



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA

Actividad 3

programación Orientada a objetos

Grupo 4

Estudiante

Andres Felipe Devia Orrego

Docente

Walter Arboleda

Medellin

2023

# CONTENIDO

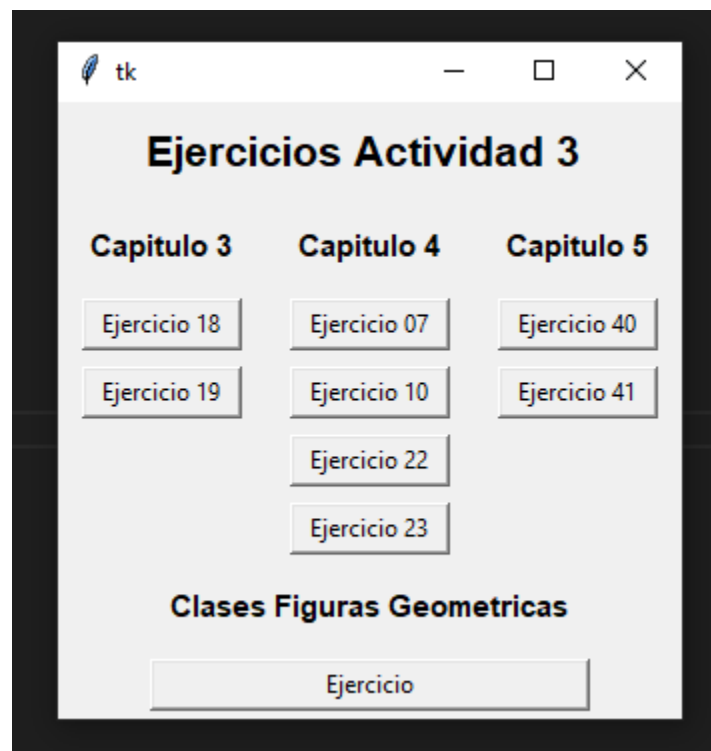
Introducción.....	4
Capítulo 3 .....	5
Ejercicio 18.....	5
Clase .....	5
Diagrama de clase .....	5
Interfaz Grafica.....	6
Ejercicio 19.....	7
Clase .....	7
Diagrama de clase .....	7
Interfaz Grafica.....	8
CAPITULO 4 .....	9
Ejercicio 07.....	9
Clase .....	9
Diagrama de clase .....	9
Interfaz Grafica.....	10
Ejercicio 10.....	11
Clase .....	11
Diagrama de clase .....	11
Interfaz Grafica.....	12
Ejercicio 22.....	13
Clase .....	13
Diagrama de clase .....	13
Interfaz Grafica.....	14
Ejercicio 23.....	15
Clase .....	15
Diagrama de clase .....	15
Interfaz Grafica.....	16
CAPITULO 5 .....	17
Ejercicio 40.....	17
Clase .....	17
Diagrama de clase .....	17
Interfaz Grafica.....	18

Ejercicio 41 .....	19
Clase .....	19
Diagrama de clase .....	19
Interfaz Grafica .....	20
Clases Figuras Geométricas .....	21
Interfaz General .....	21
Diagrama de Clases .....	21
Circulo .....	22
Rectángulo .....	23
Cuadrado .....	24
Triangulo .....	25
Rombo .....	26
Trapecio .....	27

# INTRODUCCIÓN

Los ejercicios fueron desarrollados en **Python** y divididos en una clase cada ejercicio, se mostrara la interfaz de cada ejercicio, junto con su clase, su diagrama de clase y su resultado.

La interfaz grafica base de la actividad es la siguiente:



Donde cada ejercicio será explicado a continuación:

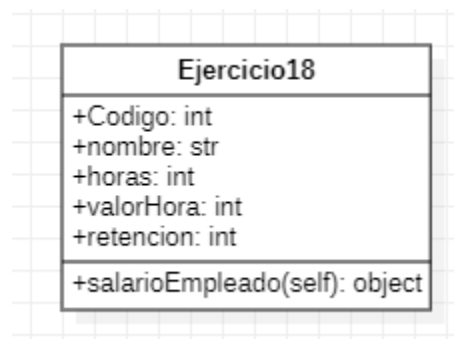
## CAPÍTULO 3

### Ejercicio 18

#### Clase

```
3 class ejercicio_18():
4     def __init__(self,Codigo,nombre,horas,valorHora,retencion):
5         self.Codigo=Codigo
6         self.nombre=nombre
7         self.horas=horas
8         self.valorHora=valorHora
9         self.retencion=retencion
10    def salarioEmpleado(self):
11        codigo=self.Codigo
12        nombre=self.nombre
13        horas=self.horas
14        valorHora=self.valorHora
15        retencion=self.retencion
16        salarioBruto=valorHora*horas
17        porcentaje=retencion/100
18        salarioNeto=salarioBruto-(salarioBruto*porcentaje)
19        return {
20            "salarioNeto":salarioNeto,
21            "salarioBruto":salarioBruto
22        }
23
```

#### Diagrama de clase



## Interfaz Grafica

**Ejercicio Actividad 18**

En el siguiente ejercicio se debe introducir la informacion solicitada del empleado  
y se retornara el codigo, nombre, salario bruto y salario neto

Codigo Del Empleado: 100141852      Nombre: Andres Felipe

Horas trabajadas al mes: 48      Valor por hora trabajada: 45000

Porcentaje de retencion en la fuente: 2     

**Respuesta**

Codigo	Nombre	Salario Bruto	Salario Neto
100141852	Andres Felipe	2160000	2116800

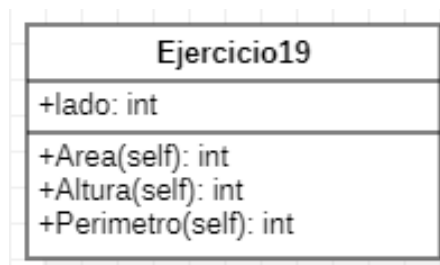
Como se puede observar el código realiza el calculo del salario bruto y realiza el descuento del salario neto.

## Ejercicio 19

### Clase

```
24 class ejercicio_19():
25     def __init__(self,lado):
26         self.lado=lado
27     def Area(self):
28         lado=self.lado
29         Area=(math.sqrt(3)*(lado*lado))/4
30         Area=round(Area,2)
31         return Area
32     def Altura(self):
33         lado=self.lado
34         Altura=(math.sqrt(3)*lado)/2
35         Altura=round(Altura,2)
36         return Altura
37     def Perimetro(self):
38         lado=self.lado
39         Perimetro=3*lado
40         Perimetro=round(Perimetro,2)
41         return Perimetro
42
```

### Diagrama de clase



## Interfaz Grafica

**Ejercicio Actividad 19**

En el siguiente ejercicio se debe introducir el valor de un lado de un triangulo equilatero  
y se retornara el area, la altura, y el perimetro del triangulo

**Lado del triangulo**

Enviar

**Respuesta**

Area	Altura	Perimetro
3.9	2.6	9.0

Regresar

Como se puede observar el código realiza el calculo de la altura, el área, y el perímetro, teniendo como dato de entrada, el lado de un triángulo equilátero.



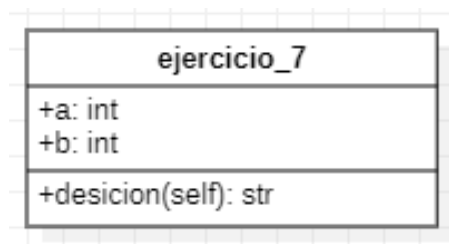
## CAPITULO 4

### Ejercicio 07

#### Clase

```
class ejercicio_7():  
    def __init__(self,a,b):  
        self.a=a  
        self.b=b  
    def desicion(self):  
        a=self.a  
        b=self.b  
        mensaje=""  
        if a > b:  
            mensaje=str(a)+" Es mayor que "+str(b)  
        elif a < b:  
            mensaje=str(a)+" Es menor que "+str(b)  
        else:  
            mensaje=str(a)+" Es igual que "+str(b)  
        return mensaje
```

#### Diagrama de clase



## Interfaz Grafica

tk

### Ejercicio Actividad 07

En el siguiente ejercicio se debe introducir dos valores numericos  
y se un mensaje diciendo si A es mayor, menor o igual que B

Valor de A  Valor de B

**Respuesta**

**Mensaje**

12 Es menor que 15

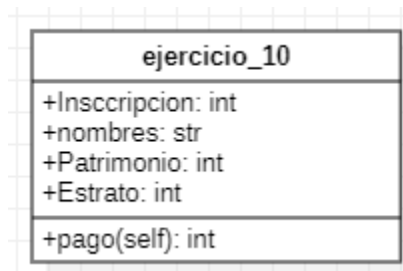
Como se puede observar el código realiza la operación para poder indicar si el valor de A, es menor, igual o mayor, que el valor de B.

