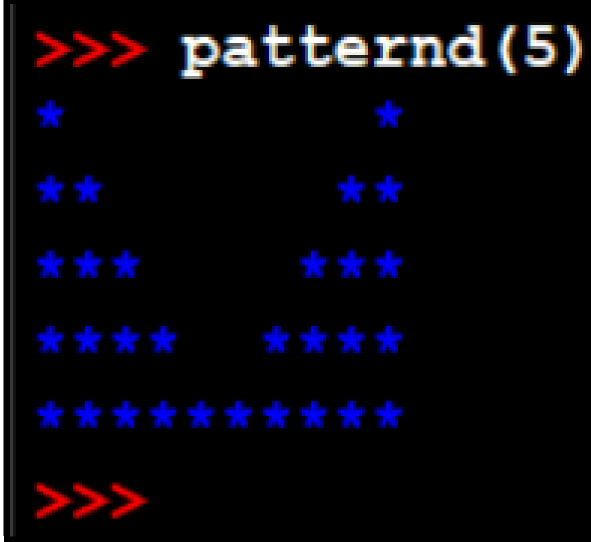


```
>>> patternc(11)
* * * * * * * * * * *
 * * * * * * * * * *
  * * * * * * * * *
   * * * * * * * *
    * * * * * * *
     * * * * * *
      * * * * *
       * * * *
        * * *
         * *
          *
>>>
```

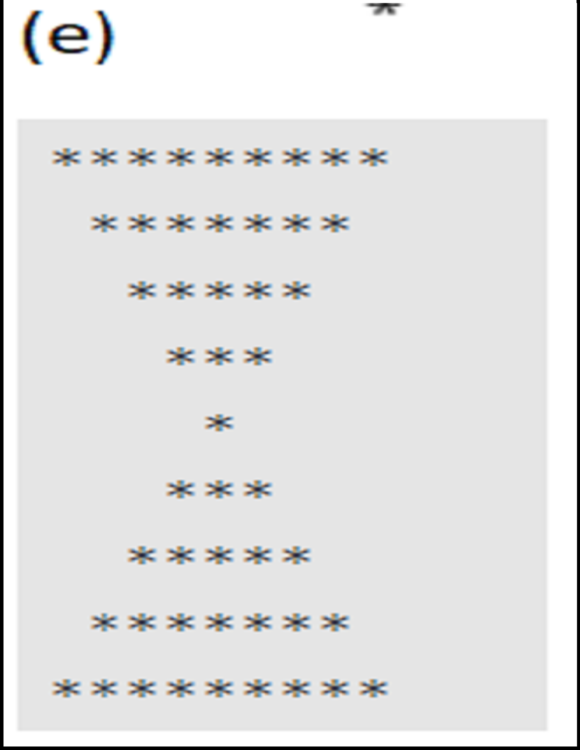


(d)

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

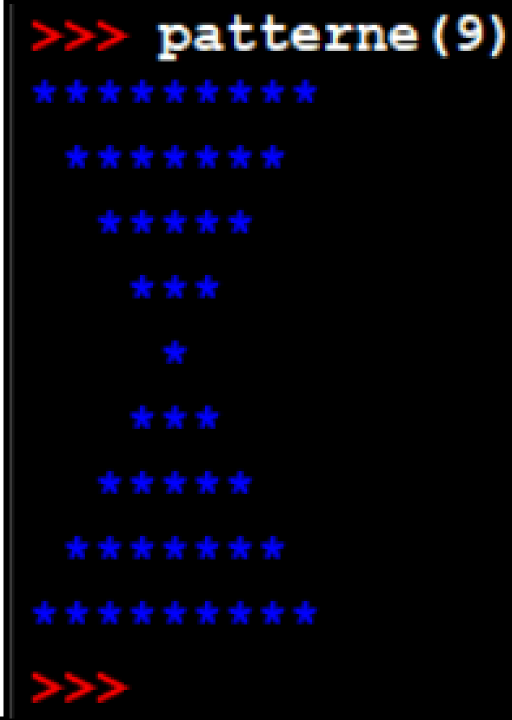


```
>>> patternd(5)
*        *
**      **
***    ***
****  ****
**********
>>>
```



