**for** i **in** range(1,6):*#outer loop for row*

**for** j **in** range(1,6):*# inner for loop for column*

**if** i<=j:

print("\*",end="")

**else**:

print(" ",end="")

print()

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

In [17]:

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** i<=j:

print("\*",end=" ")*#if we add single space in end=" "*

**else**:

print(" ",end="")

print()

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

In [18]:

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** i<=j:

print("\*",end="")

**else**:

print("",end="") *# if we remove one single space from print("")*

print()

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

In [20]:

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** i<=j:

print("\*",end="")

**else**:

print("",end=" ") *# if we add one space in end=" "*

print()

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

In [23]:

*# if we do some littel littel changes with space then also we get op different different*

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** i<=j:

print("\*",end=" ")

**else**:

print(" ",end=" ")

print()

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

In [30]:

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** j==1 **or** i==5:

print("\*",end=" ")

print()

\*

\*

\*

\*

\* \* \* \* \*

In [39]:

*#print right diagonal*

**for** i **in** range(1,6):

**for** j **in** range(1,6):

**if** j==6-i :

print("\*",end="")

**else**:

print(" ",end="")

print()

\*

\*

\*

\*

\*

In [46]:

col=10

**for** i **in** range(1,11): *# i==j for left diagonal j==(col+1)-i for right diagonal all other for 4 sides*

**for** j **in** range(1,11):

**if** j==1 **or** i==10 **or** j==(col+1)-i **or** j==10 **or** i==1 **or** i==j:

print("\*",end="")

**else**:

print(" ",end="")

print()

\*\*\*\*\*\*\*\*\*\*

\*\* \*\*

\* \* \* \*

\* \* \* \*

\* \*\* \*

\* \*\* \*

\* \* \* \*

\* \* \* \*

\*\* \*\*

\*\*\*\*\*\*\*\*\*\*

In [65]:

**for** i **in** range(1,6):

flag=**True** *# flag to skip alternate one star from full piryamid*

**for** j **in** range(1,10):

**if** j>=6-i **and** j<=4+i **and** flag:

print("\*",end=" ")

flag=**False**

**else**:

print(" " , end=" ")

flag=**True**

print();

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

In [66]:

*#logik to print full piryamid*

**for** i **in** range(1,6):

**for** j **in** range(1,10):

**if** j>=6-i **and** j<=4+i :

print("\*",end=" ")

**else**:

print(" " , end=" ")

print();

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

In [68]:

**for** i **in** range (1,6):

**for** j **in** range (1,10):

*#if j==6-i or j==4+i or i==5 :*

**if** j==6-i **or** j==4+i **or** i==5 **and** j%**2**==0: # base contain only even number position star that's why we write here logic

*# j%2==0 to get correct base*

print("\*",end=" ")

**else**:

print(" ",end=" ")

print()

\*

\* \*

\* \*

\* \*

\* \* \* \* \* \*

In [69]:

**for** i **in** range (1,6):

count=i

**for** j **in** range (1,10):

**if** j>=6-i **and** j<=4+i:

print(count,end=" ")

**if** j<5:

count+=1

**else**:

count-=1

**else**:

print(" ",end=" ")

print()

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

In [70]:

**for** i **in** range (1,6):

count=1 *# just change here to get passcal triangle*

**for** j **in** range (1,10):

**if** j>=6-i **and** j<=4+i:

print(count,end=" ")

**if** j<5:

count+=1

**else**:

count-=1

**else**:

print(" ",end=" ")

print()

1

1 2 1

1 2 3 2 1

1 2 3 4 3 2 1

1 2 3 4 5 4 3 2 1

In [71]:

row=6

**for** num **in** range(row):

**for** i **in** range(num):

print(num,end="")

print()

1

22

333

4444

55555

In [72]:

row=5

b=0

**for** i **in** range (row,0,-1):

b+=1

**for** j **in** range(1,i+1):

print(b,end="")

print()

11111

2222

333

44

5

In [76]:

rows =5

**for** ro **in** range(1,rows+1):

**for** col **in** range(1,ro+1):

print(col,end="")

print("")

1

12

123

1234

12345

In [77]:

rows=5

**for** i **in** range(rows,0,-1):

num=i

**for** j **in** range(0,i):

print(num,end="")

print()

55555

4444

333

22

1

In [78]:

rows=5

n=rows *# we write like this here*

**for** i **in** range(rows,0,-1):

*#n=5 this also work here but to improve code efficiency*

**for** j **in** range(1,i+1):

print(n,end="")

print()

55555

5555

555

55

5

In [79]:

rows=6

**for** row **in** range(1,rows):

**for** col **in** range(row,0,-1):

print(col,end="")

print()

