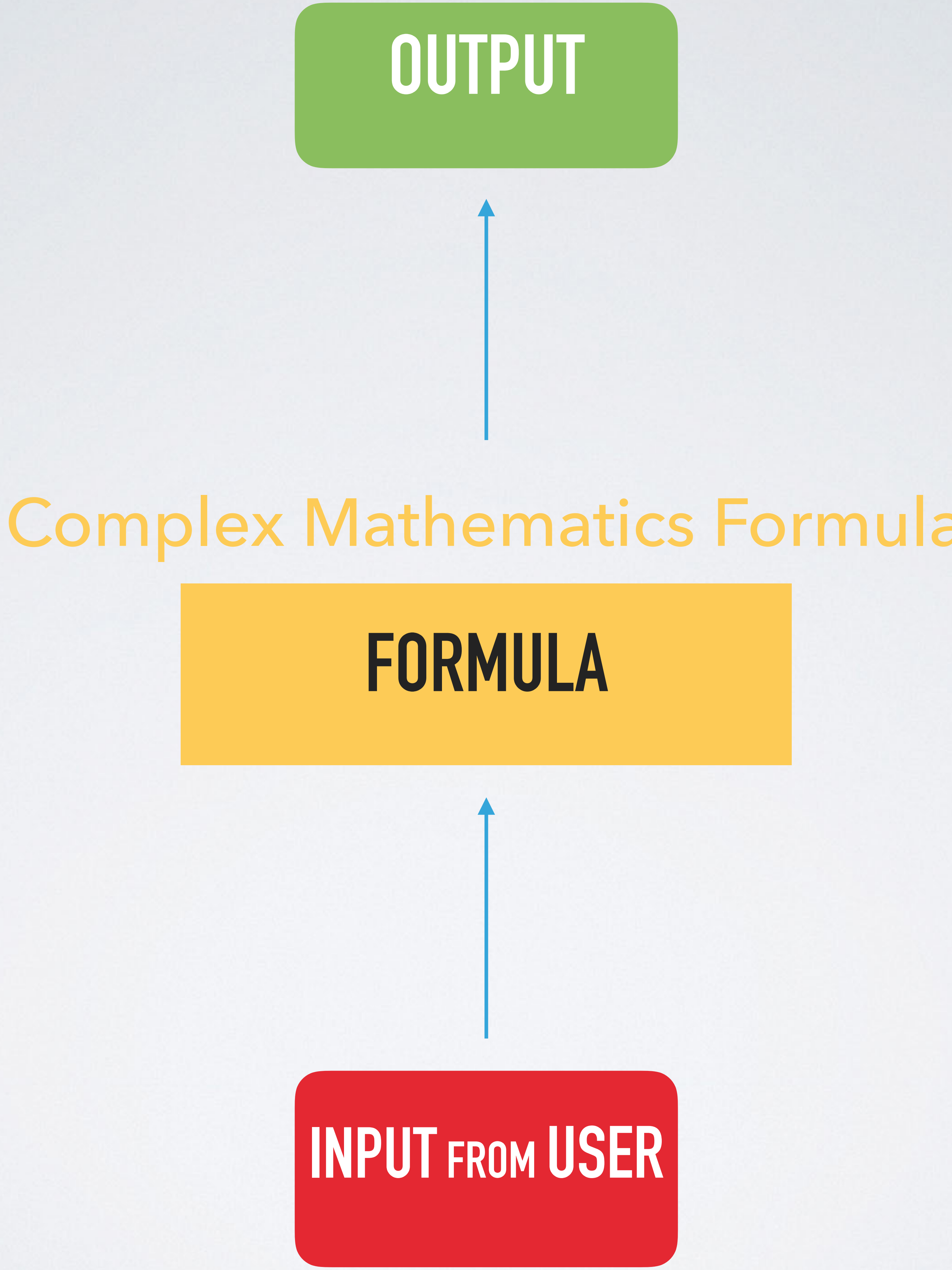
NUMERICAL METHODS IN PYTHON

WORKING


OUTPUT
Complex Mathematics Formula
FORMULA
INPUT FROM USER

Storage

VARIABLE = 5

What is Coding?