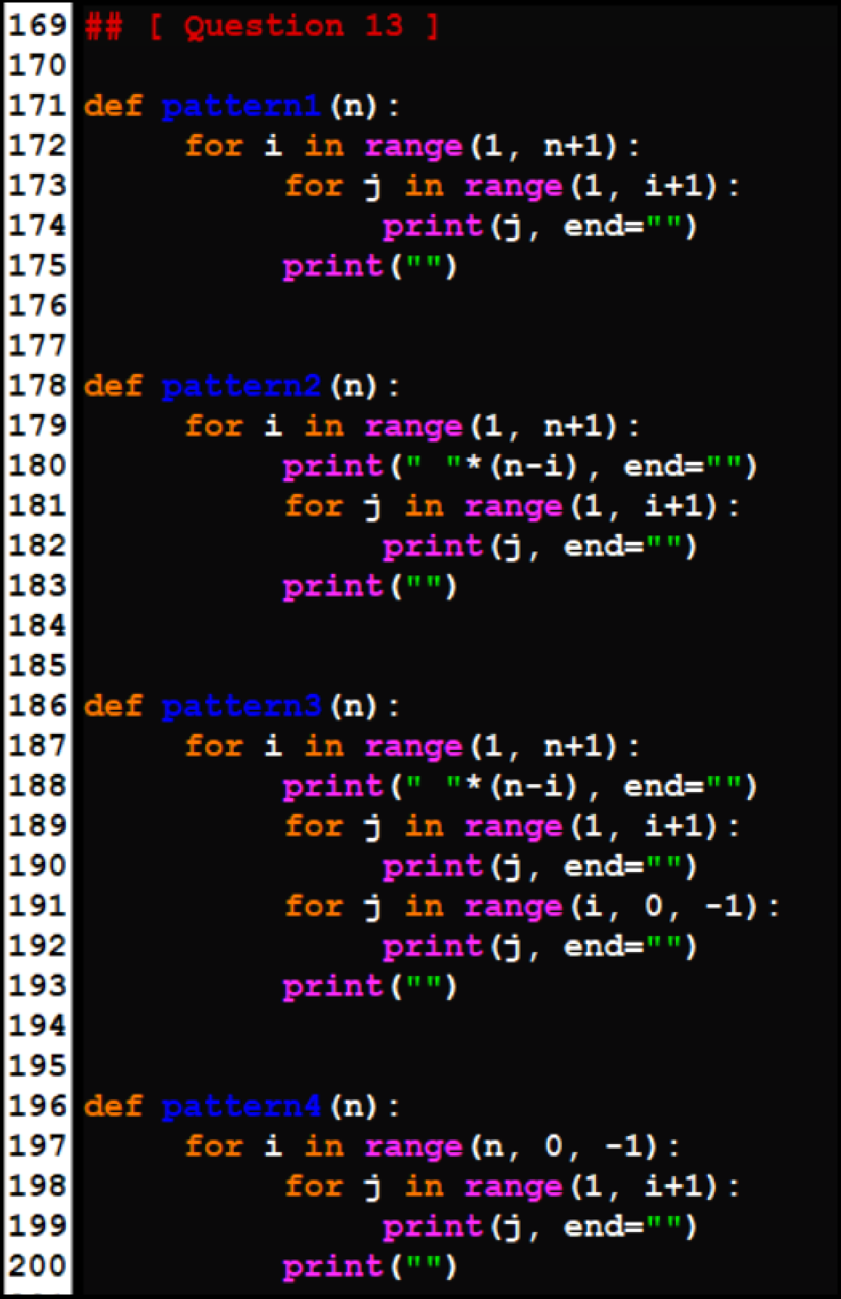
(e)