1

21

321

4321

54321

In [80]:

rows=5

**for** i **in** range(rows,0,-1):

**for** j **in** range(0,i+1):

print(j,end="")

print()

012345

01234

0123

012

01

In [81]:

*#pyramid of natural no less than 10*

curNo=1

stop=2

row=3

**for** i **in** range(row):

**for** col **in** range(1,stop):

print(curNo,end="")

curNo+=1

print("")

stop+=2

1

234

56789

In [82]:

start=1

stop=2

curNo=stop

**for** row **in** range(2,6):

**for** col **in** range(start , stop):

curNo-=1

print(curNo,end="")

print("")

start=stop

stop+=row

curNo=stop

1

32

654

10987

In [83]:

row=6

**for** i **in** range(1,row+1):

**for** j **in** range(1,i-1):

print(j,end="")

**for** j **in** range(i-1,0,-1):

print(j,end="")

print()

1

121

12321

1234321

123454321

In [85]:

row=6

**for** i **in** range(0,row):

**for** j **in** range(row-1,i,-1):

print(j,"",end="")

**for** k **in** range(i):

print('',end="")

**for** l **in** range(i+1,row):

print(l,"",end="")

print()

5 4 3 2 1 1 2 3 4 5

5 4 3 2 2 3 4 5

5 4 3 3 4 5

5 4 4 5

5 5

In [88]:

row=5

lastevenNo=2\*row

evenNo=lastevenNo

**for** i **in** range(1,row+1):

evenNo=lastevenNo

**for** j **in** range(i):

print(evenNo,end=" ")

evenNo-=2

print("")

10

10 8

10 8 6

10 8 6 4

10 8 6 4 2

In [89]:

row=7

**for** i **in** range(0,row):

**for** j **in** range(0,i+1):

print(i\*j,end="")

print()

0

01

024

0369

0481216

0510152025

061218243036

In [90]:

row =5

i=1

**while** i<=row:

j=1

**while** j<=i:

print((i\*2-1),end='')

j=j+1

i=i+1

print()

1

33

555

7777

99999

In [93]:

rows=6

**for** row **in** range(1,rows):

num=1

**for** j **in** range(rows,0,-1):

**if** j>row:

print("",end="")

**else**:

print(num,end="")

num+=1

print()

1

12

123

1234

12345

In [95]:

size=7

m=(2\*size)-2

**for** i **in** range(0,size):

**for** j **in** range(0,m):

print(end=" ")

m=m-1 *# dec m after each loop*

**for** j **in** range(0,i+1):

print("\*",end=" ")

print()

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \* \*

In [96]:

size=7

m=2\*size-2

**for** i **in** range(size,-1,-1):

**for** j **in** range(m,0,-1):

print(end=" ")

m=m+1 *# dec m after each loop*

**for** j **in** range(0,i+1):

print("\*",end=" ")

print()

\* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

In [98]:

row =5

**for** i **in** range(0,row):

**for** j **in** range(0,i+1):

print("\*",end=" ")

print()

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

In [14]:

*# Python 3.x code to demonstrate star pattern*

*# Function to demonstrate printing pattern of alphabets*

**def** alphabat(n):

*# initializing value corresponding to 'A'*

*# ASCII value*

num = 65

*# outer loop to handle number of rows*

*# 5 in this case*

**for** i **in** range(0, n):

*# inner loop to handle number of columns*

*# values changing acc. to outer loop*

**for** j **in** range(0, i+1):

*# explicitely converting to char*

ch = chr(num)

*# printing char value*

print(ch, end=" ")

*# incrementing number*

num = num + 1

*# ending line after each row*

print("**\r**")

*# Driver Code*

n = 5

alphabat(n)

A

B B

C C C

D D D D

E E E E E

In [20]:

*# Python code 3.x to demonstrate star pattern*

*# Function to demonstrate printing pattern of alphabets*

**def** contalpha(n):

*# initializing value corresponding to 'A'*

*# ASCII value*

num = 65

*# outer loop to handle number of rows*

**for** i **in** range(0, n):

*# inner loop to handle number of columns*

*# values changing acc. to outer loop*

**for** j **in** range(0, i+1):

*# explicitely converting to char*

ch = chr(num)

*# printing char value*

print(ch, end=" ")

*# incrementing at each column*

num = num +1 *# just change is over here just take this increment inside inner for loop*

*# ending line after each row*

print("**\r**")

*# Driver code*

n = 5

contalpha(n)

A

B C

D E F

G H I J

K L M N O

In [21]:

*# Python 3.x code to demonstrate star pattern*

*# Function to demonstrate printing pattern*

**def** pypart2(n):

*# number of spaces*

k = 2\*n - 2

*# outer loop to handle number of rows*

**for** i **in** range(0, n):

*# inner loop to handle number spaces*

*# values changing acc. to requirement*

**for** j **in** range(0, k):

print(end=" ")

*# decrementing k after each loop*

k = k - 2

*# inner loop to handle number of columns*

*# values changing acc. to outer loop*

**for** j **in** range(0, i+1):

*# printing stars*

print("\* ", end="")

*# ending line after each row*

print("**\r**")

*# Driver Code*

n = 5

pypart2(n)

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

In [25]:

row = int(input("Enter number of rows (even): "))

n = row//2

print("floor division n is",n)

print("Generated butterfly pattern is:**\n**")

*# Upper part*

**for** i **in** range(1,n+1): *#outer for loop for row*

**for** j **in** range(1, 2\*n+1): *#inner for loop for column*

**if** j>i **and** j< 2\*n+1-i:

print(" ", end="")

**else**:

print("\* ", end="")

print()

*# Lower part*

**for** i **in** range(n,0,-1):

**for** j **in** range(2\*n,0,-1):

**if** j>i **and** j< 2\*n+1-i:

print(" ", end="")

**else**:

print("\* ", end="")

print()

Enter number of rows (even): 16

floor division n is 8

Generated butterfly pattern is:

\* \*

\* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \*

\* \*

In [29]:

row = int(input('Enter number of rows required: '))

**for** i **in** range(row):

**for** j **in** range(i+1):

**if** j==0 **or** j==i **or** i==row-1:

print('\*',end=" ")

**else**:

print(" ", end=" ")

print()

Enter number of rows required: 7

\*

\* \*

\* \*

\* \*

\* \*

\* \*

\* \* \* \* \* \* \*

In [31]:

row = int(input('Enter number of rows required: '))

**for** i **in** range(row):

**for** j **in** range(row-i):

print(' ', end=' ') *# printing space required and staying in same line*

**for** j **in** range(2\*i+1):

**if** j==0 **or** j==2\*i **or** i==row-1:

print('\*',end=' ')

**else**:

print(' ', end=' ')

print() *# printing new line*

Enter number of rows required: 6

\*

\* \*

\* \*

\* \*

\* \*

\* \* \* \* \* \* \* \* \* \* \*

In [33]:

*# Generating Inverse Pyramid Pattern Using Stars*

row = int(input('Enter number of rows required: '))

**for** i **in** range(row,0,-1):

**for** j **in** range(row-i):

print(' ', end=' ') *# printing space and staying in same line*

**for** j **in** range(2\*i-1):

print('\*',end=' ') *# printing \* and staying in same line*

print() *# printing new line*

Enter number of rows required: 8

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

In [34]:

*# Reading number of rows*

row = int(input('Enter how many lines? '))

*# Generating pattern*

**for** i **in** range(1,row+1):

*# for space*

**for** j **in** range(1, row+1-i):

print(' ', end='')

*# for decreasing pattern*

**for** j **in** range(i,0,-1):

print(j, end='')

*# for increasing pattern*

**for** j **in** range(2,i+1):

print(j, end='')

*# Moving to next line*

print()

Enter how many lines? 7

1

212

32123

4321234

543212345

65432123456

7654321234567

In [36]:

*# Python Program to Generate Flag of Nepal*

*# Generating Triangle Shape*

**def** triangleShape(n):

**for** i **in** range(n):

**for** k **in** range(i+1):

print('\*',end=' ')

print()

*# Generating Pole Shape*

**def** poleShape(n):

**for** i **in** range(n):

print('\*')

*# Input and Function Call*

row = int(input('Enter number of rows: '))

triangleShape(row)

triangleShape(row)

poleShape(row)

Enter number of rows: 9

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\*

\*

\*

\*

\*

\*

\*

\*

\*

In [37]:

*#https://www.codesansar.com/python-programming-examples/generate-christmas-tree-pattern.htm*

*# use this link for more such a programs*

*#closely done with all only few of them are remaining*

In [38]:

n = int(input("Enter number of rows: "))

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

a = 65

**for** j **in** range(1, i+1):

print("**%c**" %(a), end="")

a +=1

print()

Enter number of rows: 7

A

AB

ABC

ABCD

ABCDE

ABCDEF

ABCDEFG

In [39]:

*# Reading number of rows*

row = int(input('Enter how many lines? '))

a = 64

*# Generating pattern*

**for** i **in** range(1,row+1):

*# for space*

**for** j **in** range(1, row+1-i):

print(' ', end='')

*# for decreasing pattern*

**for** j **in** range(i,0,-1):

print('**%c**' % (a+j), end='')

*# for increasing pattern*

**for** j **in** range(2,i+1):

print('**%c**' % (a+j), end='')

