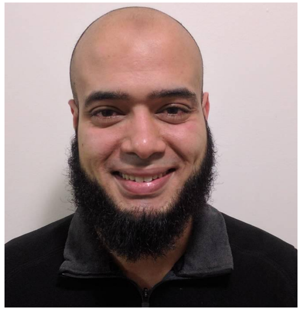
Algorithms

Algorithms: What and Why

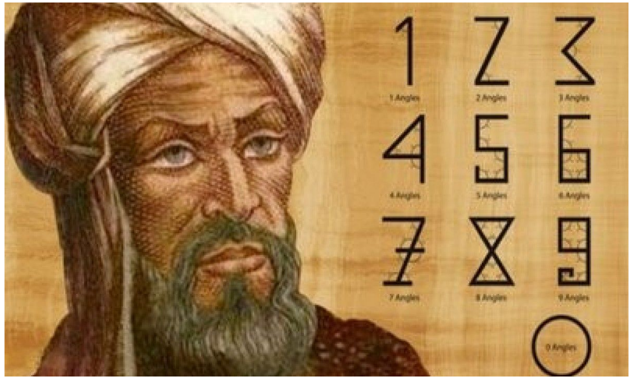
Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



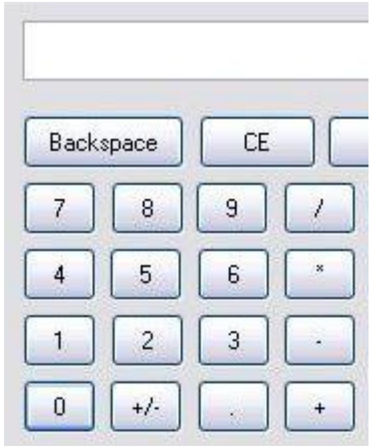
Little History: The **father** of Algorithm

- Muḥammad ibn Mūsā al-**Khwārizmī** was a Persian polymath
  - produced vastly influential works in mathematics, astronomy, and geography
  - presented the first systematic solution of linear and quadratic **equations** (algorithms)
  - considered the founder of algebra
    - The term algebra itself comes from the title of his book
  - His name (Khwārizmī) gave rise to the terms **algorithm**



A simple calculator

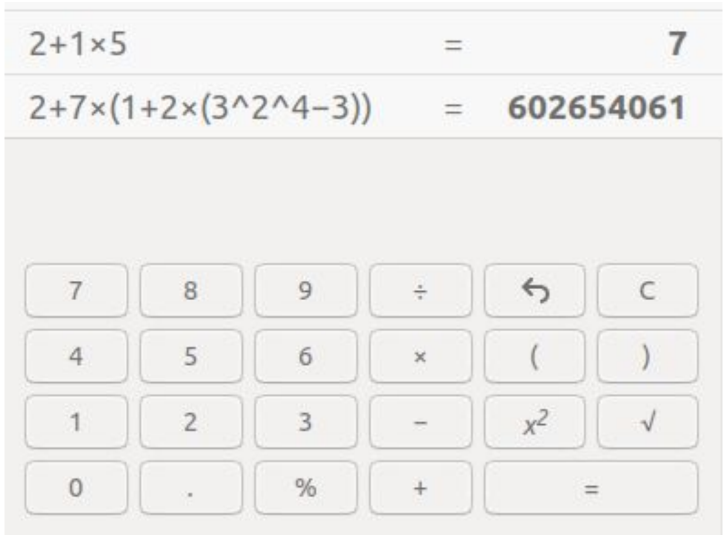
- Do you remember your first calculator program?
- It was most likely a simple one
  - Read an expression of 2 numbers: e.g. 51.7+ 12
    - Possible operations: + - * /
  - Be careful about division by zero
  - Numbers might not be separated from the operator (see example)
- Many projects, such as that one, or similar ones like building a hospital system, involve a specific skill
  From code to requirements.
- The main art is your **design and implementation** skills
  - This could be very challenging!



Img src

An advanced Calculator

- Let’s build a more complex calculator
    - The expression may have several numbers
    - Operators: + - * / and also ^ (power)
    - Parentheses: ( )
- Now, how can you compute the answer?
    - 2+1*5: mathematically, this is 7, NOT 3*5 = 15
    - What about: 2+7×(1+2×(3^2^4−3)) ?
        - 3^2^4 = 3 ^ 16 NOT 9^4
        - () must be applied first
- Hmm: you can’t just translate requirements!
    - There is a part involving unusual **thinking** style!
    - It involves more computations!



```
2+1×5                =          7
2+7×(1+2×(3^2^4−3))  =  602654061
```

Algorithmic Thinking

- To correctly solve the expression: 2+7×(1+2×(3^2^4−3))
    - We need a careful **step-by-step approach** to evaluate correctly
    - Mathematically : consider the associativity (left to right?) and precedence (* vs +)
    - Overall: it takes good deal of **thinking** to get it right
- Edsger Dijkstra is a popular Dutch computer scientist.
    - He invented: the **shunting-yard** **algorithm**
        - a **step by step** procedure that can solve this challenge
    - First, he converted this expression (we call it infix) to another structure (we call it postfix)
    - Then, evaluating the new structure is an easy task
    - Both tasks also made use of the **stack** data structure

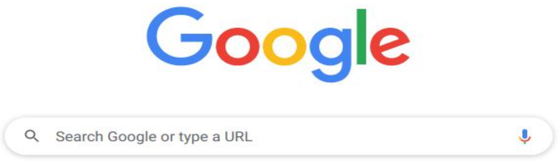
More Challenges!

