Introduction to Automata Theory

What is Automata Theory?

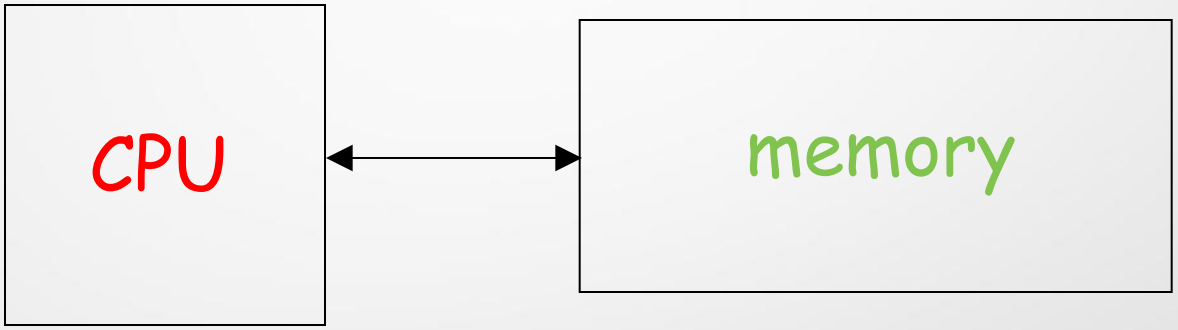
- A branch of **theoretical computer science**.
- Studies **abstract machines** (automata) and **formal languages**.
- Provides the **mathematical foundation** of computation.

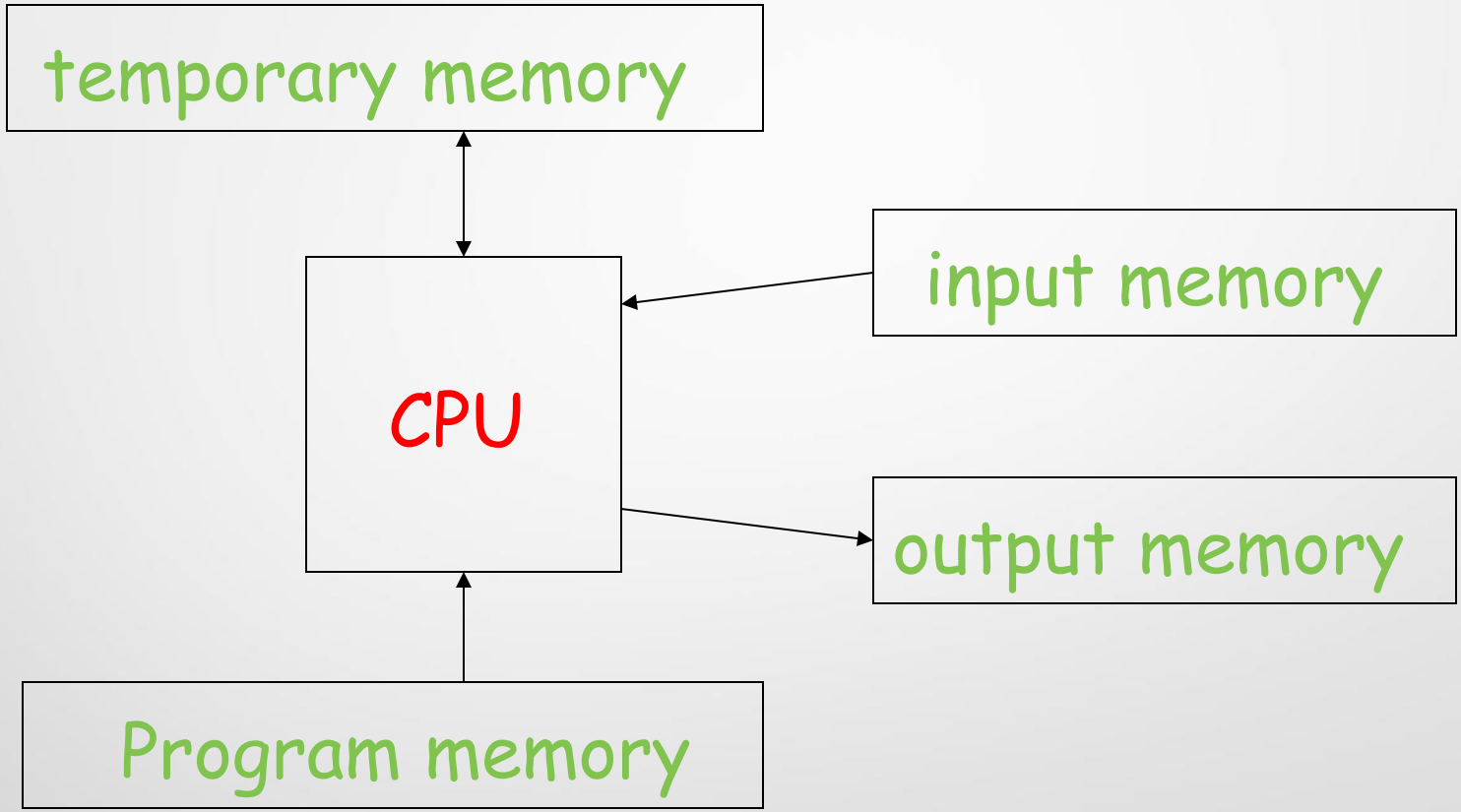
Why Study Automata Theory?