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

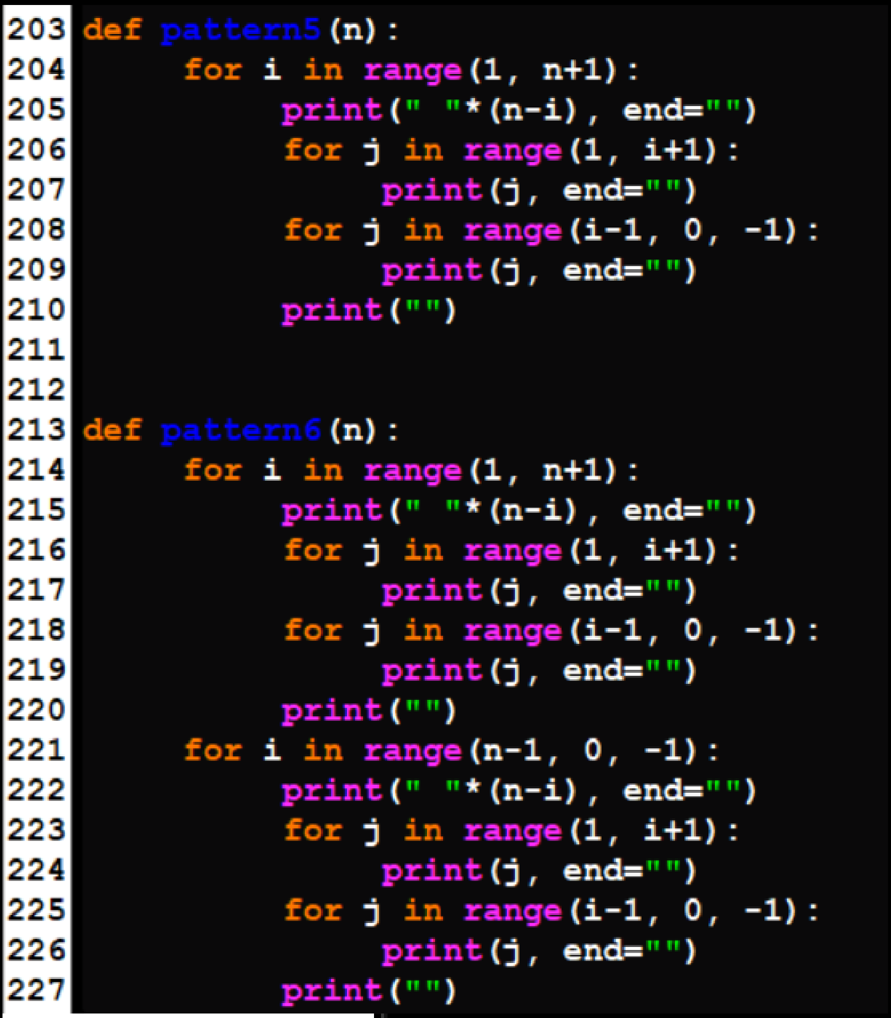


```
>>> patterne(9)
*********
 *******
  *****
   ***
    *
   ***
  *****
 *******
*********
>>>
```

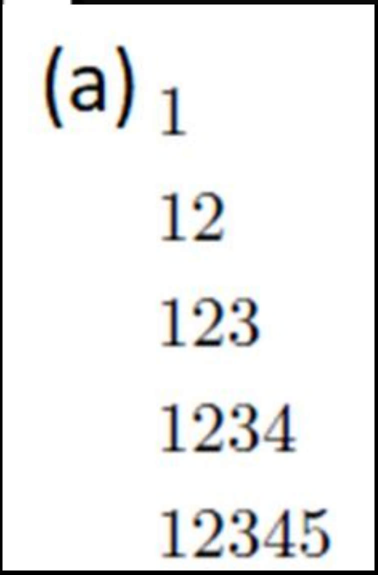
13.Write a Python program to print the below patterns.



