Su estructura consiste en:

1. Se crea una *Def* para contener la búsqueda.

def binary\_search(nums, target):

1. Establecer como variables, el *Begin* y el *final* de la matriz, donde el *Begin* es el primer elemento y el *final* el último elemento.

left, right = 0, len(nums) - 1

1. Se crea el Loop *While*.

while left <= right:

1. Calcular el *MID* de la lista.

mid = (left + right) // 2

1. Comparar el *MID* con el número que buscas o *Query.*

if nums[mid] == target:

1. Si el *MID* es igual al *Query*, se devuelve el *índice* del *Query.*

return mid

1. Si el *Query* es mayor que el *MID*, se actualiza el valor de *Begin* a *MID + 1.*

elif nums[mid] < target:

left = mid + 1

Para que el loop busque en >>>>>>> de la matriz

1. Si el *Query* es menor que el *MID*, se actualiza el valor de *Begin* a *MID – 1. Esto puede ser un elif o un else*

else:

right = mid – 1

Para que el loop busque en <<<<<<< de la matriz

1. Si el *Query* no está en la lista, se hace un *Return* *-1* o *0.*

return None

Ejemplo del código:

def binary\_search (nums, target):

    left, right = 0, len(nums) - 1

    while left <= right:

        mid = (left + right) // 2

        if nums[mid] == target:

            return mid

        elif nums[mid] < target:

            left = mid + 1

        else:

            right = mid - 1

    return None