- Understand **what can be computed** and **what cannot**.
- Basis for **compiler design** and **programming languages**.
- Applications in **text processing, AI, and software verification**.

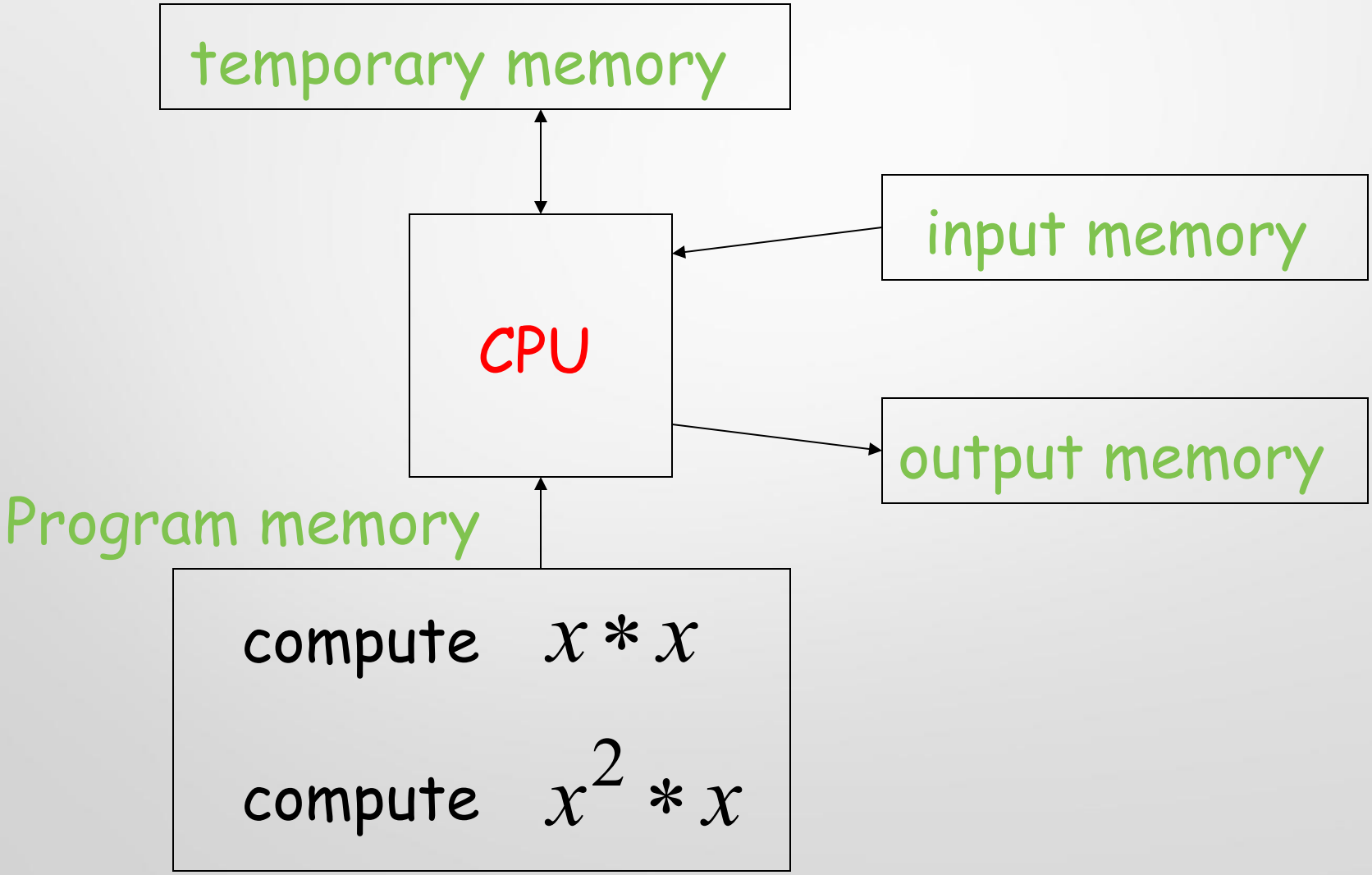
Models of Computation

Computation: Computation is a general term for any type of **information processing**

