

Description of coding

NO.1

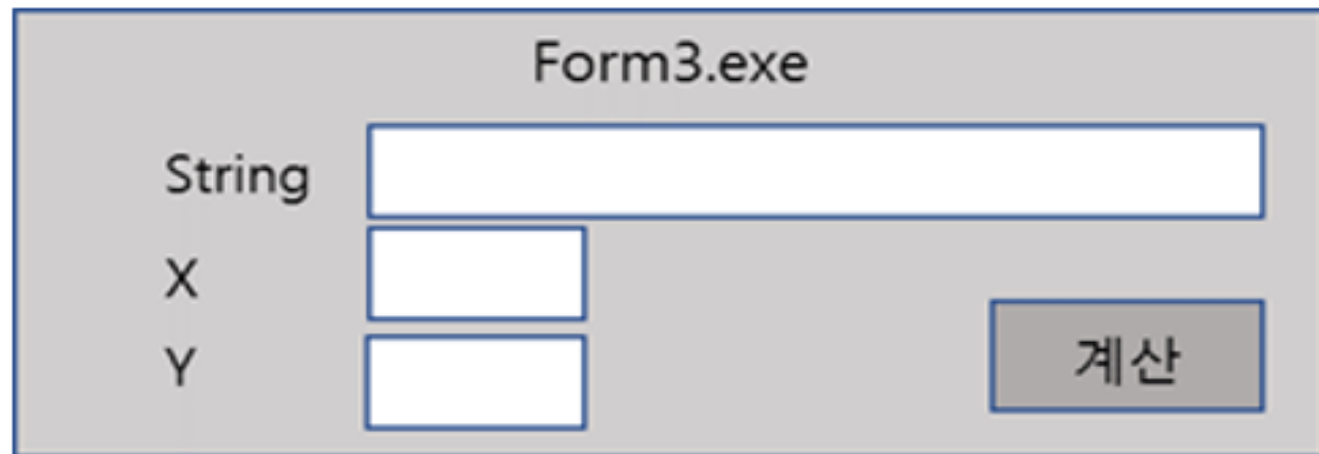
2. Please develop the string calculate function on program

String: $y = 20 + ((10 * x) / (100 - x))$

X input: textbox

Y result: textbox

* The input string is flexible, and you don't need to draw the UI. Just using console is also fine.



The image shows a screenshot of a Windows application window titled "Form3.exe". The window has a light gray background. On the left side, there are three labels: "String", "X", and "Y". To the right of "String" is a wide text input box. To the right of "X" is a smaller text input box. To the right of "Y" is another smaller text input box. On the right side of the window, there is a button with the Korean text "계산" (Gae-san, meaning "Calculate").

NO.1 - 문자열 함수 계산 알고리즘

Def : 중위 표기법(Infix notation)

-> 후위 표기법(postfix notation)

-> 후위 표기법 계산

Issue : 연산자 우선순위, 소괄호 우선순위

Str_fx : 입력 받는 문자열 수식

ioBox : 연산자 스택 대기 공간

tBox : 후위 표기법 완성 공간

Str_fx

$Y = 20 + ((10 * X) / (100 - x))$

ioBox

* / + - (

tBox

20 10 x * 100 x - / +

NO.1 - 문자열 함수 계산 알고리즘

```
class strCalculator:
    chop = dict(zip(['*', '/', '+', '-', '(', ')'], [5, 5, 3, 3, 1, 1]))

    def __init__(self, fx):
        _, self.fx = fx.split('=')

    def change_L(self):
        reC = re.compile(r'(?:(?<=[^\d\.\.])?(?=\d)|(?=[^\d\.\.]))', re.MULTILINE)
        return [x for x in re.sub(reC, ' ', self.fx).split(' ') if x]
```

- chop : 연산자의 우선 순위 비교를 위한 가중치
- def change_L:
입력 받은 문자열 함수 식을 각각 리스트의 원소로 변환

● 정규 표현식

r'(?:' Capture하지 않는 그룹, 여기서는 조건문처럼 쓰인다.
r'(?<=[^\d\.\.])' look backward로 숫자나 점이 아닌 문자가 왼쪽
r' (?=\d) ' 오른쪽은 숫자가 있는 지점
r'|(?=[^\d\.\.])' 만약, 숫자나 점 다음이라면, 그 다음은 숫자나 소수점이 아니어야 한다.