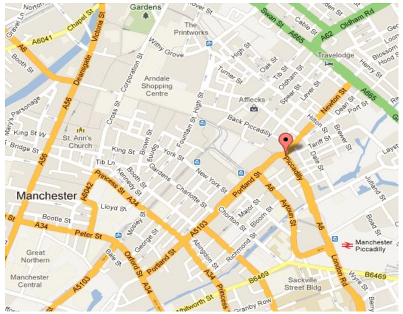
- Given 1000 integers, find how many combinations of three numbers **add up** to make 400? E.g. 100+160+140
  - We can do 3 loops to try every set of 3 numbers!
  - Hmm..1000000000 operations! Far too many computations!
  - What if we have 10^6 numbers!
- Given 10^6 integers, can you sort them from small to large?
  - Hmm...nothing **direct** in mind!
- Given airport locations and the cost of every direct flight:
  Can you find the cheapest route from, say, Cairo to New York?
  - Hmm...how can we represent the relationship between different airports?
- Facebook has ~3 billion users. Suggest to Mostafa *all new friends* such that there are exactly 5+ common friends between Mostafa and each new friend?

Performance

- In real life applications, we care about many factors
  - E.g. Usability, Security, Maintainability, Reliability, Scalability, etc
- One of the major factors is the **performance**
  - Efficiency for time and space (memory) are desirable features in all applications
  - Imagine if Facebook was a slow application! How can it be that efficient with such a huge number of users?
- This is where the **algorithms** field play a crucial role!
  - But there are many other factors. *Distributed systems* are the major key.
  - Find an efficient computational solution. The criteria? Time and space constraints

High performance real life apps





Algorithms

- **Algorithm**: A sequence of steps that transforms *some input to an output* to solve a **computational** problem.
    - Such as Decision, Search, Counting and Optimization problems
    - Tip: *Any procedure is an algorithm, but in CS we typically mean computational problems*
- Nature of Computational problem
    - Includes many **computations**
    - A direct requirement translation is not usually possible/doable
    - Or doable, but the direct idea is too slow or takes too much memory

Algorithms: problem examples

- Given an array of numbers, **sort** it from small to large
  - Input: [10, 1, 5, 2, 6, 0]
  - Output: [0, 1, 2, 5, 6, 10]
- Given a **sorted** array, return the index of a given number
  - Input: [10, 20, **70**, 101], target = 70
  - Output: index = 2
  - We can make a simple loop to **search** for the number in the array
  - However, there is a faster way to search for a number in a **sorted** array
    - It is called **binary search**.
    - In a book of 2000 pages, do you search **page by page** to find page #705? No
- Print the first 1000 prime numbers
  - No input here. However, each algorithm must have some output!

Algorithms Fields

- Sort & Search algorithms
- Graph algorithms (E.g. shortest path problem)
- Dynamic Programming & Greedy algorithms
- String algorithms (E.g. used in search engines to do matching)
- Game theory & Numerical algorithms
- Number-theoretic algorithms
- Combinatorial algorithms
- Computational geometric algorithms
- Note: ***We can’t solve every problem efficiently***

Algorithms Analysis

- Given an algorithm, we need to:
  - 1) Prove its **Correctness**
  - 2) **Measure time** efficiency
  - 3) **Measure space** (memory) efficiency
- Steps 2 and 3 allow us to **compare** algorithms
  - Fall in a complementary area: Computational **Complexity**
- An example of sorting algorithms: **time** perspective
  - **Selection sort algorithm:** orders N numbers using N*N operations
    - E.g. if N is 1000, it takes around ~10^6 operations
  - **Quicksort algorithm:** orders N numbers using N * $\log_2$N operations
    - E.g. if N is 1000, it takes around ~10^4 operations
    - Then Quicksort is a **faster** algorithm

Other types of Algorithms

- Example: **Machine Learning** (ML) algorithms
  - Given an email, how can we know if it is a spam?
    - Writing a huge amount of if/else conditions? No. Use ML
  - Given an image, how can we identify the people inside it?
  - If we can prepare a large dataset with input (e.g. email) to output (is spam?), we can **learn from the data**.
- Example: **Genetic** Algorithms
  - a heuristic search algorithm used to solve search and optimization problems
- In the industry, we should use the right tools for a problem. Without studying the different fields, you will be puzzled
  - Don’t be an ignorant CS graduate!

“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”