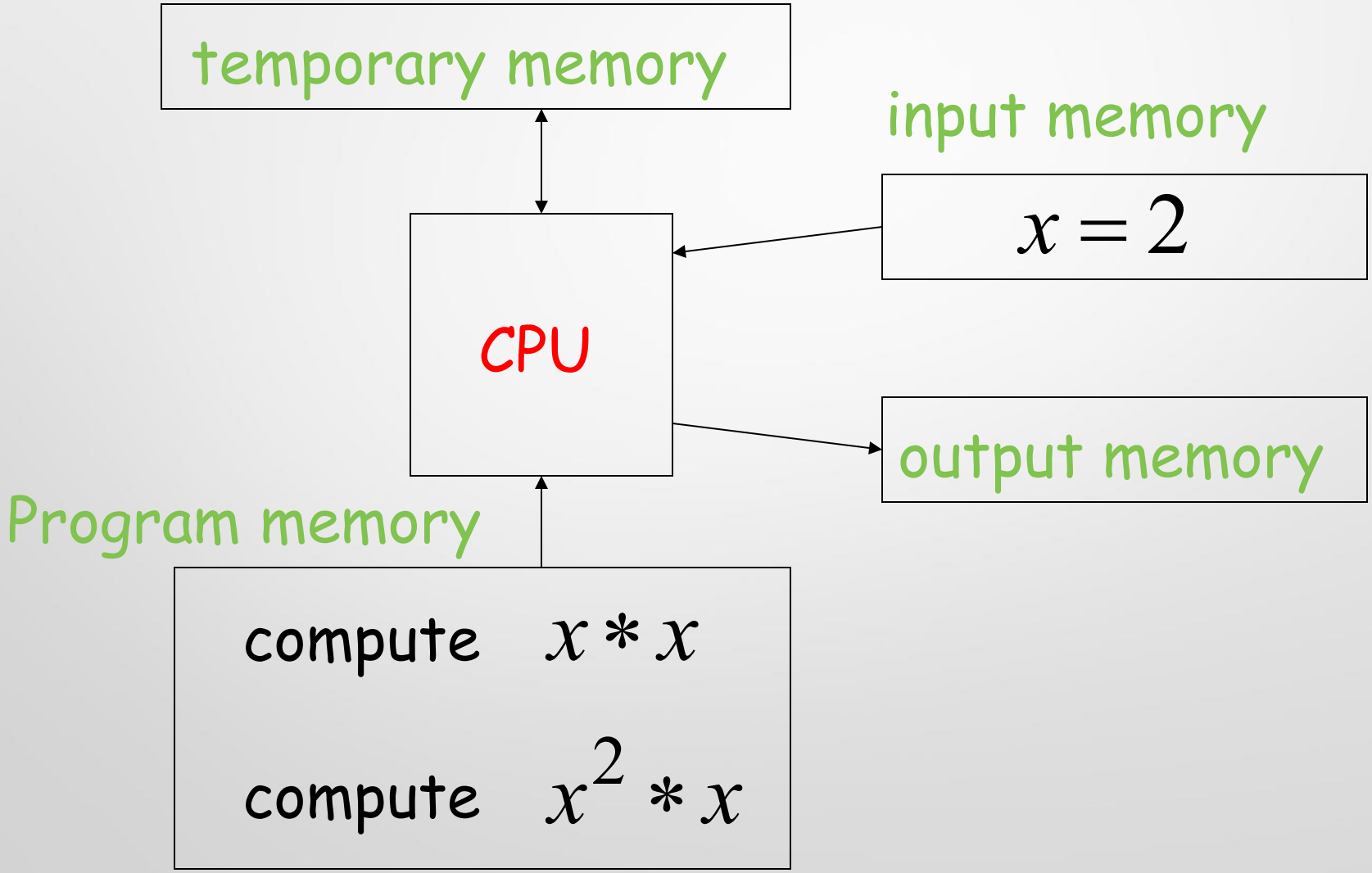



temporary memory
input memory
CPU
output memory
Program memory

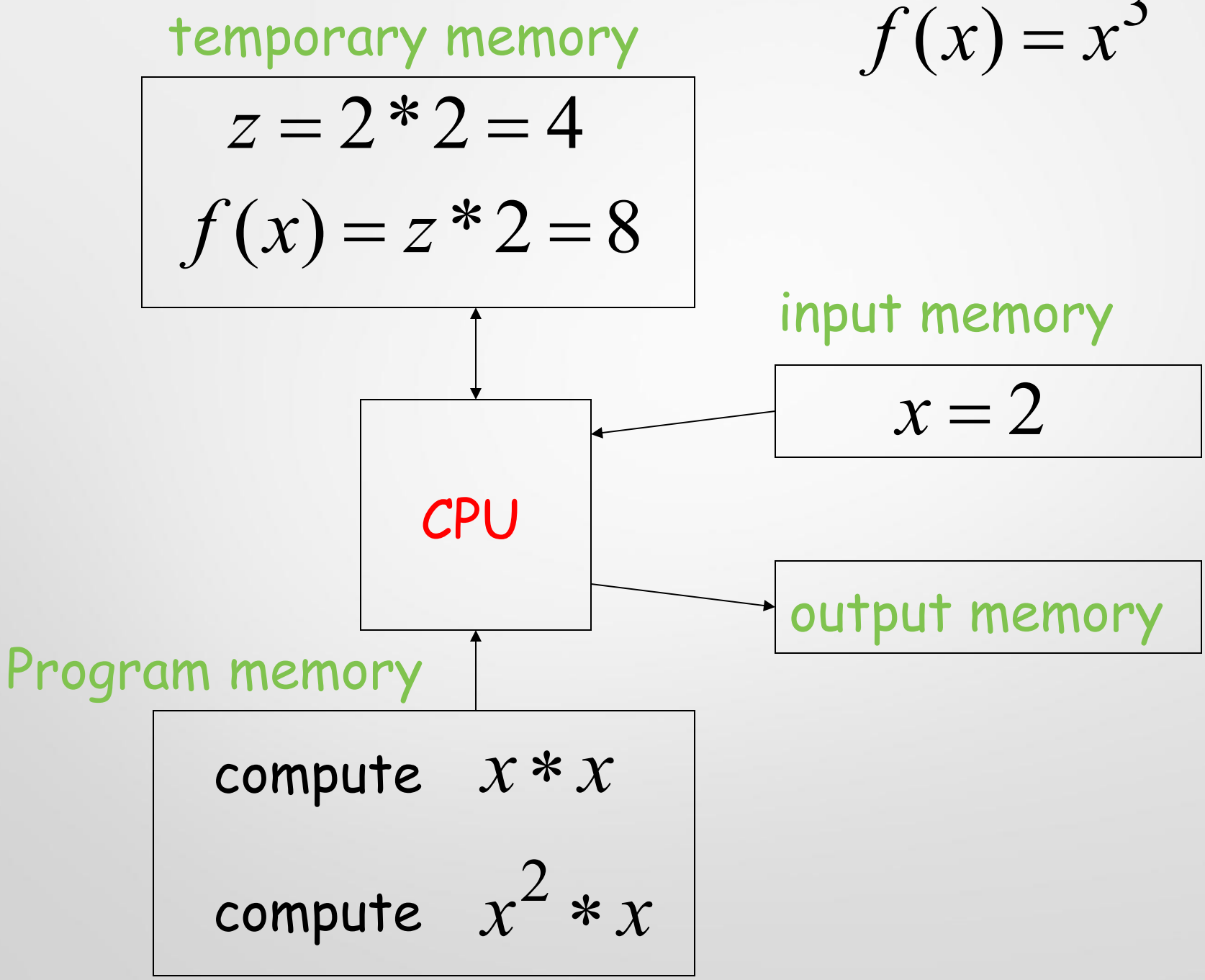
Example: $f(x) = x^3$