## Ejercicio 10

### Clase

```
59 class ejercicio_10():
60     def __init__(self,Inscripcion,nombres,Patrimonio,Estrato,):
61         self.Inscripcion=Inscripcion
62         self.nombres=nombres
63         self.Patrimonio=Patrimonio
64         self.Estrato=Estrato
65     def pago(self):
66         Inscripcion=self.Inscripcion
67         nombres=self.nombres
68         Patrimonio=self.Patrimonio
69         Estrato=self.Estrato
70
71         valor=50000
72         if Patrimonio>2000000 and Estrato>3:
73             valor=(Patrimonio*0.03)+valor
74
75         return valor
```

### Diagrama de clase



## Interfaz Grafica

**Ejercicio Actividad 10**

En el siguiente ejercicio se debe introducir la informacion socioeconomica de un estudiante  
y se retornara el valor que debe pagar el estudiante de su matricula

Numero de Inscripcion: 456152      Nombre: Andres Felipe

Patrimonio: 1500000      Estrato Social: 4

**Respuesta**

Inscripcion	Nombre	Patrimonio	Pago Total
456152	Andres Felipe	1500000	50000

Regresar

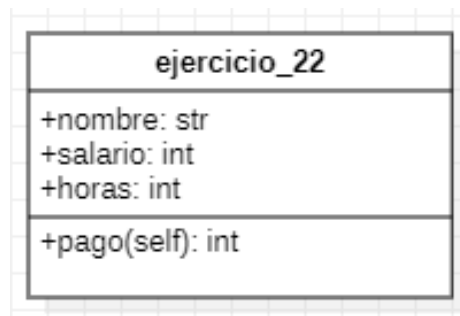
Como se puede observar el código realiza la operación para calcular el pago total de la matricula de un estudiante

## Ejercicio 22

### Clase

```
77 class ejercicio_22():
78     def __init__(self,nombre,salario,horas):
79         self.nombre=nombre
80         self.salario=salario
81         self.horas=horas
82     def pago(self):
83         mensual=int(self.salario)*int(self.horas)
84         if mensual>450000 :
85             return mensual
```

### Diagrama de clase



## Interfaz Grafica

The screenshot shows a Tkinter window titled "tk" with a standard Mac OS window title bar (close, zoom, and scroll buttons). The main content area has a light gray background and contains the following elements:

- Ejercicio Actividad 22**: A bold black title at the top center.
- Instructions**: A paragraph of text stating: "En el siguiente ejercicio se debe introducir la informacion de un empleado y se retornara el nombre y salario mensual si este es mayor a 450000, si no lo es, solo se mostrara el nombre".
- Form Fields**:
  - Nombre del Empleado**: A text input field containing "Andres Felipe".
  - Salario Basico por Hora**: A text input field containing "45000".
  - Horas trabajadas en el Mes**: A text input field containing "58".
- Enviar Button**: A button labeled "Enviar" located to the right of the "Horas trabajadas en el Mes" field.
- Resultado Section**: A section titled **Respuesta** in green text, followed by a table with two columns: **Nombre** and **Salario Mensual**. The table contains one row with the values "Andres Felipe" and "2610000".
- Regresar Button**: A button labeled "Regresar" located at the bottom left of the window.

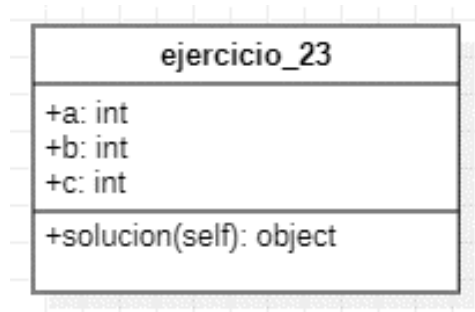
Como se puede observar el código realiza la operación para calcular el salario mensual de un empleado.