String slicing 보다 9이상의 숫자들도 순서대로 계산 할 수 있어서 유연하다.

● 출력 결과 : ['y','=','20','+','(','(','10','+','10','*','x',')', '/', '(', '10', '-', 'x', ')', ')']

NO.1 - 문자열 함수 계산 알고리즘

```
def postFix(Str, chOp, x):
    tBox = []
    ioBox = []
    for ar in Str:
        if ar.lower() == 'x':
            ar = int(x)
        if ar not in chOp:
            tBox.append(ar)
        elif ar == '(':
            ioBox.append(ar)
        elif ar == ')':
            while ioBox != [] and ioBox[-1] != '(':
                tBox.append(ioBox.pop())
            ioBox.pop()
        else:
            while ioBox != [] and chOp[ioBox[-1]] > chOp[ar]:
                tBox.append(ioBox.pop())
            ioBox.append(ar)
    while ioBox:
        tBox.append(ioBox.pop())
    return(tBox)
```

● 후위 표기법 변환

Str : change_L 함수로 리턴 받은 리스트

chOp : 연산자 비교 dictionary

X = 사용자 입력 x 값

피연산자이면 바로 tBox로 이동하고 연산자이면 if문을 거쳐 우선순위를 결정하여 ioBox에 대기 tBox이동한다.

Else문에서

현 순위의 연산자가 ioBox의 마지막 연산자보다 가중치가 적으면 ioBox의 가장 나중에 들어온 값이 tBox로 이동한다.

ar이 ')'일 때 ioBox의 '(' 만나기 전까지 pop을 시켜 소괄호 우선 순위를 해결한다.

NO.1 - 문자열 함수 계산 알고리즘

```
def workProcess(fx,x):

    Cwork = strCalculator(fx)
    Str = Cwork.change_L()
    chOp = Cwork.chOp
    tBox = postFix(Str,chOp,x)

    calBox = []
    for i in tBox:
        if i not in chOp:
            calBox.append(i)

        if i in chOp:
            op = i
            R = calBox.pop()
            L = calBox.pop()
            calBox.append(Cwork.calRL(L,R,op))

    return(calBox[0])
```

● 후위 표기법 계산

fx : 사용자 입력 문자열 함수 식

x = 사용자 입력 x 값

Cwork class객체 생성 -> postFix함수 실행

-> tBox 후위 표기 리스트 리턴.

tBox에서 피연산자이면 calBox에 이동 연산자이면 calBox에서 뒤에서 2개의 값을 가져와 연산자에 맞는 계산을 수행 한 후 결과 값을 calBox에 입력한다. 주의사항은 먼저 pop한 값이 오른쪽 뒤에 pop한 값이 왼쪽에 위치하여 연산을 수행한다.