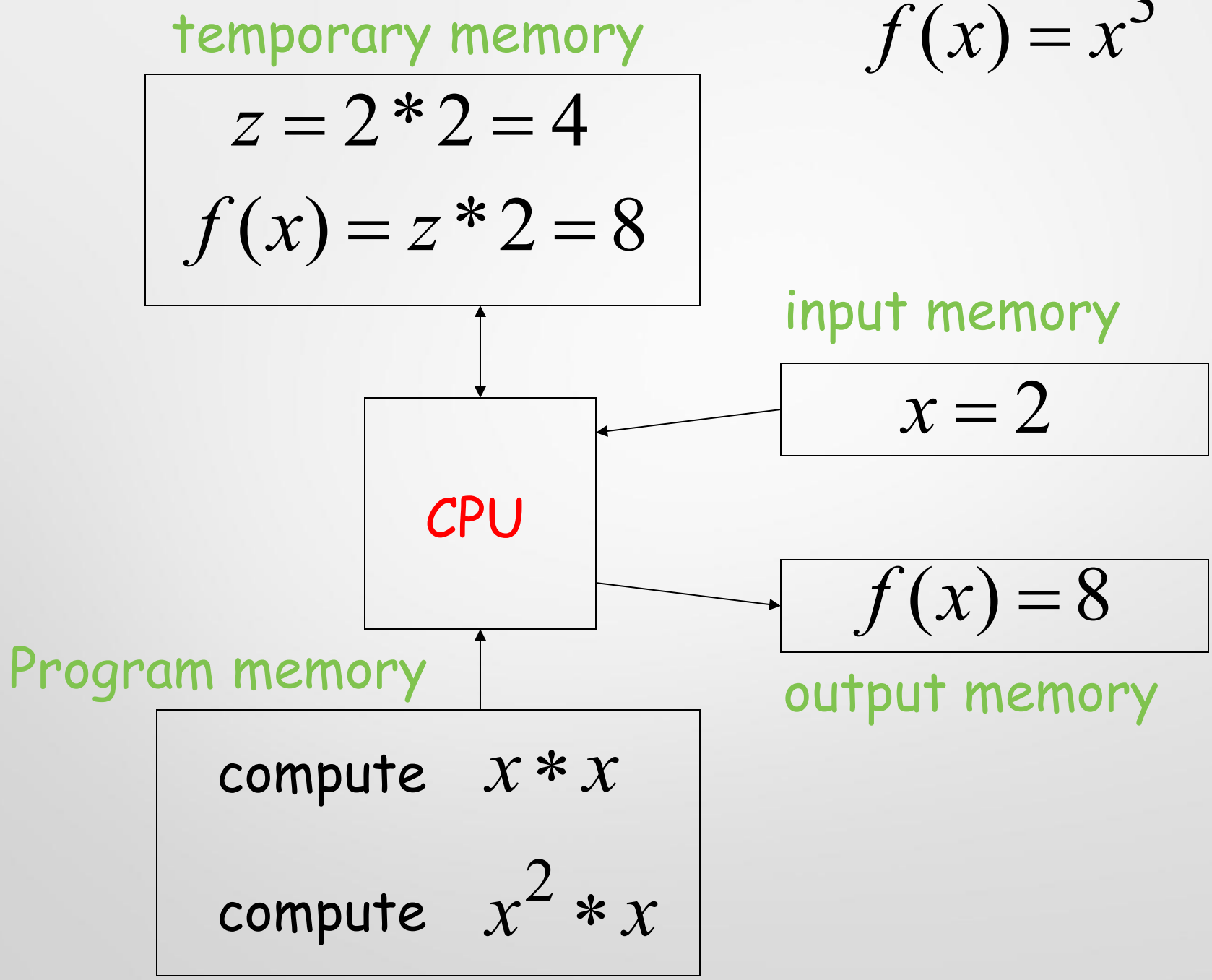
$$f(x) = x^3$$


temporary memory
input memory
$x = 2$
CPU
output memory
Program memory
compute $x * x$
compute $x^2 * x$