## Ejercicio 23

### Clase

```
87 class ejercicio_23():
88     def __init__(self,a,b,c):
89         self.a = a
90         self.b = b
91         self.c = c
92     def solucion(self):
93         valores = [self.a,self.b,self.c]
94         disc=(valores[1]**2)-(4*valores[0]*valores[2])
95         if disc<0:
96             return{
97                 "valor1":"No tiene Solucion",
98                 "valor2":"No tiene Solucion"
99             }
100         resultado1 = ((-1*valores[1])+math.sqrt((valores[1]**2)-(4*valores[0]*valores[2])))/(2*valores[0])
101         resultado2 = ((-1*valores[1])-math.sqrt((valores[1]**2)-(4*valores[0]*valores[2])))/(2*valores[0])
102         return{
103             "valor1":resultado1,
104             "valor2":resultado2
105         }
106
```

### Diagrama de clase



## Interfaz Grafica

The image shows a Tkinter window titled 'tk' with a light gray background. At the top, the title bar contains the text 'tk' and standard window controls (minimize, maximize, close). The main content area has a title 'Ejercicio Actividad 23' in bold black font. Below the title, a text label reads: 'En el siguiente ejercicio se debe introducir los los parametros de la ecuacion  $AX^2+BX+C$  y se retornara las posibles soluciones a la ecuacion cuadratica'. There are three input fields: 'Valor de A' with the value '1', 'Valor de B' with the value '6', and 'Valor de C' with the value '8'. To the right of the 'Valor de C' field is an 'Enviar' button. Below these inputs, the word 'Respuesta' is displayed in green. Underneath, there are two labels: 'Valor 1' and 'Valor 2'. Below 'Valor 1' is the value '-2.0', and below 'Valor 2' is the value '-4.0'. At the bottom left, there is a 'Regresar' button.

**Ejercicio Actividad 23**

En el siguiente ejercicio se debe introducir los los parametros de la ecuacion  $AX^2+BX+C$   
y se retornara las posibles soluciones a la ecuacion cuadratica

Valor de A: 1      Valor de B: 6

Valor de C: 8      **Enviar**

**Respuesta**

**Valor 1**      **Valor 2**

-2.0      -4.0

**Regresar**

Como se puede observar el código realiza las operaciones necesarias para hallar los valores de la ecuación cuadratica



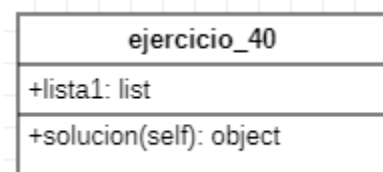
## CAPITULO 5

### Ejercicio 40

Clase

```
107 class ejercicio_40():
108     def __init__(self, lista1):
109         self.lista1 = lista1
110     def solucion(self):
111         lista1=self.lista1
112         cuadrados=[]
113         raicez=[]
114         cubos=[]
115         for i in lista1:
116             cuadrados.append(i**2)
117             raicez.append(round(math.sqrt(i),3))
118             cubos.append(i**3)
119         print(cubos)
120         return{
121             "cuadrados":cuadrados,
122             "raicez":raicez,
123             "cubos":cubos
124         }
125
```

Diagrama de clase



## Interfaz Grafica

**Ejercicio Actividad 40**

En el siguiente ejercicio se debe introducir una lista de numeros  
y se retornara la raiz cuadrada, el cuadrado, y el cubo de cada numero

Lista de numeros:  
2,1,3,5,

Numero a añadir a la lista  Añadir

Enviar

**Respuesta**

	Cuadrados	Cubos	Raicez
2	4	8	1.414
1	1	1	1.0
3	9	27	1.732
5	25	125	2.236

Regresar

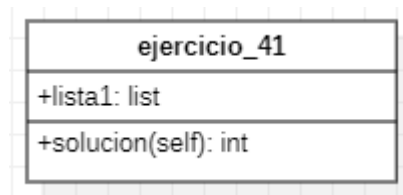
Como se puede observar el código realiza los cuadrados, cubos, y raíces de la lista de números que introduzcamos.

