

程序设计语言与方法(C语言)

第一章 概述

教其他人做一件事

- 选一件事，如“微信或QQ加好友”、“扫码支付”等
 - 该让他如何做呢？
 - 如何告诉他？
 - 他该怎么做呢？
 - 如果你也不知道怎么做呢？

程序

- ❖ 程序是什么？
 - ❖ 蓝本、计划、解决方案：为解决某一问题，而需要进行的一系列操作（处理、步骤、……）
 - ❖ 为解决问题而存在
- ❖ 谁来执行程序？（程序的执行主体）
- ❖ 为什么会执行程序？（程序的作用）

设计程序

- ❖ 了解问题（目的）
- ❖ 分析问题（做什么，结果）
- ❖ 找寻解决问题的方法（如何做）
- ❖ 制定步骤序列（怎么做）
- ❖ 验证程序（正确性保证）

编制程序

- ❖ 为什么编制程序?
 - ❖ 编制程序的目的是什么? 指导、规范执行者的行为
- ❖ 为谁编制程序?
 - ❖ 执行主体能做什么?
- ❖ 如何书写程序?
 - ❖ 执行主体的阅读和理解能力如何?

一个好的程序编制者

- ❖ 让程序的执行主体确切明白你的意思
 - ❖ 直白、明了的每个步骤（详细程度）
 - ❖ 严格的步骤序列（差错控制）
 - ❖ 明确的操作对象和结果（达到目的）

如何成为一个好的程序编制者

- 明白问题

重点是解决问题的思路

- 分析问题

找到解决问题的方法

编制执行者可用的程序

- 设计流程

团队协同！！

- 编制程序

- 测试程序（不一定会亲自去做一遍！）

计算1到10的累加和

- ❖ 如何求解（寻求解决方案）
- ❖ 编制解决方案（描述求解过程）

你懂的，我懂吗？！

请让我看得懂你的描述，我才能完成任务哦～

❖ 准备

- ❖ 一些可擦除的空白卡片，10张数字卡片（从1到10）

方法1

❖ 执行主体的能力?