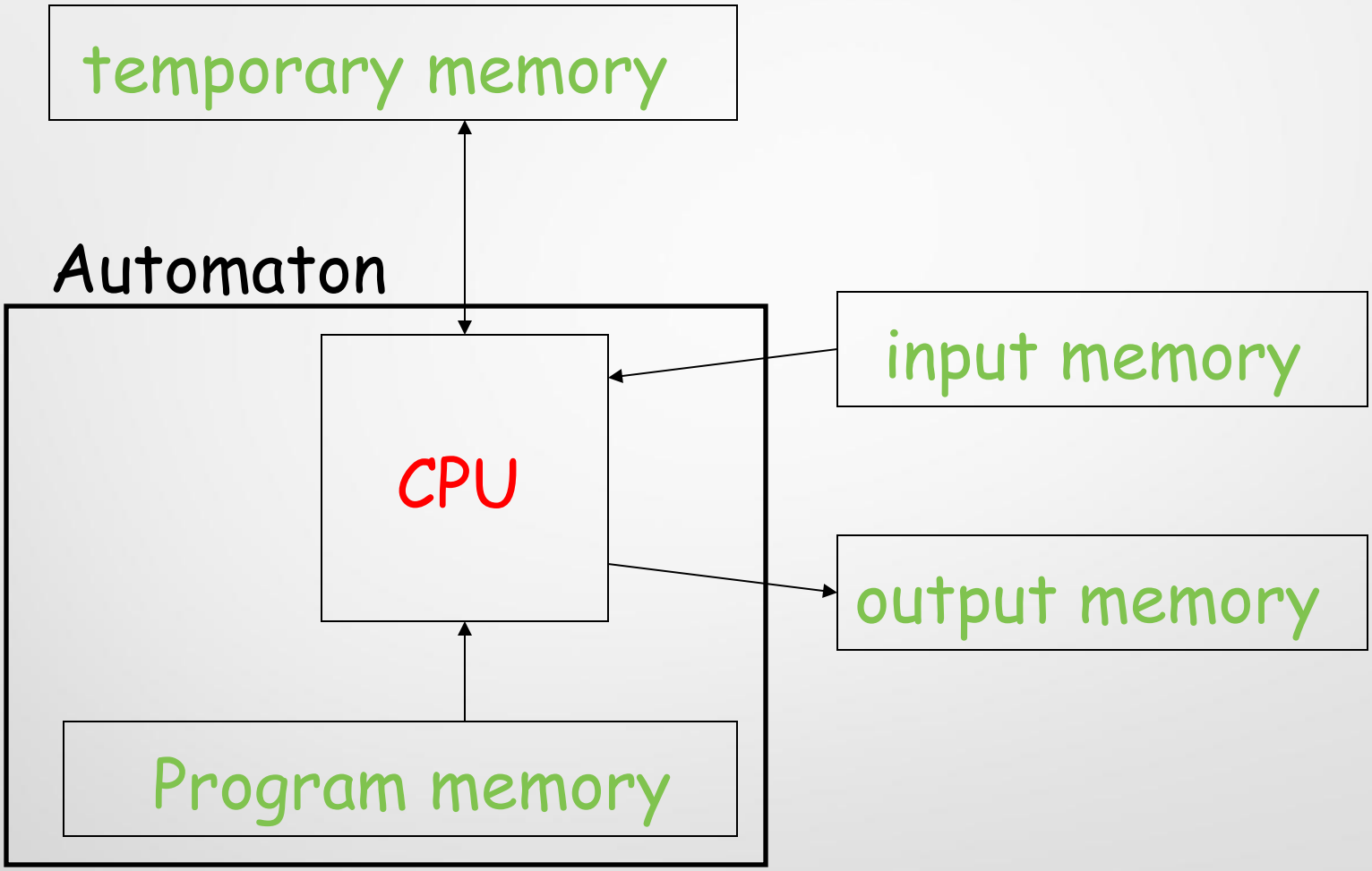
$$f(x) = x^3$$


temporary memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
input memory
$x = 2$
CPU
output memory
Program memory
compute $x * x$
compute $x^2 * x$

$f(x) = x^3$


temporary memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
input memory
$x = 2$
CPU
$f(x) = 8$
output memory
Program memory
compute $x * x$
compute $x^2 * x$