## Ejercicio 41

### Clase

```
126 class ejercicio_41():
127     def __init__(self, lista1):
128         self.lista1 = lista1
129     def solucion(self):
130         lista1=self.lista1
131         mayor=0
132         for i in lista1:
133             if i>mayor:
134                 mayor=i
135         return mayor
136
```

### Diagrama de clase



## Interfaz Grafica

tk

### Ejercicio Actividad 41

En el siguiente ejercicio se debe introducir una lista de numeros  
y se retornara el numero mayor entre todos

Lista de numeros:  
3,1,5,2,8,13,15,2,29,1,

Numero a añadir a la lista  Añadir

Enviar

## Respuesta

El numero mayor de la lista es **29**

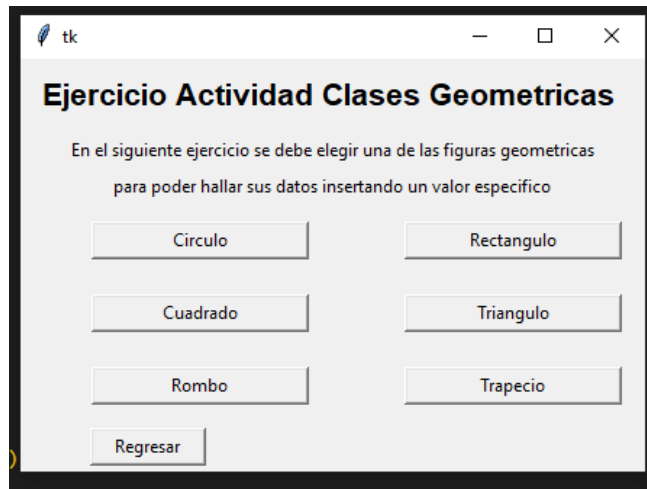
Regresar

Como se puede observar el código acumula una lista de números y retorna el mayor de todos estos.

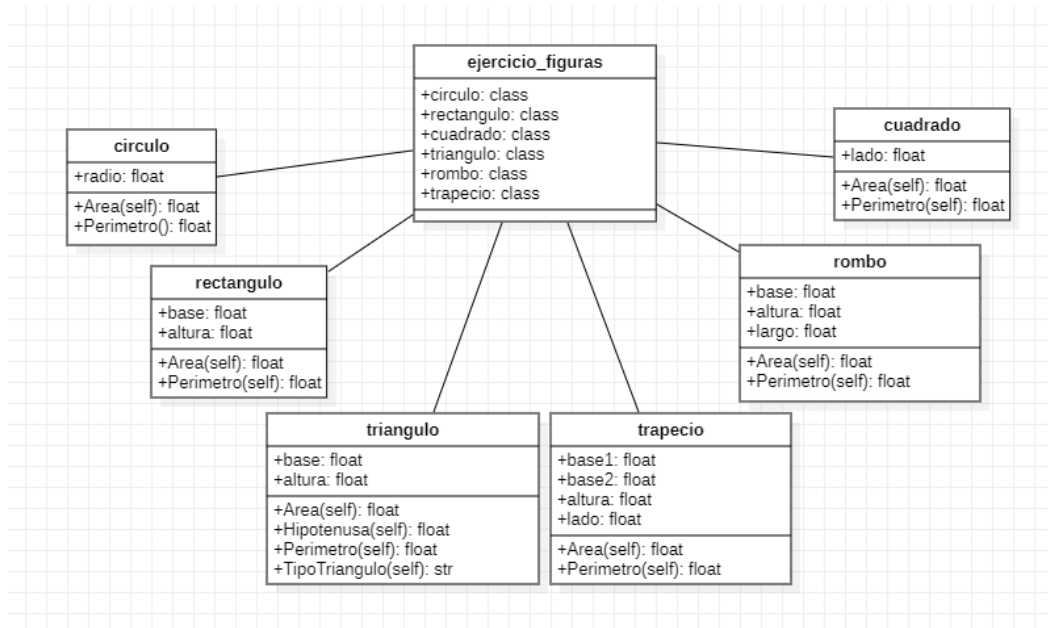
# CLASES FIGURAS GEOMÉTRICAS

Para este conjunto de ejercicios mostrar su interfaz de manera individual, pero su diagrama se representara a continuación:

## Interfaz General



## Diagrama de Clases



## Circulo

```
138 class circulo():
139     def __init__(self,radio):
140         self.radio = radio
141     def Area(self):
142         return math.pi*math.pow(self.radio,2)
143     def Perimetro(self):
144         return 2*math.pi*self.radio
```

The image shows a Tkinter window titled "tk" with a title bar containing standard window controls. The window content is as follows:

- Valores del Circulo**: Main title of the window.
- En el siguiente ejercicio se debe introducir Radio del circulo y se devolvera el valor del area y del perimetro**: Instructional text.
- Radio del Circulo**: Label for the input field.
- : Text input field containing the value 3.
- Enviar**: Button to submit the input.
- Respuesta**: Large green text indicating the result is displayed below.
- Area**: Label for the calculated area.
- Perimetro**: Label for the calculated perimeter.
- 28.274333882308138**: The calculated area value.
- 18.84955592153876**: The calculated perimeter value.
- Regresar**: Button to return to the input screen.