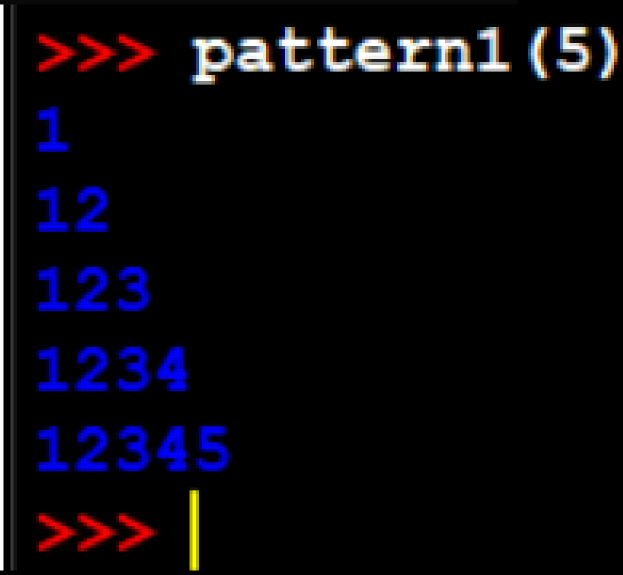
```
## [ Question 13 ]

def pattern1(n):
    for i in range(1, n+1):
        for j in range(1, i+1):
            print(j, end="")
        print("")


def pattern2(n):
    for i in range(1, n+1):
        print(" "*(n-i), end="")
        for j in range(1, i+1):
            print(j, end="")
        print("")


def pattern3(n):
    for i in range(1, n+1):
        print(" "*(n-i), end="")
        for j in range(1, i+1):
            print(j, end="")
        for j in range(i, 0, -1):
            print(j, end="")
        print("")


def pattern4(n):
    for i in range(n, 0, -1):
        for j in range(1, i+1):
            print(j, end="")
        print("")
```



```python
def pattern5(n):
    for i in range(1, n+1):
        print(" "*(n-i), end="")
        for j in range(1, i+1):
            print(j, end="")
        for j in range(i-1, 0, -1):
            print(j, end="")
        print("")


def pattern6(n):
    for i in range(1, n+1):
        print(" "*(n-i), end="")
        for j in range(1, i+1):
            print(j, end="")
        for j in range(i-1, 0, -1):
            print(j, end="")
        print("")
    for i in range(n-1, 0, -1):
        print(" "*(n-i), end="")
        for j in range(1, i+1):
            print(j, end="")
        for j in range(i-1, 0, -1):
            print(j, end="")
        print("")
```

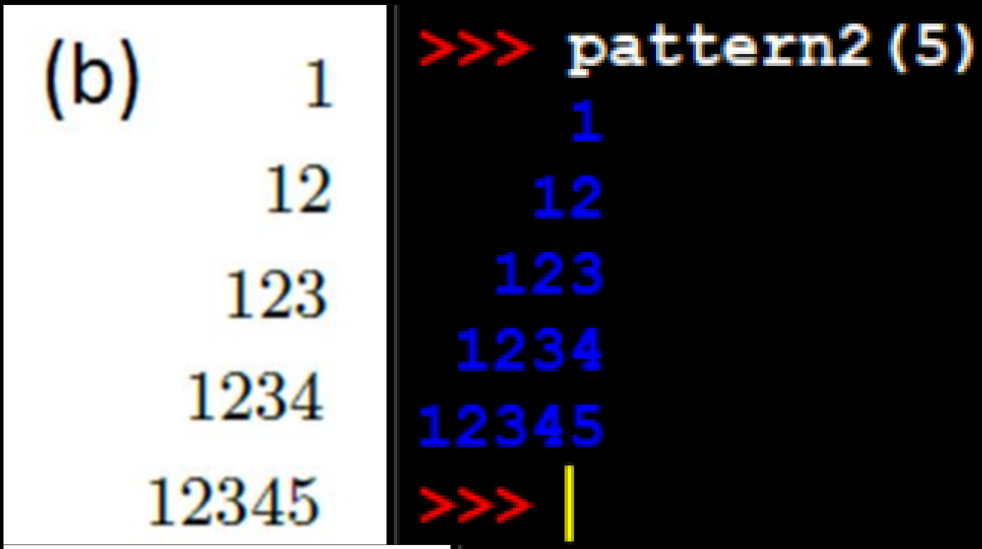


(a)
```
1
12
123
1234
12345
```



```
>>> pattern1(5)
1
12
123
1234
12345
>>> 
```