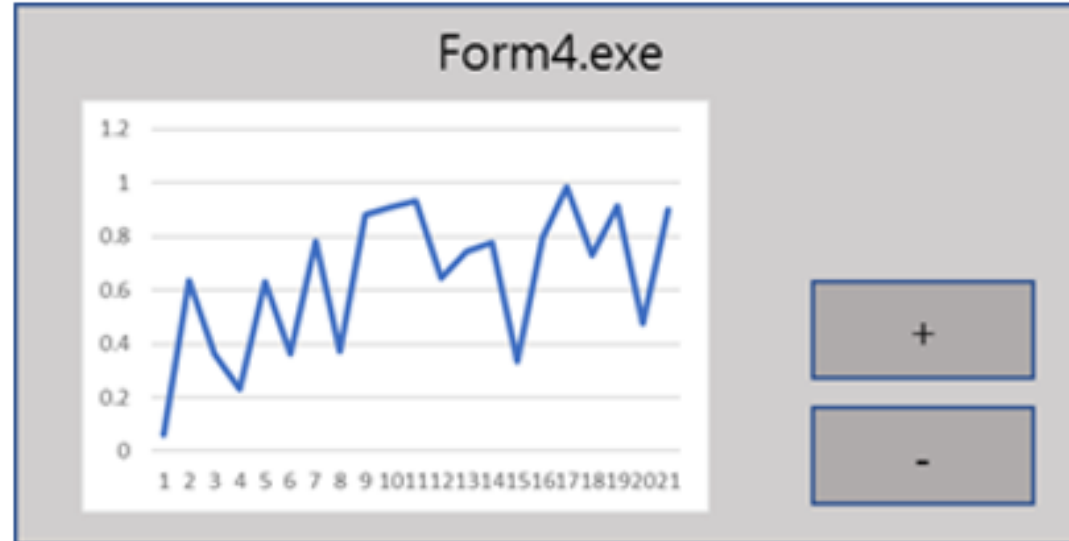
NO.2

3. Please display a draw line chart, and develop changing line thickness function when you click the button.

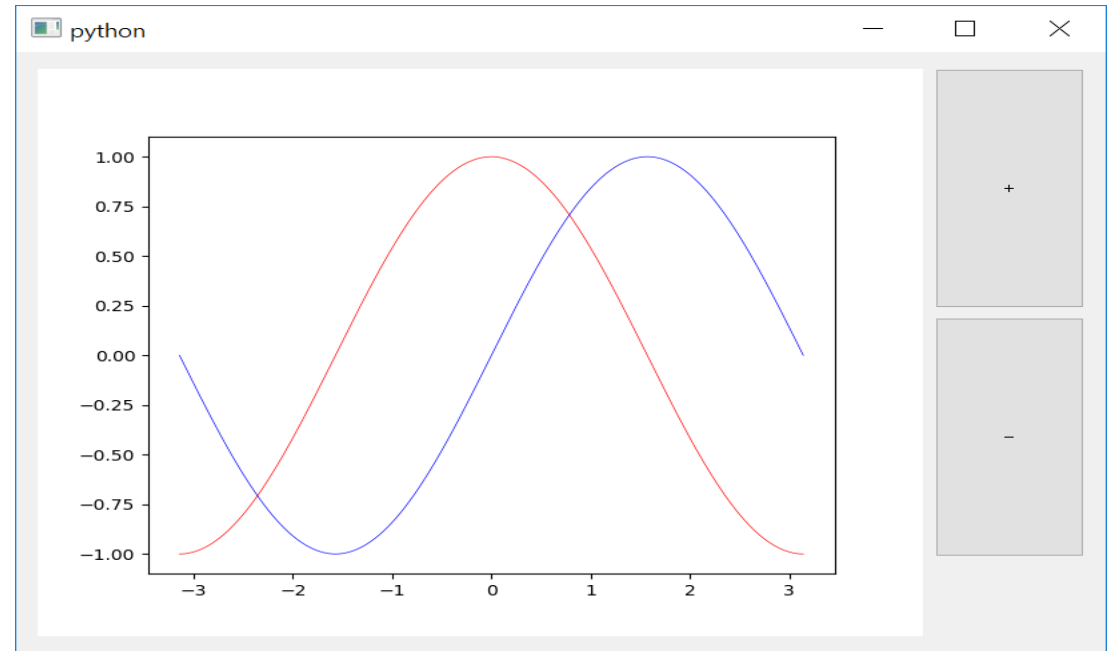
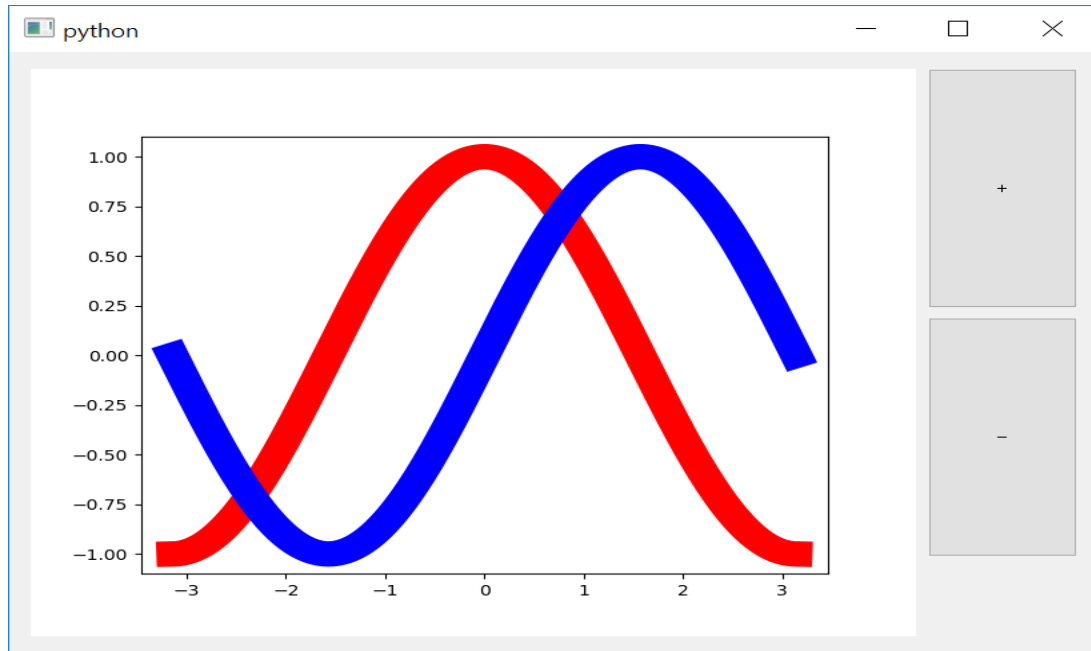
Chart Library: MS Chart (You can use other Chart)