Automaton

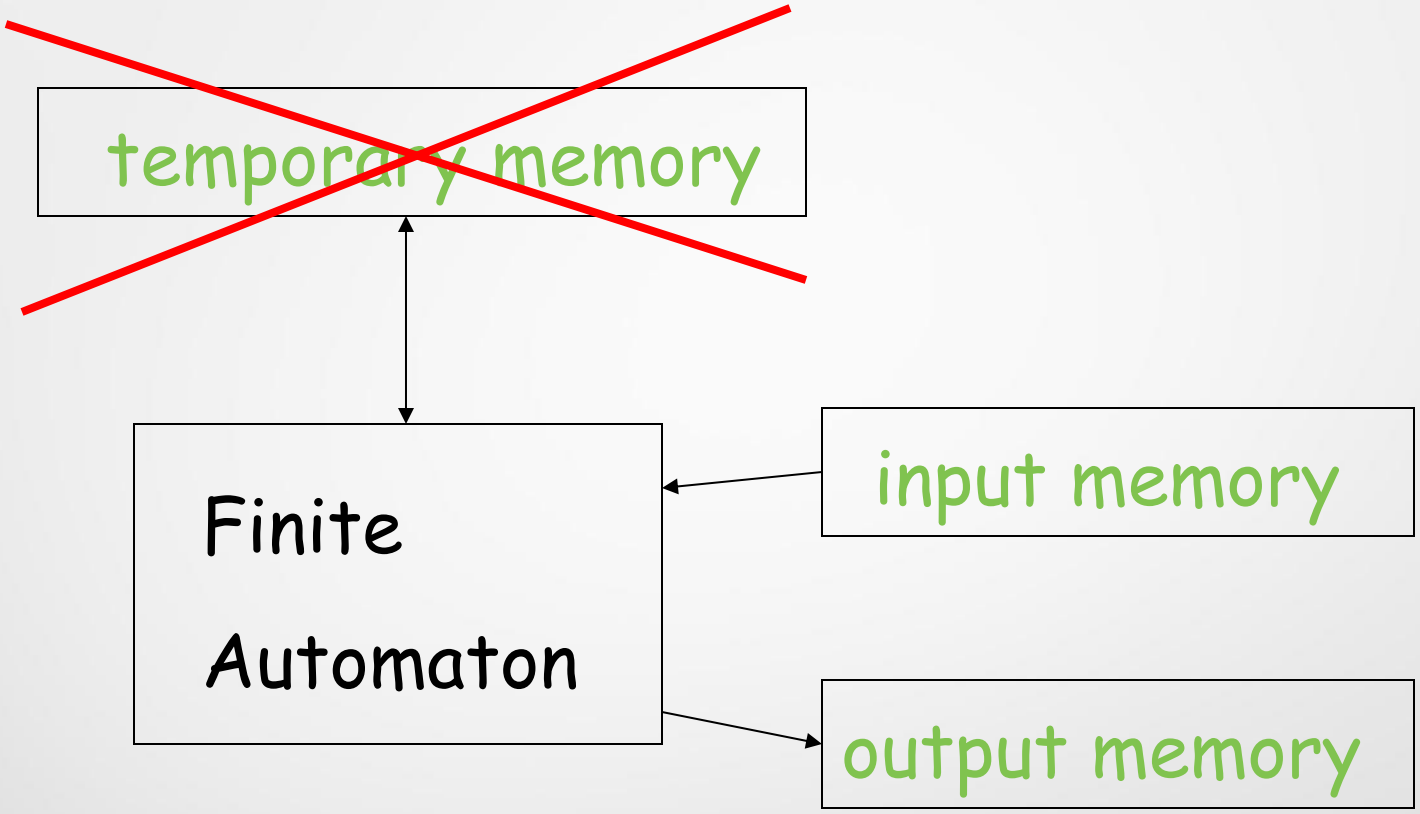

temporary memory
Automaton
input memory
CPU
output memory
Program memory

Different Kinds of Automata

Automata are distinguished by the temporary memory

- **Finite Automata:** no temporary memory
- **Pushdown Automata:** stack
- **Turing Machines:** random access memory