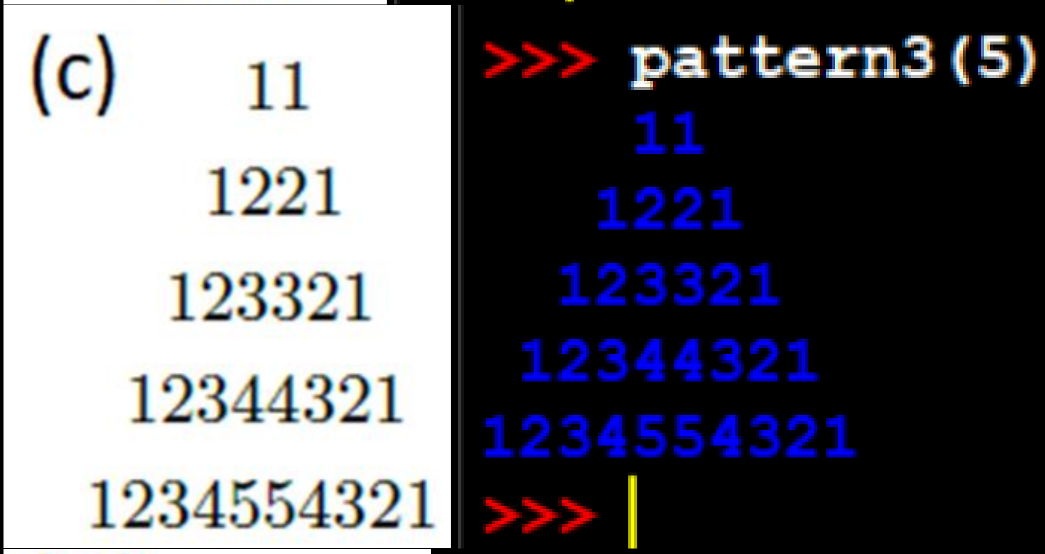
(b)



```
    1
   12
  123
 1234
12345
```

```
>>> pattern2(5)
    1
   12
  123
 1234
12345
>>> 
```

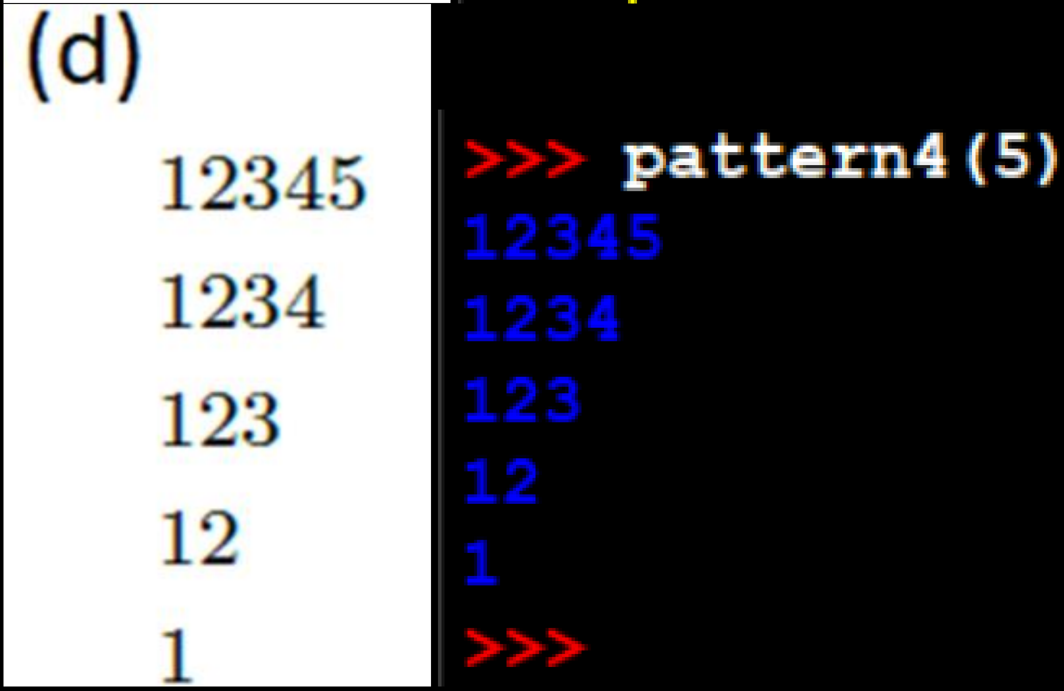
(c)



```
    11
   1221
  123321
 12344321
1234554321
```

```
>>> pattern3(5)
    11
   1221
  123321
 12344321
1234554321
>>> 
```

