# Stack

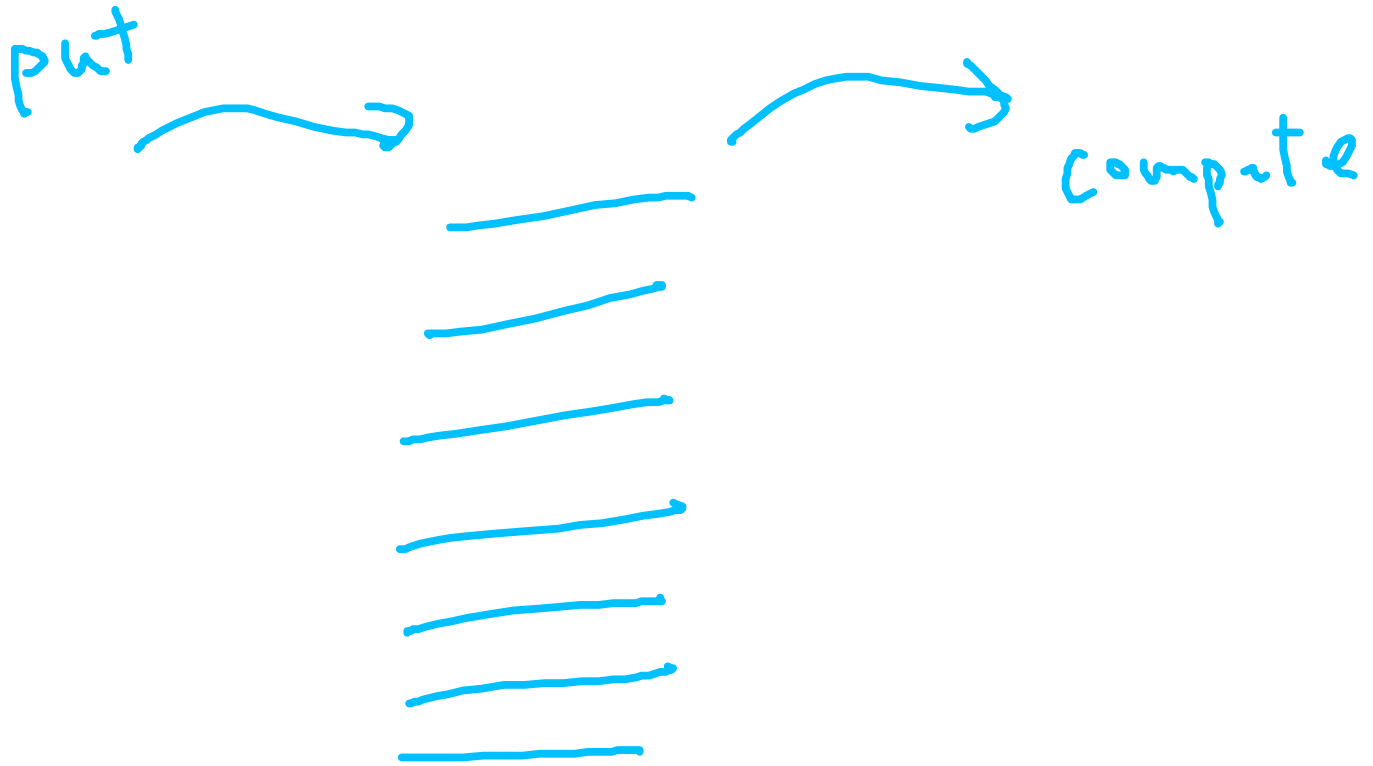
Hsuan-Tien Lin

Dept. of CSIE, NTU

March 16, 2021

# Intuition

# Stack




mimic: “pile of documents” on your desk

# Stack: Last-In-First-Out (LIFO)

## Stack

(constant-time) operations:

- `insertTop(data)`, often called `push(data)`
- `removeTop()`, often called `pop()`
- `getTop()`, often called `peek()`

—LIFO: 擠電梯, 洗盤子

very restricted data structure, but important for computers
—will discuss some cases later

# A Simple Application: Parentheses Balancing

- in C, the following characters show up in pairs: (), [], {}, ""

```
good: {xxx(xxxxxx)xxxxx"xxxx"x}
bad:  {xxx(xxxxxx}xxxxx"xxxx"x}
```

- the LISP programming language

```
(append (pow (* (+ 3 5) 2) 4) 3)
```

how can we check parentheses balancing?

# Stack Solution to Parentheses Balancing

## inner-most parentheses pair $\Longrightarrow$ top-most plate

'(': 堆盤子上去 ; ')': 拿盤子下來

## Parentheses Balancing Algorithm

**for** each *c* in the input **do**
  **if** *c* is a left character
    push *c* to the stack
  **else if** *c* is a right character
    pop *d* from the stack and check if match
  **end if**
**end for**

many more sophisticated use in compiler design (will see some)

# System Stack

- recall: function call $\Leftrightarrow$ 拿新的草稿紙來算
- old (original) scrap paper: temporarily not used, 可以壓在下面

**System Stack: 一疊草稿紙, each paper (stack frame) contains**

- return address: where to return to the previous scrap paper
- local variables (including parameters): to be used for calculating within this function
- previous frame pointer: to be used when escaping from this function

some related issues: security attack?