

Binary Search.

```
def binary_search(s_list, x):  
    """Find the position p (0 <= p <= len(s_list) where element x  
    should be _inserted_ in list s_list, which should be sorted  
    in _increasing_ order. (Insertion might mean: _Replace_ an  
    element which is already present in s_list.)"""  
  
    # Length of the list:  
    n = len(s_list)  
  
    # Simple case:  
    if s_list[0] >= x:  
        return 0  
    if x > s_list[-1]:  
        return n  
  
    # Otherwise: s_list[0] < x <= s_list[n-1]  
    left = 0  
    right = n-1  
  
    # So we have s_list[left] < x <= s_list[right];  
    # we shall _maintain_ these inequalities during  
    # the following loop:  
    while (right - left) > 1:  
        # mid = largest INTEGER <= (right + left) / 2:  
        # Symbol "/" means "integer division"; not  
        # to be confused with "normal" division "/".  
        mid = (right + left) // 2  
        if x <= s_list[mid]:  
            right = mid  
        else:  
            left = mid  
  
    return right
```