Run it ONCE and then REPEAT

For Coding what we require to learn?

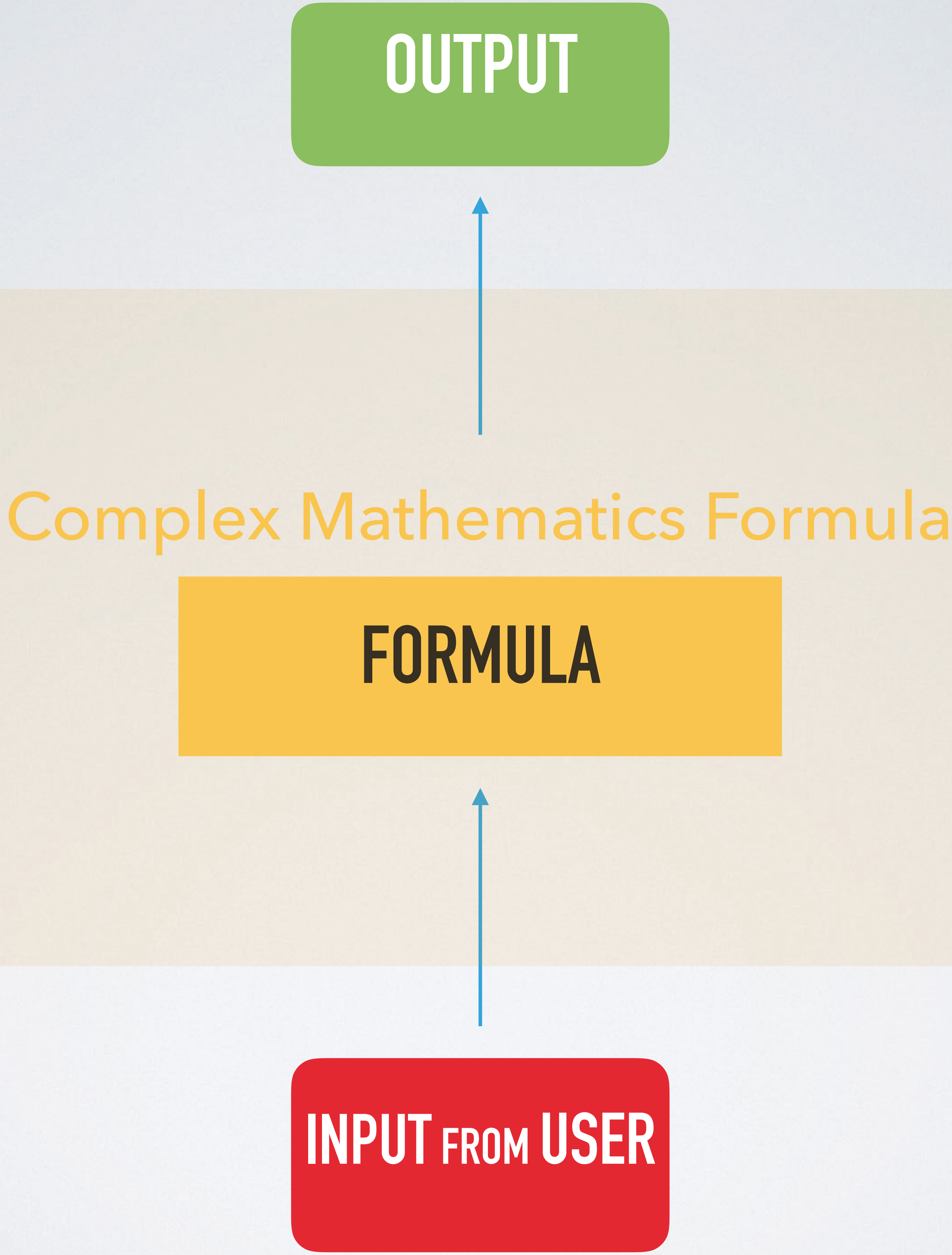
1. *UnderStand DATA Flow*

2. *Learn to teach first step to Computer*

3. *Never Focus on Language*

cmath math