## Rectángulo

```
145 class rectangulo():
146     def __init__(self,base,altura):
147         self.base = base
148         self.altura = altura
149     def Area(self):
150         return self.base*self.altura
151     def Perimetro(self):
152         return (2*self.base)+(2*self.altura)
```

A Tkinter window titled "Valores del Rectangulo" with a light gray background. The window contains the following elements:

- A title bar with a feather icon, the text "tk", and standard window controls (minimize, maximize, close).
- A main title "Valores del Rectangulo" in bold black font.
- Instructions: "En el siguiente ejercicio se debe introducir Base y Altura del Rectangulo y se devolvera el valor del area y del perimetro".
- Input fields: "Base del rectangulo" with value "2" and "Altura del Rectangulo" with value "3".
- An "Enviar" button.
- A green "Respuesta" label.
- Output fields: "Area" with value "6.0" and "Perimetro" with value "10.0".
- A "Regresar" button at the bottom left.

Area	Perimetro
6.0	10.0

## Cuadrado

```
153 class cuadrado():
154     def __init__(self,lado):
155         self.lado = lado
156     def Area(self):
157         return self.lado**2
158     def Perimetro(self):
159         return 4*self.lado
```

tk

### Valores del Cuadrado

En el siguiente ejercicio se debe introducir el lado del cuadrado y se devolvera el valor del area y del perimetro

lado del Cuadrado

Enviar

### Respuesta

Area	Perimetro
25.0	20.0

Regresar



## Triangulo

```
160 class triangulo():
161     def __init__(self,base,altura):
162         self.base = base
163         self.altura = altura
164     def Area(self):
165         return (self.altura*self.base)/2
166     def Hipotenusa(self):
167         return math.sqrt((self.altura**2)+(self.base**2))
168     def Perimetro(self):
169         return self.base+self.altura+self.Hipotenusa()
170     def TipoTriangulo(self):
171         base=self.base
172         altura=self.altura
173         hipotenusa=self.Hipotenusa()
174         if base==altura and base==hipotenusa and altura==hipotenusa:
175             return "Equilatero"
176         elif base!=altura and base!=hipotenusa and altura!=hipotenusa:
177             return "Escaleno"
178         else:
179             return "Isoceles"
```

tk

### Valores del Triangulo

En el siguiente ejercicio se debe introducir Base y Altura de un Triangulo Rectangulo y se devolvera el valor del area, el perimetro y el tipo de triangulo

Base del Triangulo

Altura del Triangulo

### Respuesta

Area	Tipo de Triangulo	Perimetro
6.0	Escaleno	12.0

## Rombo

```
180 class rombo():
181     def __init__(self,lado,ancho,largo):
182         self.lado = lado
183         self.ancho = ancho
184         self.largo = largo
185     def Area(self):
186         return (self.largo*self.ancho)/2
187     def Perimetro(self):
188         return 4*self.lado
```

tk

### Valores del Rombo

En el siguiente ejercicio se debe introducir Ancho, Largo y el lado de un Rombo y se devolvera el valor del area y del perimetro

Ancho del Rombo

Largo del Rombo

Lado del Rombo

**Respuesta**

Area	Perimetro
7.5	8.0

## Trapezio

```
189 class trapezio():
190     def __init__(self,lado,altura,base1,base2):
191         self.altura = altura
192         self.lado = lado
193         self.base1 = base1
194         self.base2 = base2
195     def Area(self):
196         return ((self.base1+self.base2)/2)*self.altura
197     def Perimetro(self):
198         return self.lado+self.lado+self.base1+self.base2
199
```

 tk

### Valores del Trapecio

En el siguiente ejercicio se debe introducir las 2 bases, la altura y el lado de un Trapecio y se devolvera el valor del area y del perimetro

Base 1 del Trapecio	<input type="text" value="5"/>
Base 2 del Trapecio	<input type="text" value="8"/>
Lado del Trapecio	<input type="text" value="2"/>
Altura del Trapecio	<input type="text" value="3"/>

Enviar

### Respuesta

Area	Perimetro
19.5	17.0

Regresar