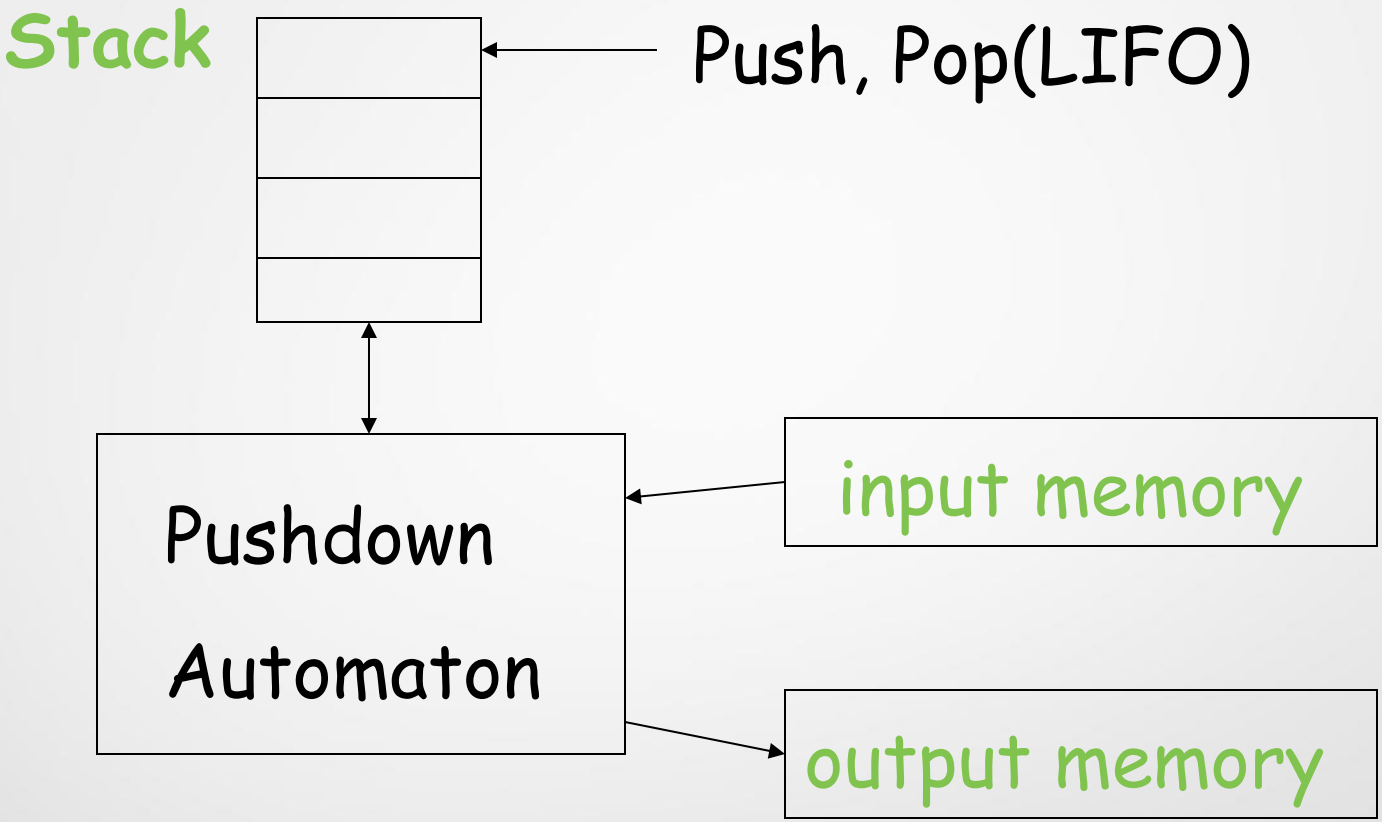
Finite Automaton



Example: Vending Machines

(small computing power)

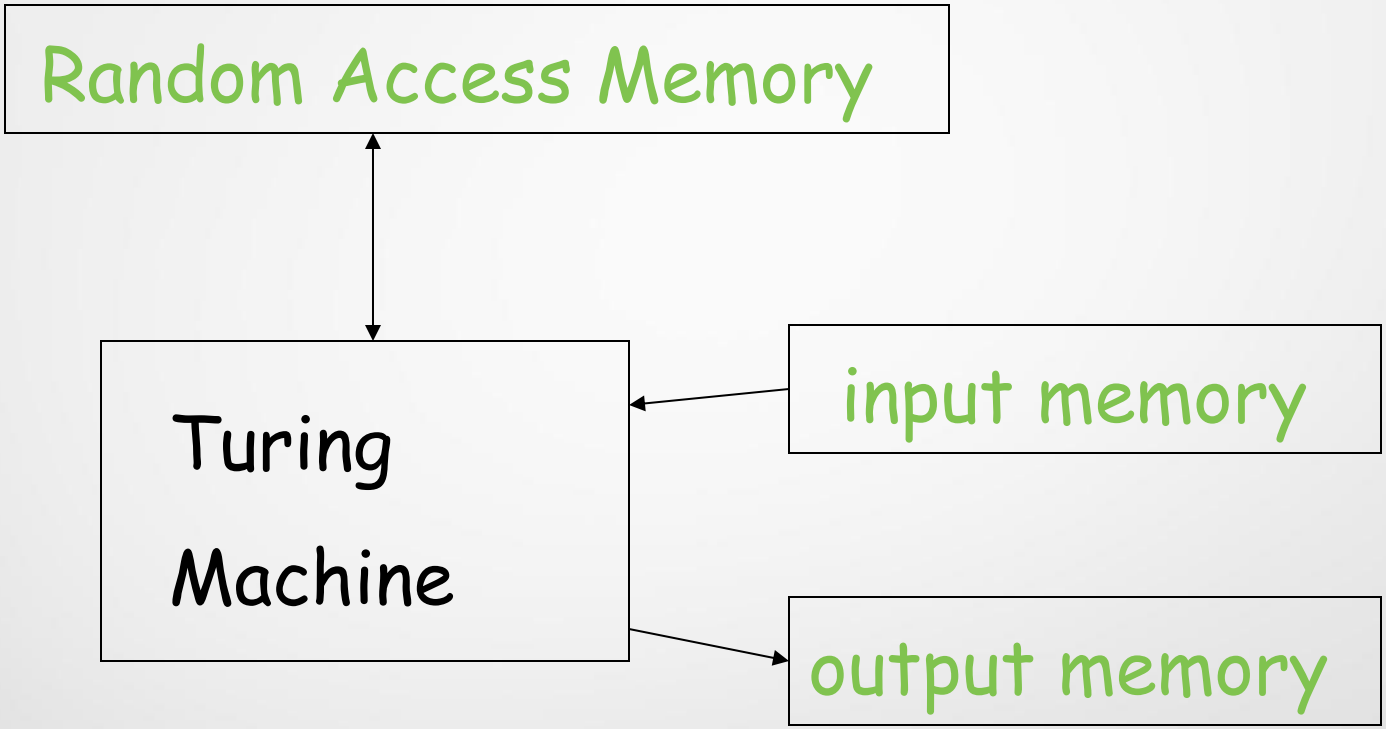
Pushdown Automaton



Example: Compilers for Programming Languages

(medium computing power)

Turing Machine



Examples: Any Algorithm

(highest computing power)

Power of Automata

Less power ⟶ More power

Solve more
computational problems