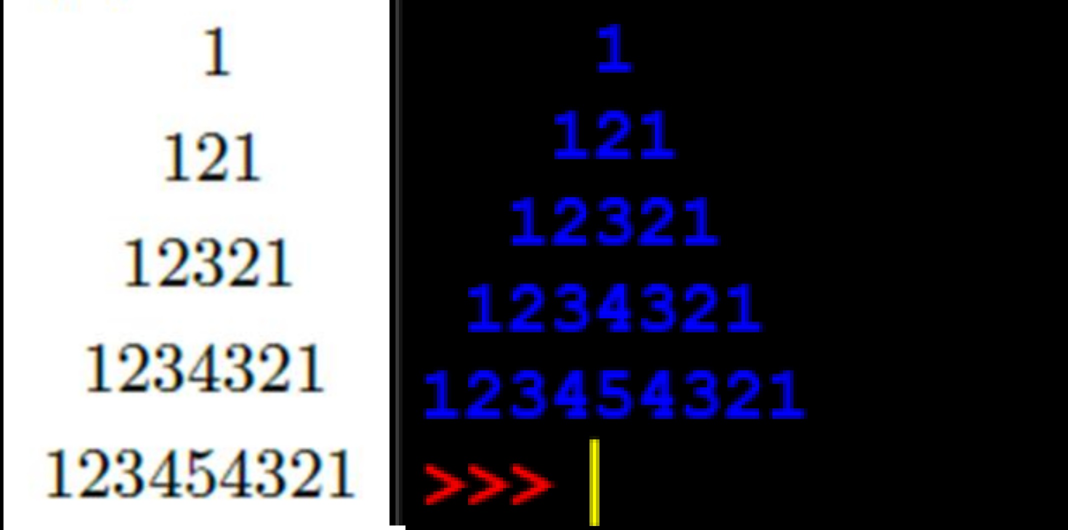
(d)



```
12345
1234
123
12
1
```

```
>>> pattern4(5)
12345
1234
123
12
1
>>> 
```

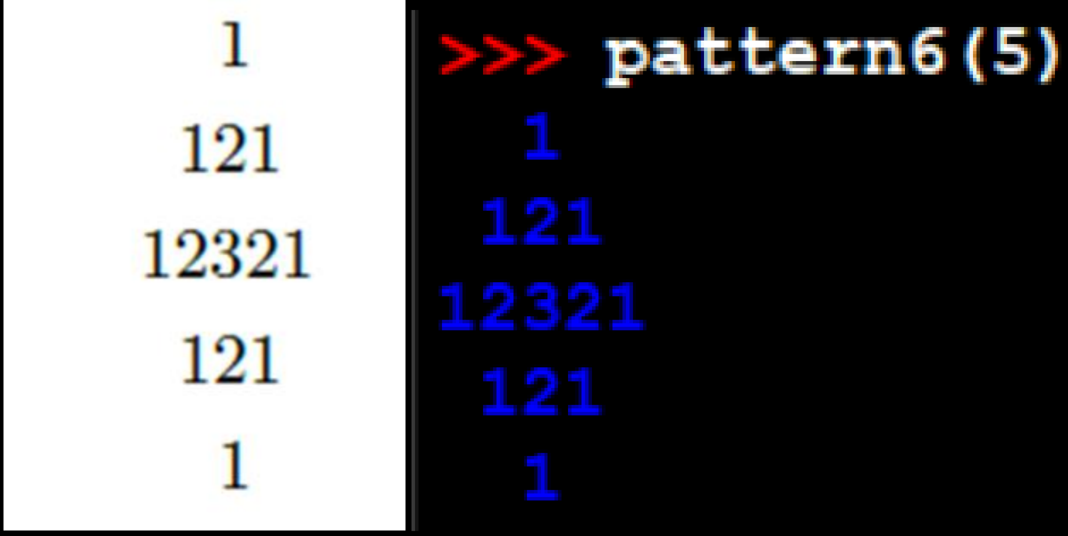
(e)



```
    1
   121
  12321
 1234321
123454321
```

```
>>> pattern5(5)
    1
   121
  12321
 1234321
123454321
>>> 
```

(f)



```
  1
 121
12321
 121
  1
```

```
>>> pattern6(5)
  1
 121
12321
 121
  1
```