+ Line Width: Button

- Line Width: Button



NO.2 – 차트 이벤트 입력

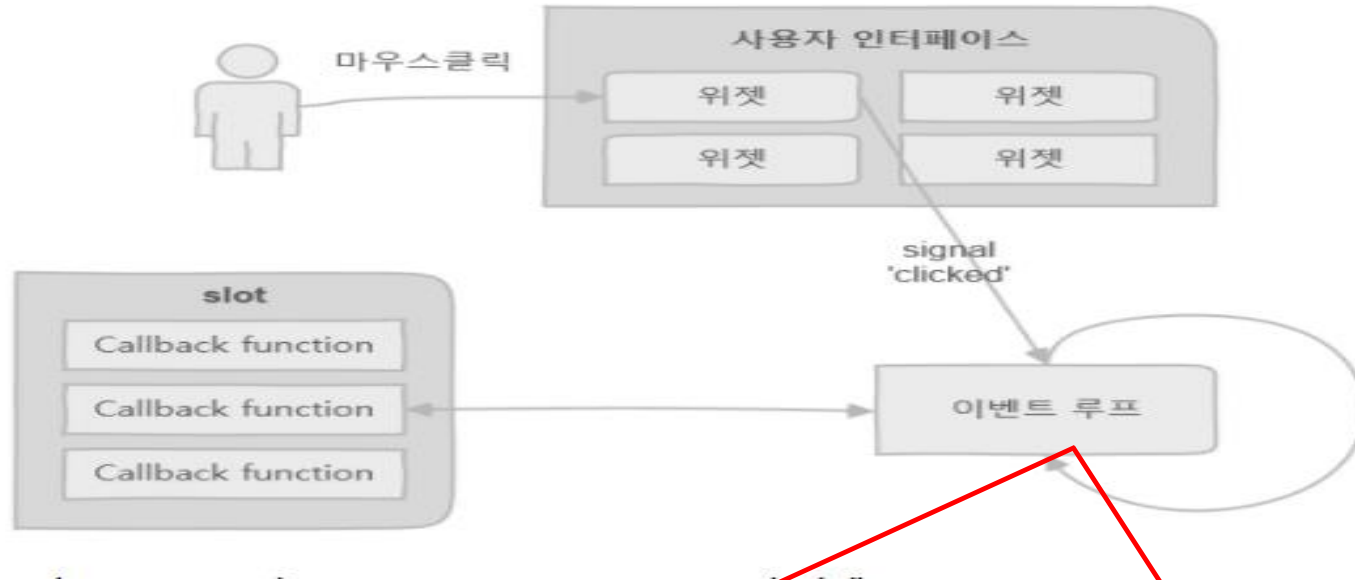


Module : PyQt5

[doc : http://pyqt.sourceforge.net/Docs/PyQt5/](http://pyqt.sourceforge.net/Docs/PyQt5/)

