CITY
UNIVERSITY OF LONDON
EST 1894

STEM Digital Academy

School of Science & Technology

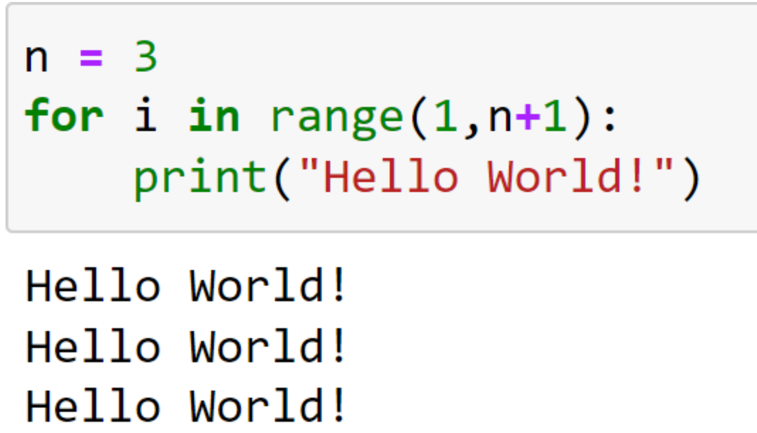
www.city.ac.uk

Introduction to Sorting Algorithms

Programming and Algorithms

Lecture by

Dr Daniil Osudin



```python
n = 3
for i in range(1,n+1):
    print("Hello World!")
```

```
Hello World!
Hello World!
Hello World!
```

What will we Cover?

- Introducing sorting algorithms
- Simple sorting algorithm
- Understanding the efficiency of the algorithm using big-O notation

Why use Sort Algorithms?

Sorting – a classic subject in computer science

Examples

- Sports league tables
- Ranking countries based on GDP
- Sorting products based on their price
- Sorting restaurants based on their review rating

Purpose of Sorting Algorithms

- Sorting algorithms – creative approach to problem solving
- Sorting algorithms are an efficient approach to sorting the elements of sequential containers
- Applications in other computing areas, such as search algorithms
  - ordered linear search and binary search needed the list to be sorted

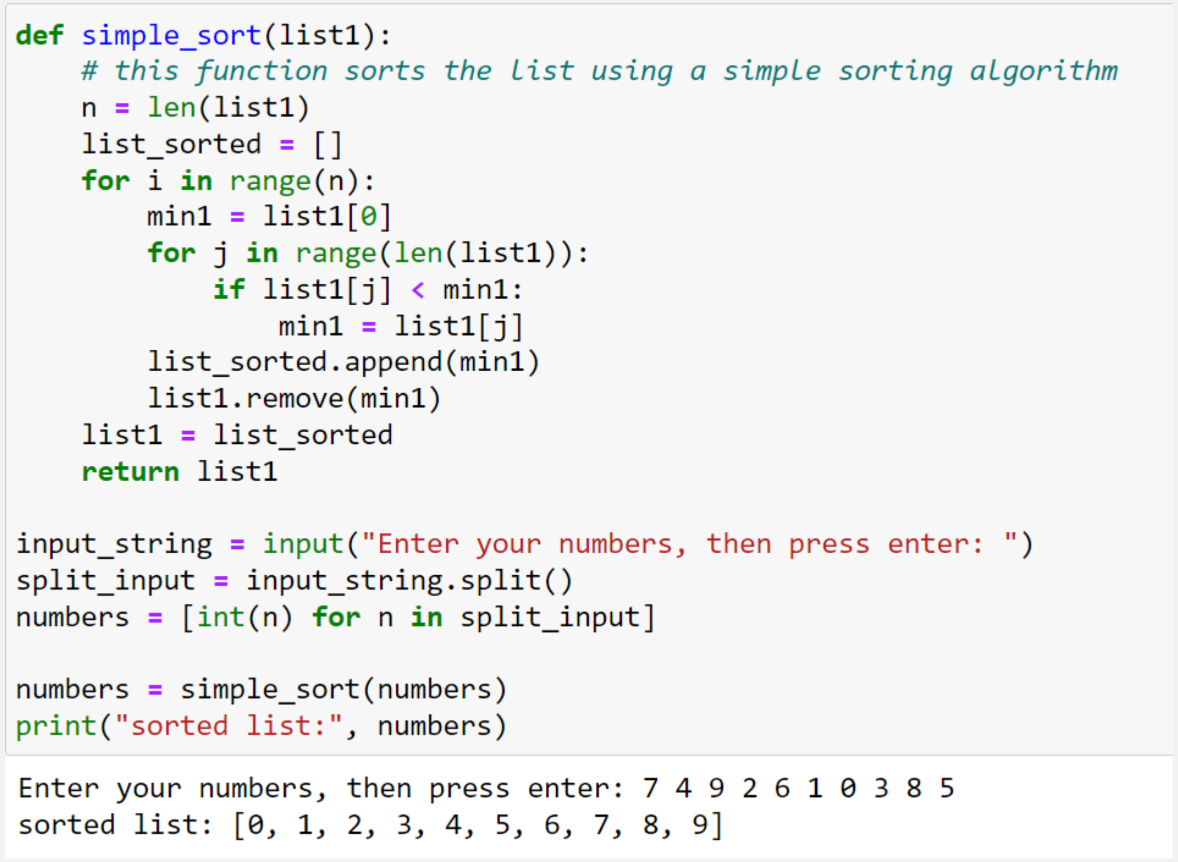
Simple Sorting Algorithm I

- The simplest way of sorting a list is to first find the smallest element in the list
- Then place it at position 0 of the new list
- Then find the second smallest element in the list and place it at position 1
- Repeat this process for all elements in the list

Simple Sorting Algorithm II

```
n = len(list)
list_sorted = []
for i in range(n)
    min = list[0]
    for j in range(len(list))
        if list[j] < min
            min = list[j]
    list_sorted.append(min)
    list.remove(min)
list = list_sorted
```

Simple Sorting Example



```
def simple_sort(list1):
    # this function sorts the list using a simple sorting algorithm
    n = len(list1)
    list_sorted = []
    for i in range(n):
        min1 = list1[0]
        for j in range(len(list1)):
            if list1[j] < min1:
                min1 = list1[j]
        list_sorted.append(min1)
        list1.remove(min1)
    list1 = list_sorted
    return list1

input_string = input("Enter your numbers, then press enter: ")
split_input = input_string.split()
numbers = [int(n) for n in split_input]

numbers = simple_sort(numbers)
print("sorted list:", numbers)
```

```
Enter your numbers, then press enter: 7 4 9 2 6 1 0 3 8 5
sorted list: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Analysis

- This algorithm will always execute the outer loop **n** times and the inner loop **n** times.
- This means that the best, average and worst-case complexities are the same.
- The big-O notation for this sorting algorithm is **$O(n^2)$**
- This algorithm is inefficient and other sorting algorithms are always preferable

Try It Yourself

Write a program in python environment that takes a string as an input and sorts in alphabetical order using the simple sorting algorithm above