WORKING



Complex Mathematics **cmath**

Mathematics **math**

Statistics

Numpy

Matplotlib

QUADRATIC EQUATION

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
d = (b**2) - (4*a*c)
```

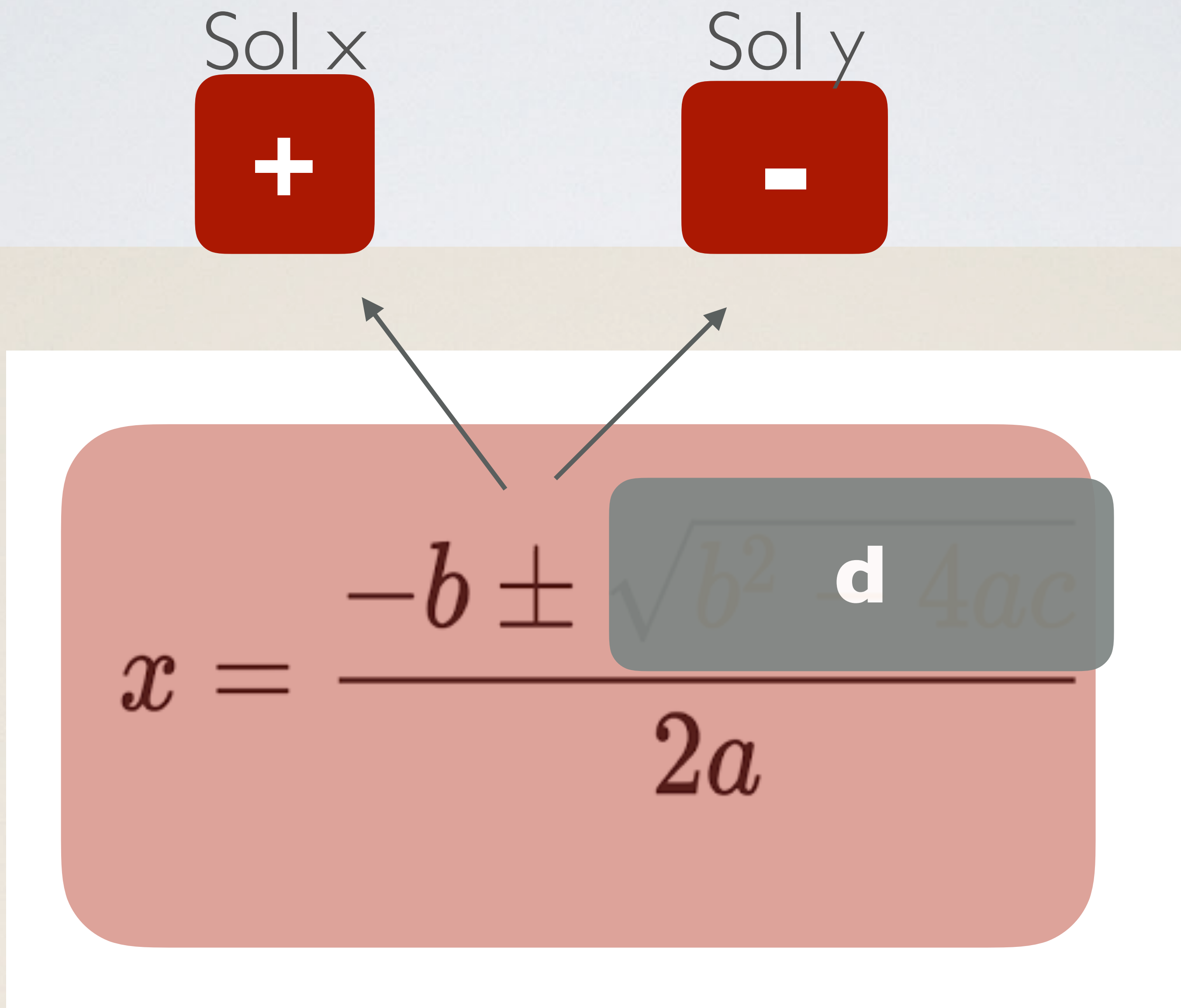
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Function

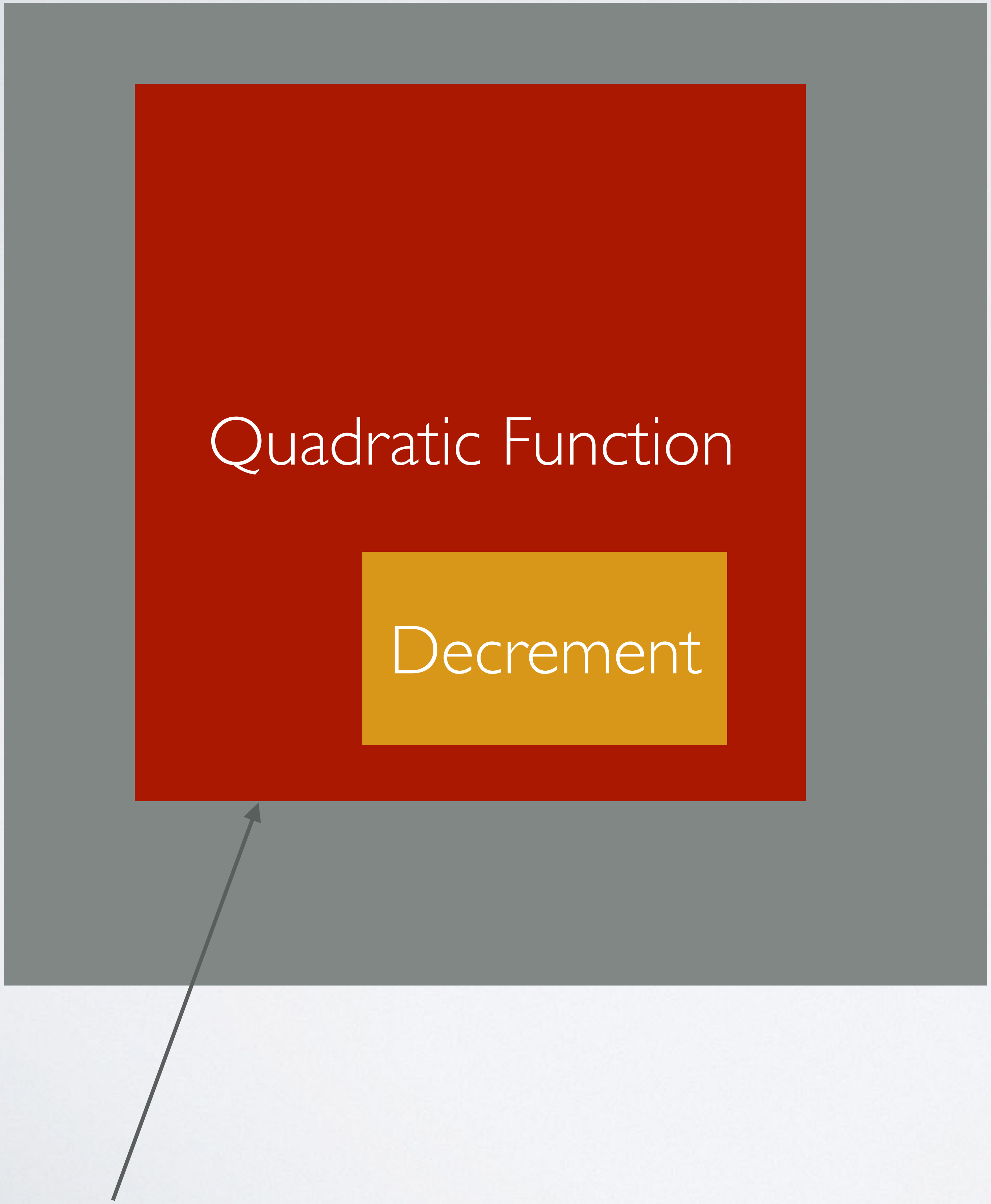
```
def decrement(a,b,c):
    d = (b**2) - (4*a*c)
        return d
```

A = 1
B = 5
C = 6

```
solX = (-b - cmath.sqrt(decrement(a, b, c))) / (2*a)
```




```
print(“ Hello World {} {} ”.format(sol_x, sol_y))
```


Quadratic Function
Decrement

A = 1
B = 5
C = 6

a = int(input('a: '))
b = int(input('b: '))
c = int(input('c: '))

A = input(‘a: ’)

A = int(input(‘a: ’))

```
def decrement(a=a, b=b, c=c):
        d = (b**2) - (4*a*c)
            return d
```

decrement = **decrement()**

Default Values

Conditional Statements

If decrement > 0 :

print("Decrement is greater than 0")

elif decrement == 0 :

print("Decrement is equal to 0")

else:

print("All other Cases")

==

>

<

!=

FACTORIAL OF A NUMBER

FOR LOOP

for i in range(1, 10):

print('Hello World') → 9

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

print('Hello World') → 10

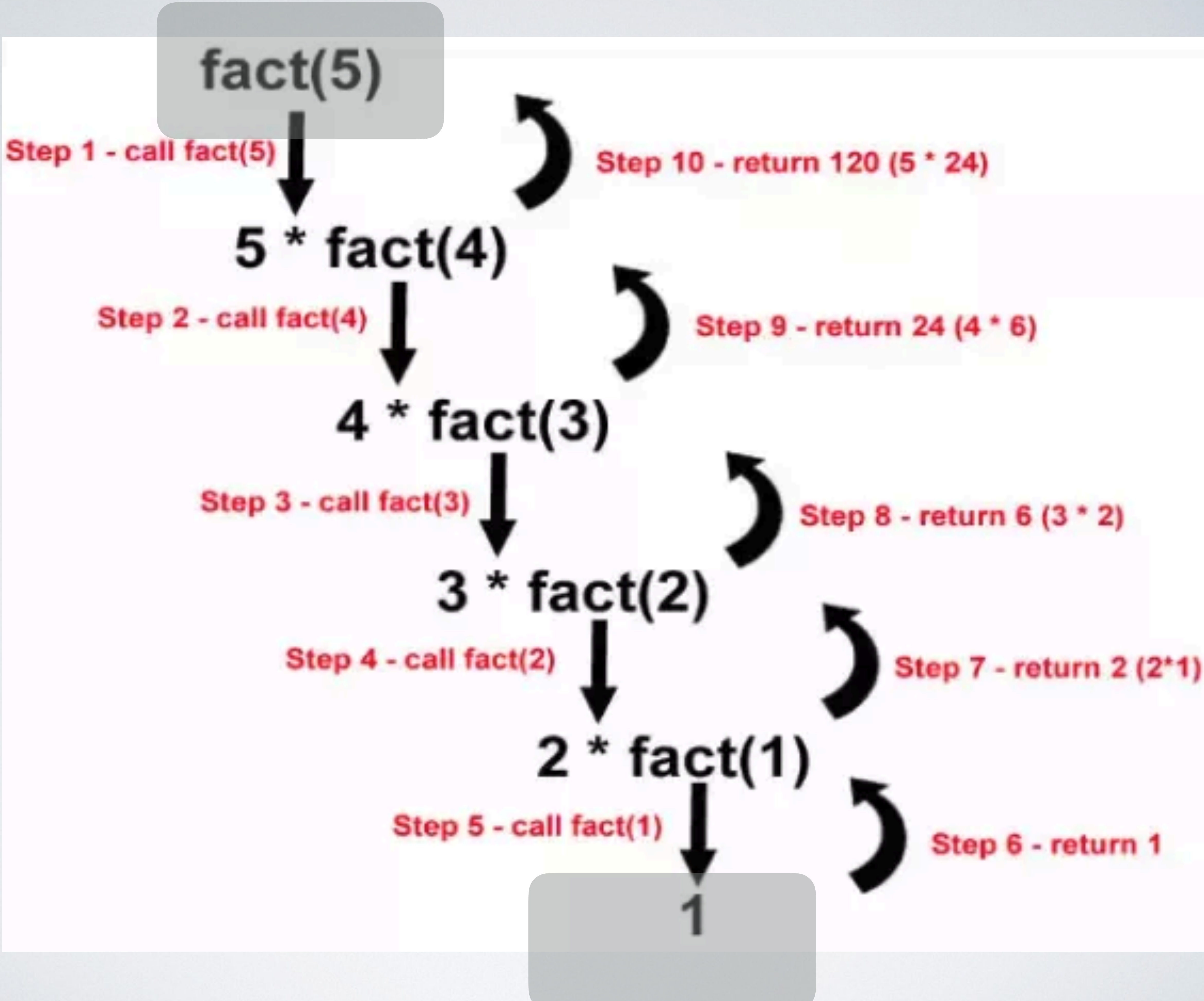
RANGE

range(1, 10)

1 2 3 4 5 6 7 8 9

range(**[start], stop[, step]**)
`start`: Starting number of the sequence.
`stop`: Generate numbers up to, but not including this number.
`step`: Difference between each number in the sequence.

RECURSIVE


fact(5)
Step 1 - call fact(5)
Step 10 - return 120 (5 * 24)
5 * fact(4)
Step 2 - call fact(4)
Step 9 - return 24 (4 * 6)
4 * fact(3)
Step 3 - call fact(3)
Step 8 - return 6 (3 * 2)
3 * fact(2)
Step 4 - call fact(2)
Step 7 - return 2 (2*1)
2 * fact(1)
Step 5 - call fact(1)
Step 6 - return 1
1

FOR LOOP

iter() → Iterator

next(iter()) → Next Iterator

```
For element in myList:
    print(element)
```

(tab)

← **EXAMPLE**

EULER METHOD

Euler's Method

The Taylor series can be written as

$$y(x+h) = y(x) + y'(x)h + \frac{y''(x)}{2!}h^2 + \frac{y'''(x)}{3!}h^3 + \cdots$$

By truncation the series at the first derivative term, the approximate solution of Euler's method is obtained. Thus

$$y(x+h) = y(x) + y'(x)h$$

The initial conditions in this case should be the value of $y(x)$ at initial x. This method is known as point-slope method because it predicts the next point using the slope $y'(x)$.

Example

Example 1: Find the numerical solution of the following differential equation over the domain [0, 2].

$$y' = xy, \quad y(0) = 1$$

Analytical Solution: $y = e^{x^2/2}$

$y' = xy,$ Question

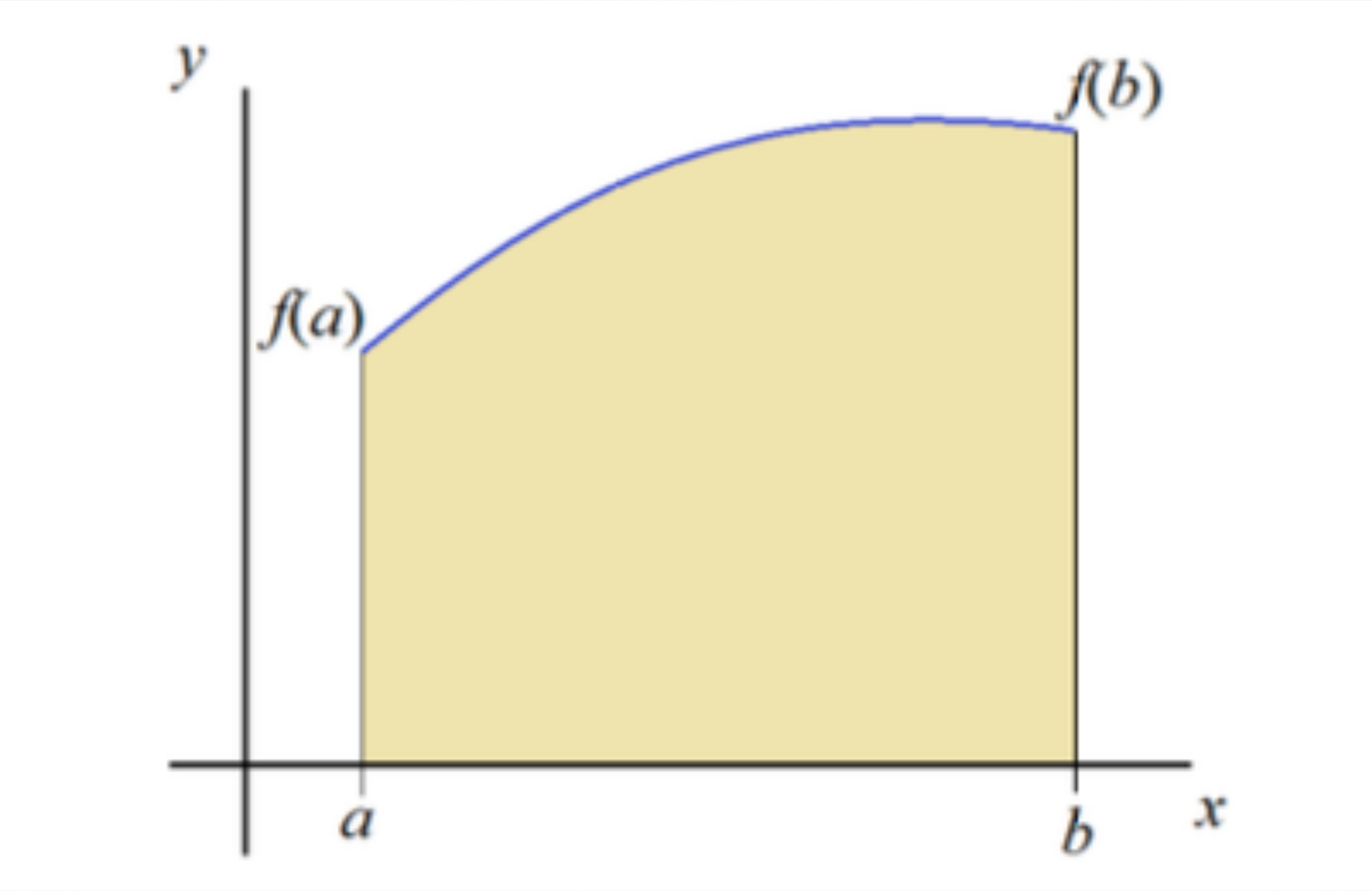
$y(0) = 1$ Initial Value of y

x	Xn
0	1
Lower	Upper

H = 0.5

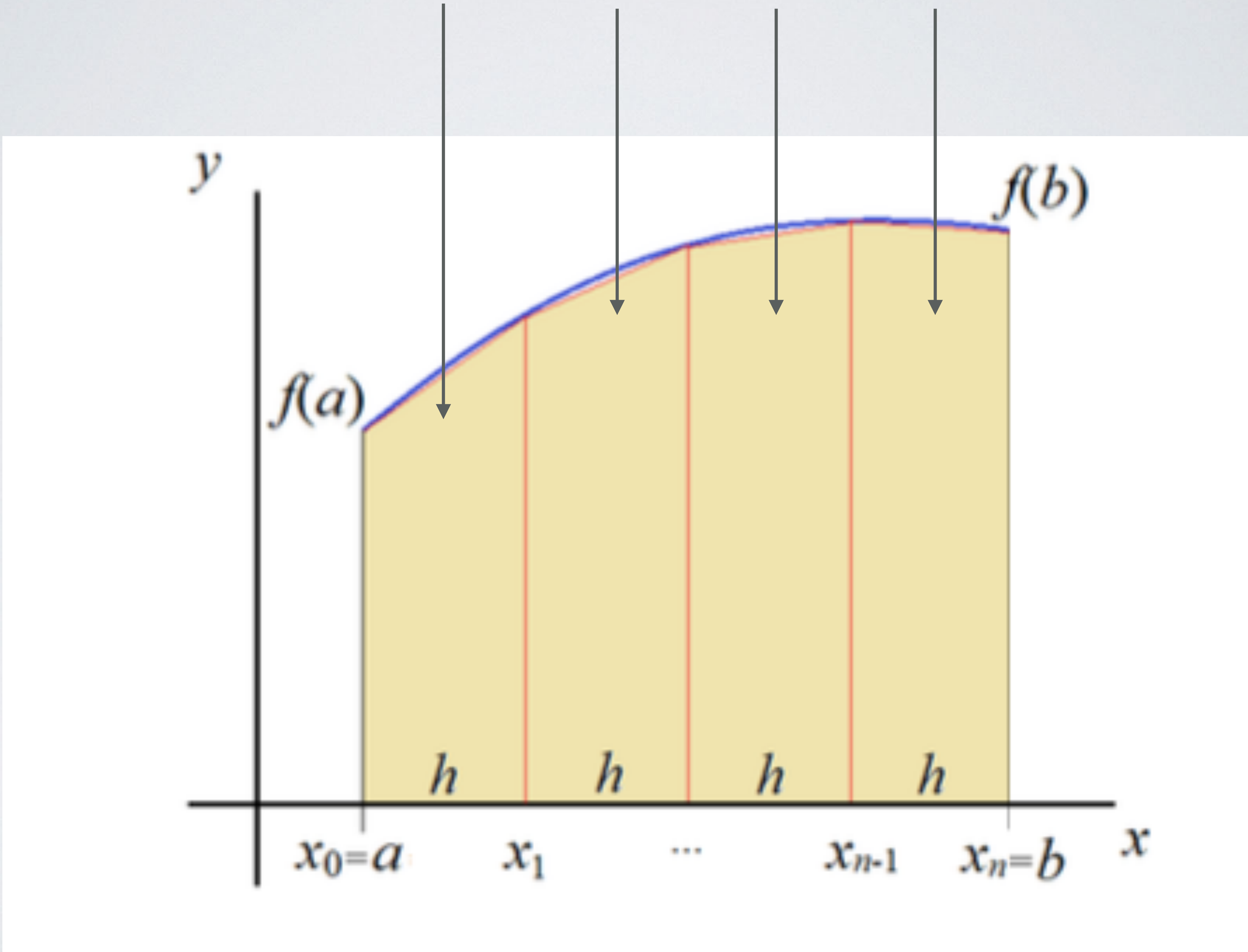
Number of Steps (n)

TRAPEZOIDAL RULE



Area of this enclosed segment

Number of Divisions



Example

Example: Find the value of the integral

$$\int_0^{\frac{\pi}{2}} x \sin x \, dx$$

Solution: The analytical integration gives 1.

Question = F

a Lower Limit

B Upper Limit

Number of Divisions (n)

Inputing the formula (S)

x = a + h

x = a + 2h

The area of the first section is

$$A = h[f(x_0) + f(x_1)]/2$$

The integral will be equal to the sum of the trapezoidal areas:

$$I = \frac{h}{2}[f(x_0) + f(x_1)] + \frac{h}{2}[f(x_1) + f(x_2)] + \cdots + \frac{h}{2}[f(x_{n-2}) + f(x_{n-1})] + \frac{h}{2}[f(x_{n-1}) + f(x_n)]$$

So,

$$I = h\left\{\frac{1}{2}[f(x_0) + f(x_n)] + f(x_1) + f(x_2) + \cdots + f(x_{n-2}) + f(x_{n-1})\right\}$$

or

$$I = h\left\{\frac{1}{2}[f(x_a) + f(x_b)] + f(x_1) + f(x_2) + \cdots + f(x_{n-2}) + f(x_{n-1})\right\}$$

Which can be programmed with in a single for-loop. Since the step size, h, is constant, the notation of x_i can be implemented in the code as $x_1 \rightarrow$ `a+h`, $x_2 \rightarrow$ `a+2h` and so on.

S