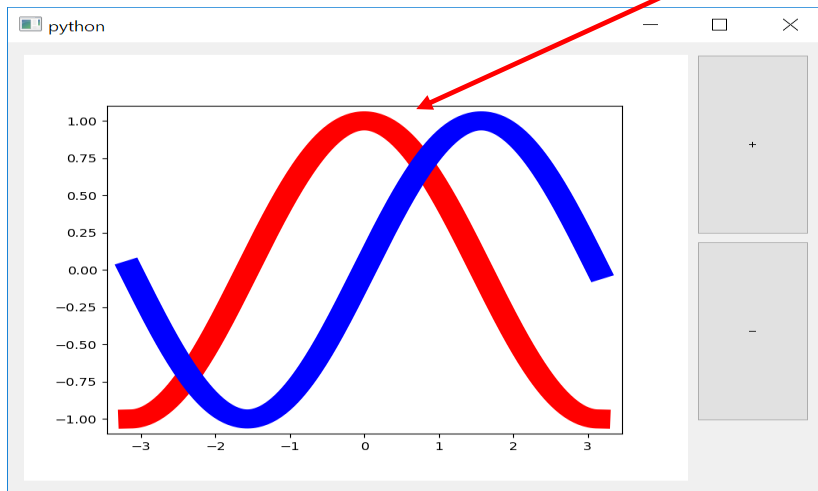
```
import sys
import numpy as np
from PyQt5.QtWidgets import *
import matplotlib.pyplot as plt
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as FigureCanvas
```


NO.2 - PyQt5



GUI 프로그램 작동원리

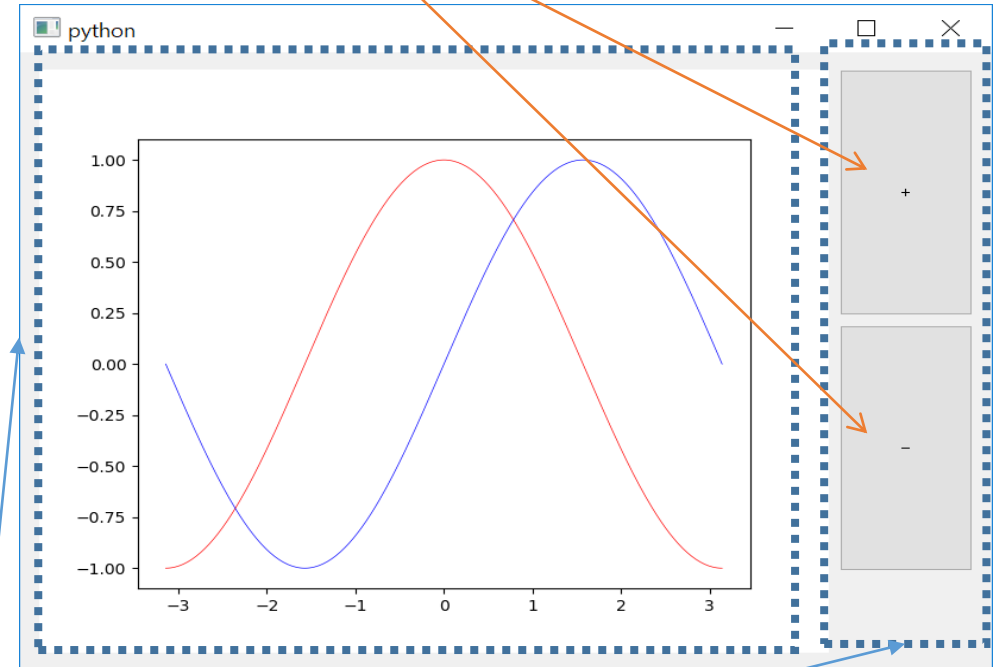
1. 사용자 버튼 위젯 클릭
2. Signal 발생
3. Call slot in signal
4. Event loop



NO.2 - PyQt5

```
def setupUI(self):  
    self.setGeometry(500,200,700,500)  
    self.plus = QPushButton('+')  
    self.plus.clicked.connect(self.putSignal)  
    self.minus = QPushButton('-')  
    self.minus.clicked.connect(self.putSignal)  
    self.plus.setSizePolicy(  
        QSizePolicy.Preferred,  
        QSizePolicy.Expanding)  
    self.minus.setSizePolicy(  
        QSizePolicy.Preferred,  
        QSizePolicy.Preferred)  
    self.fig = plt.Figure()  
    self.canvas = FigureCanvas(self.fig)  
  
    LayButton = QVBoxLayout()  
    LayButton.addWidget(self.plus,3)  
    LayButton.addWidget(self.minus,3)  
    LayButton.addStretch(1)  
  
    LayFig = QVBoxLayout()  
    LayFig.addWidget(self.canvas)  
  
    layout = QHBoxLayout()  
    layout.addLayout(LayFig)  
    layout.addLayout(LayButton)  
    layout.setStretchFactor(LayFig, 1)  
    layout.setStretchFactor(LayButton, 0)  
  
    self.setLayout(layout)
```