❖ 我只会两个整数的加法

❖ 合适的方法

❖ 只用加法

- $1+2+3+4+5+6+7+8+9+10$

- $3+3+4+5+6+7+8+9+10$

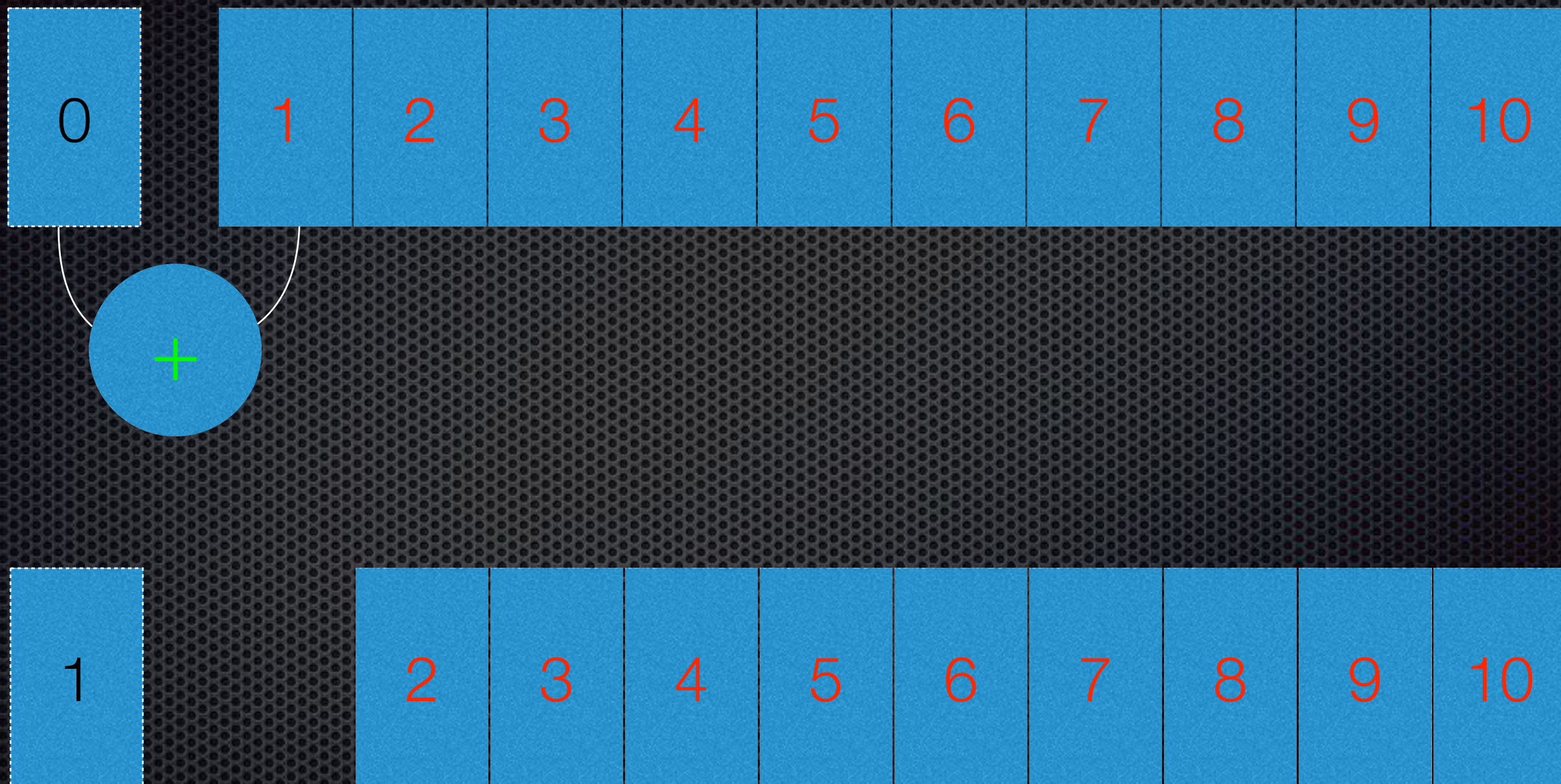
