Sharif University of Technology
Department of Computer Engineering

Fundamentals of Programming

Python Language

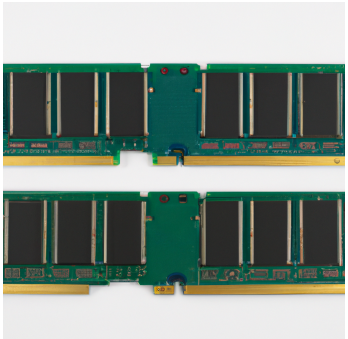
Arman Malekzadeh
PhD Candidate in Artificial Intelligence

Table of contents

Fundamental Concepts

Memory Unit

- The memory unit in a computer is known as Random Access Memory (RAM).
- This component plays a crucial role in your computer's operation.
- It is responsible for temporarily storing or "remembering" information that the computer's processor uses in real-time.



Memory Unit

- The more RAM a computer has, the more information it can process simultaneously, leading to smoother multitasking and faster load times for software and applications.
- However, it's important to note that RAM is volatile memory, meaning that when the computer is turned off, all information stored in RAM disappears.
- It's like a workspace for the computer - once you're done with your work and leave (or in this case, shut down), the workspace is cleared out.

Secondary Storage Unit

Secondary Storage Unit

- The secondary storage unit in a computer is typically a hard disk drive (HDD) or a solid-state drive (SSD).
- Its primary responsibility is to store data and information on a long-term basis, even when the computer is switched off or not in use.
- Unlike the primary storage, which is temporary and volatile, the secondary storage unit provides permanent and non-volatile storage.
- It holds the operating system, software applications, files, and personal data of the user.
- In essence, it's like a library of resources that your computer can access whenever needed.

Algorithm

Definition

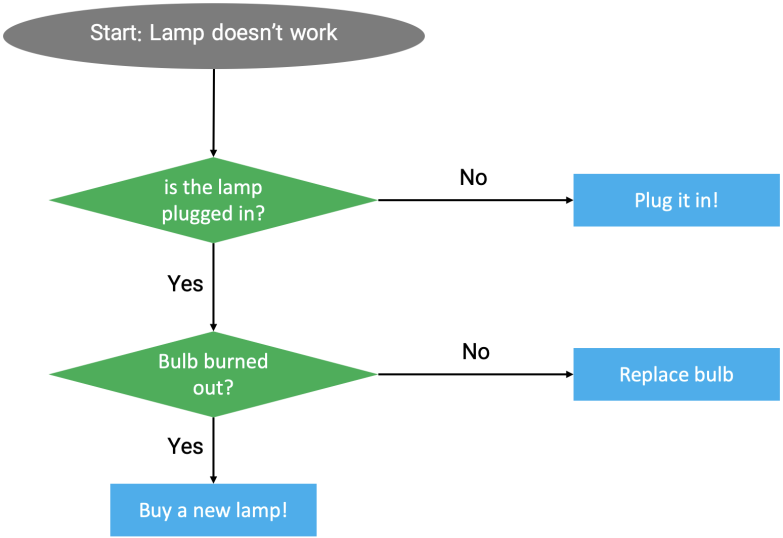
An algorithm is a finite sequence of instructions for performing a task.

- Finiteness: Termination after a finite number of steps
- Definiteness: Steps should not be vague or open to interpretation!
- Input: From zero to multiple inputs are allowed.
- Output: At least one output is needed! It should be somehow calculated from the inputs.
- Effectiveness: All directives should be simple enough in theory for an individual to execute using just a pen and paper. It's crucial that each task is not only explicit, but also practically achievable.
- Generality: The algorithm should work for all inputs of the problem, not just one!

Algorithm: Examples

- Cake Recipe
- Face detection (a type of classification algorithm)
- Long Divisions
- Algorithmic Trading (Stock Market)
- PageRank
- Brute Force for Finding a Password: Try every possibility until you get the result!
- Greedy Mountain Climbing

Flowchart: Example



Bonus Questions

Bonus Questions

- Does every procedure refer to an algorithm?

Bonus Questions

- Does every procedure refer to an algorithm?

- An example of a greedy algorithm in daily life

Program vs. Algorithm

Definition

An algorithm is a finite sequence of instructions for performing a task.

Algorithm: Finding the maximum element of a vector V

```
1: procedure FindMaximum(V)
2:     max ← V_0
3:     for v ∈ V do
4:         if v > max then
5:             max ← v
6:         end if
7:     end for
8:     return max
9: end procedure
```

Program vs. Algorithm

Definition

A program is essentially the representation of an algorithm, expressed through a specific programming language.

Python Code:

```
def find_maximum(arr):
    max_element = arr[0]
    for element in arr:
        if element > max_element:
            max_element = element
    return max_element
```

Python: History

- Python was developed in the late 1980s by Guido van Rossum.
- According to Van Rossum, he wanted to write a programming language that was accessible to everyone. Van Rossum's goals for Python included making an easy and intuitive language that was just as powerful as its more complicated competitors.
- He was engrossed in the published scripts of the 1970s BBC comedy series, "Monty Python's Flying Circus", while he initiated the development of Python. Seeking a name that was succinct, distinctive, and held a touch of mystique, he chose to christen the language as Python.

Figure: Guido Van Rossum

References

References I

[1] B Downey, A. (2015). Think Python: How to Think Like a Computer Scientist-2nd Edition.

[2] Deitel, H. M., Deitel, P. J. (2004). C: How to program. Pearson Educación.

Sharif University of Technology
Department of Computer Engineering

Arman Malekzadeh

Fundamentals of Programming

Python Language