*# Moving to next line*

print()

Enter how many lines? 7

A

BAB

CBABC

DCBABCD

EDCBABCDE

FEDCBABCDEF

GFEDCBABCDEFG

In [40]:

*# Reading number of rows*

row = int(input('Enter how many lines? '))

a = 64

*# Generating pattern*

**for** i **in** range(1,row+1):

*# for space*

**for** j **in** range(1, row+1-i):

print(' ', end='')

*# for increasing pattern*

**for** j **in** range(1,i+1):

print('**%c**' %(a+j), end='')

*# for decreasing pattern*

**for** j **in** range(i-1,0,-1):

print('**%c**' %(a+j), end='')

*# Moving to next line*

print()

Enter how many lines? 6

A

ABA

ABCBA

ABCDCBA

ABCDEDCBA

ABCDEFEDCBA

In [41]:

*# Reading number of rows*

row = int(input('Enter how many lines? '))

a = 65 *# ASCII value for 'A'*

*# Generating pattern*

**for** i **in** range(1,row+1):

*# for space*

**for** j **in** range(1, row+1-i):

print(' ', end='')

*# for pattern*

**for** j **in** range(0, 2\*i-1):

print('**%c**' % (a+j), end='')

*# Moving to next line*

print()

Enter how many lines? 6

A

ABC

ABCDE

ABCDEFG

ABCDEFGHI

ABCDEFGHIJK

In [43]:

*# Hourglass pattern in Python*

*# Reading number of rows*

row = int(input("Enter number of rows: "))

print("Generated Hourglass Pattern is: ")

*# Upper-half*

**for** i **in** range(row, 0, -1):

**for** j **in** range(row-i):

print(" ", end=" ")

**for** j **in** range(1, 2\*i):

print("\*", end=" ")

print()

*# Lower-half*

**for** i **in** range(2, row+1):

**for** j **in** range(row-i):

print(" ", end=" ")

**for** j **in** range(1, 2\*i):

print("\*", end=" ")

print()

Enter number of rows: 6

Generated Hourglass Pattern is:

\* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \*

In [45]:

*# Hollow hourglass pattern in Python*

*# Reading number of rows*

row = int(input("Enter number of rows: "))

print("Generated Hollow Hourglass Pattern is: ")

*# Upper-half*

**for** i **in** range(row, 0, -1):

**for** j **in** range(row-i):

print(" ", end=" ")

**for** j **in** range(1, 2\*i):

**if** i==1 **or** i==row **or** j==1 **or** j==2\*i-1:

print("\*", end=" ")

**else**:

print(" ", end=" ")

print()

*# Lower-half*

**for** i **in** range(2, row+1):

**for** j **in** range(row-i):

print(" ", end=" ")

**for** j **in** range(1, 2\*i):

**if** i==1 **or** i==row **or** j==1 **or** j==2\*i-1:

print("\*", end=" ")

**else**:

print(" ", end=" ")

print()

Enter number of rows: 8

Generated Hollow Hourglass Pattern is:

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* \*

\* \*

\* \*

\* \*

\* \*

\*

\* \*

\* \*

\* \*

\* \*

\* \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

In [47]:

n = int(input('Enter the value of n: '))

**for** i **in** range(1,2\*n):

**for** j **in** range(1,2\*n):

**if** j==n **or** i==j **or** i+j==2\*n:

print('\*', end=' ')

**else**:

print(' ', end=' ')

print()

Enter the value of n: 5

\* \* \*

\* \* \*

\* \* \*

\* \* \*

\*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

In [48]:

*# Python Program to Generate Christmas Tree Pattern*

*# Generating Triangle Shape*

**def** triangleShape(n):

**for** i **in** range(n):

**for** j **in** range(n-i):

print(' ', end=' ')

**for** k **in** range(2\*i+1):

print('\*',end=' ')

print()

*# Generating Pole Shape*

**def** poleShape(n):

**for** i **in** range(n):

**for** j **in** range(n-1):

print(' ', end=' ')

print('\* \* \*')

*# Input and Function Call*

row = int(input('Enter number of rows: '))

triangleShape(row)

triangleShape(row)

poleShape(row)

Enter number of rows: 6

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \*

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

In [52]:

*# Floyd's Triangle in Python*

*# Reading number of row*

row = int(input('Enter number of rows: '))

*# Setting number to 1*

number = 1

print("Floyd's triangle is:")

**for** i **in** range(1,row+1):

**for** j **in** range(1, i+1):

print(number, end='**\t** ')

number += 1

print()

Enter number of rows: 7

Floyd's triangle is:

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

16 17 18 19 20 21

22 23 24 25 26 27 28

In [ ]: