_/_/_

* Explore algorithms for linear search, binary search, selection sort and bubble sort, and find out time complexity
* Step-count method for complexity of sum of numbers.

1) Linear search

Code	Unit Time	Frequency	Total
arr = [1, 2, 3, 4, 5, ..., n]	1	1	1
x = 4	1	1	1
For i = 0 upto ~~len(arr)~~ n do	1	n+1	n+1
if (arr[i] == x)			
print (arr[i])	1	1	1
end For	0	0	0

$$T(n) = \begin{cases} O(1), & \text{if } arr[i] == x \\ O(n), & \text{if } arr[i] \,!= x \end{cases}$$

2) Binary search

```
arr = [1, 2, 3, 4, 5}, ..., n]
x = 4
l = 0
while l <= n-1
        mid = l + (n-2)//2
        if arr[mid] == x
            return mid
        elif arr[mid] < x
            l = mid + 1
        else
            r = mid - 1
return -1
end while
```

1	$n/2$	1
2	$n/4, 3n/4$	2
3	$n/8, 3n/8, 5n/8, 7n/8$	4
4		8
⋮		⋮
x		2^{x-1}

$$T(n) = \begin{cases} O(1), & \text{if } arr[mid] == x \\ \cancel{O(\log N)} \\ O(\log_2 n), & \text{if } arr[mid] \;!= x \end{cases}$$

3) Selection sort

```
A = [7, 9, 2, 4, 10, ....]
For i = 0 upto n
    mini = i
    for j = i+1 upto n
        if A[mini] > A[j]
            mini = j
    end for
    A[i], A[mini] = {A[mini], A[i]
end For
```

$$\Rightarrow T(n) = O(n^2)$$

4) Bubble sort

```
A = [64, 34, 25, 12, 22, .....]
For i = 0 upto n
    swapped = False
    for j = 0 upto n-i-1
        if arr[j] > arr[j+1]
            arr[j], arr[j+1] = arr[j+1], arr[j]
            swapped = True
    if (swapped == False)
        break
```

$$\Rightarrow T(n) = O(n^2)$$