버튼 생성



레이아웃

NO.2 - PyQt5

```
def setupUI(self):
    self.setGeometry(500,200,700,500)
    self.plus = QPushButton('+')
    self.plus.clicked.connect(self.putSignal)
    self.minus = QPushButton('-')
    self.minus.clicked.connect(self.putSignal)
    self.plus.setSizePolicy(
        QSizePolicy.Preferred,
        QSizePolicy.Expanding)
    self.minus.setSizePolicy(
        QSizePolicy.Preferred,
        QSizePolicy.Preferred)
    self.fig = plt.Figure()
    self.canvas = FigureCanvas(self.fig)

    LayButton = QVBoxLayout()
    LayButton.addWidget(self.plus,3)
    LayButton.addWidget(self.minus,3)
    LayButton.addStretch(1)

    LayFig = QVBoxLayout()
    LayFig.addWidget(self.canvas)

    layout = QHBoxLayout()
    layout.addLayout(LayFig)
    layout.addLayout(LayButton)
    layout.setStretchFactor(LayFig, 1)
    layout.setStretchFactor(LayButton, 0)

    self.setLayout(layout)
```

- QPushButton : 버튼 위젯 생성
- 객체.clicked.connect(def...) : 시그널 연결(slot)
- 객체.setSizePolicy : 사이즈 재정의
- QVBoxLayout() : 세로로 위젯 나열 클래스
- QHBoxLayout() : 가로로 위젯 나열 클래스
- self.fig = plt.Figure() : matplotlib.figure()
self.canvas = FigureCanvas(self.fig)
- setStretchFactor : 윈도우 창이 변경 될 때 Layout size
변화 유무 선택

NO.2 - PyQt5

```
def putSignal(self):

    Pdraw = self.fig.add_subplot(111)
    sender = self.sender()

    if sender.text() == '+':
        self.prd += 1
    if sender.text() == '-' and self.prd > 0:
        self.prd -= 1

    X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
    Cy, Sy = np.cos(X), np.sin(X)
    Pdraw.plot(X, Cy, color="red", linewidth = self.prd, linestyle="-",
               label = sender.text() + str(self.prd))
    Pdraw.plot(X, Sy, color="blue", linewidth = self.prd, linestyle="-",
               label = sender.text()+ str(self.prd))

    self.canvas.draw()
    Pdraw.clear()
```

● Def putSignal() - Call slot in signal

Canvas layout에 그래프를 그리기 위한 함수

Sender()메소드 이용 이벤트를 호출한 객체의 이름을 가져와 이벤트 버튼 구분.

prd를 클래스 변수로 만들어 들어온 신호에 따라 +,-를 결정한다.

Numpy를 이용하여 코사인,사인 배열을 만든다.

Pdraw객체를 사용하여 구분된 prd 값으로 그래프 생성

Pdraw.clear() : 변경된 그래프를 canvas에 표현

NO.3

4. Please develop a program to show output from 1 to 100 as below image:

Display: we don't care. (Console or Textbox or Message box or etc.)

* 100 can be changed to any integer number

1	2	3	4
5	6	7	8
9			

NO.3 – 숫자 삼각형 만들기

```
def triangle(a):  
    c = [i for i in range(0, a+1)]  
    j = 0  
    for i in range((a//10-1)*2):  
        if i % 2 != 0:  
            j += i  
            print(' '.join(list(map(lambda x : str(x), c[j-i+1:j+1]))).center(100))  
            if c[j-i+1:j+1] == []:  
                break
```

문제에서의 숫자로 이루어진 삼각형은 규칙이 있다.

각 행의 숫자 길이는 홀수의 배열이기 때문에 각 행의 가장 끝자리는 홀수의 합 배열이다.

그렇다면 다음 행의 가장 앞의 숫자는 홀수의 합 배열에서 홀수의 배열을 빼주면 앞자리의 숫자를 구할 수 있다.

C[시작 값 : 마지막 값].center()

처음 이 문제를 접했을 때 생각난 함수가 center메소드(문자열을 중앙으로)였으며 그를 기반한 숫자 삼각형 코딩입니다.