- $6+4+5+6+7+8+9+10$

-

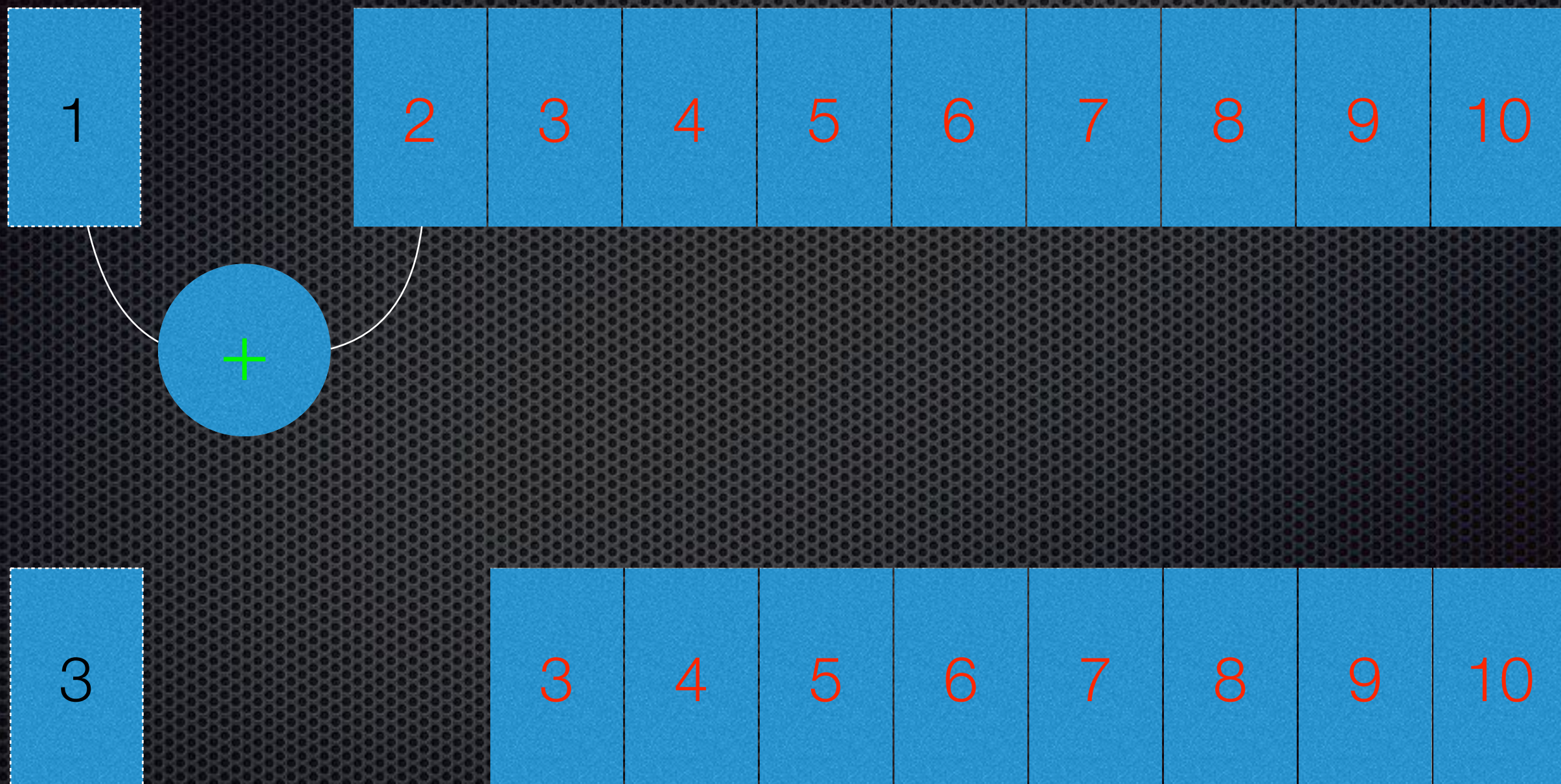
- $45+10$

- 55

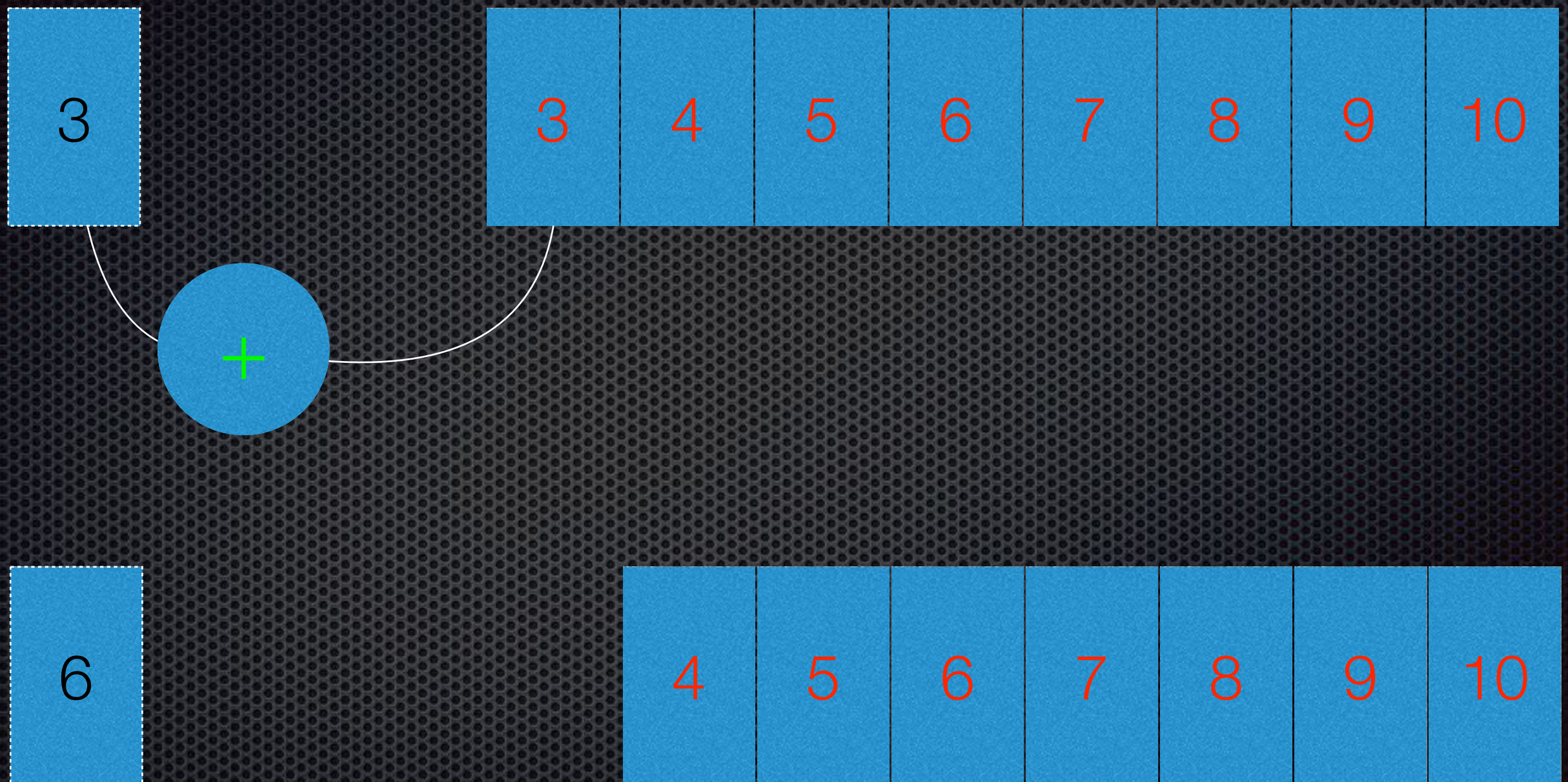
整理方法1



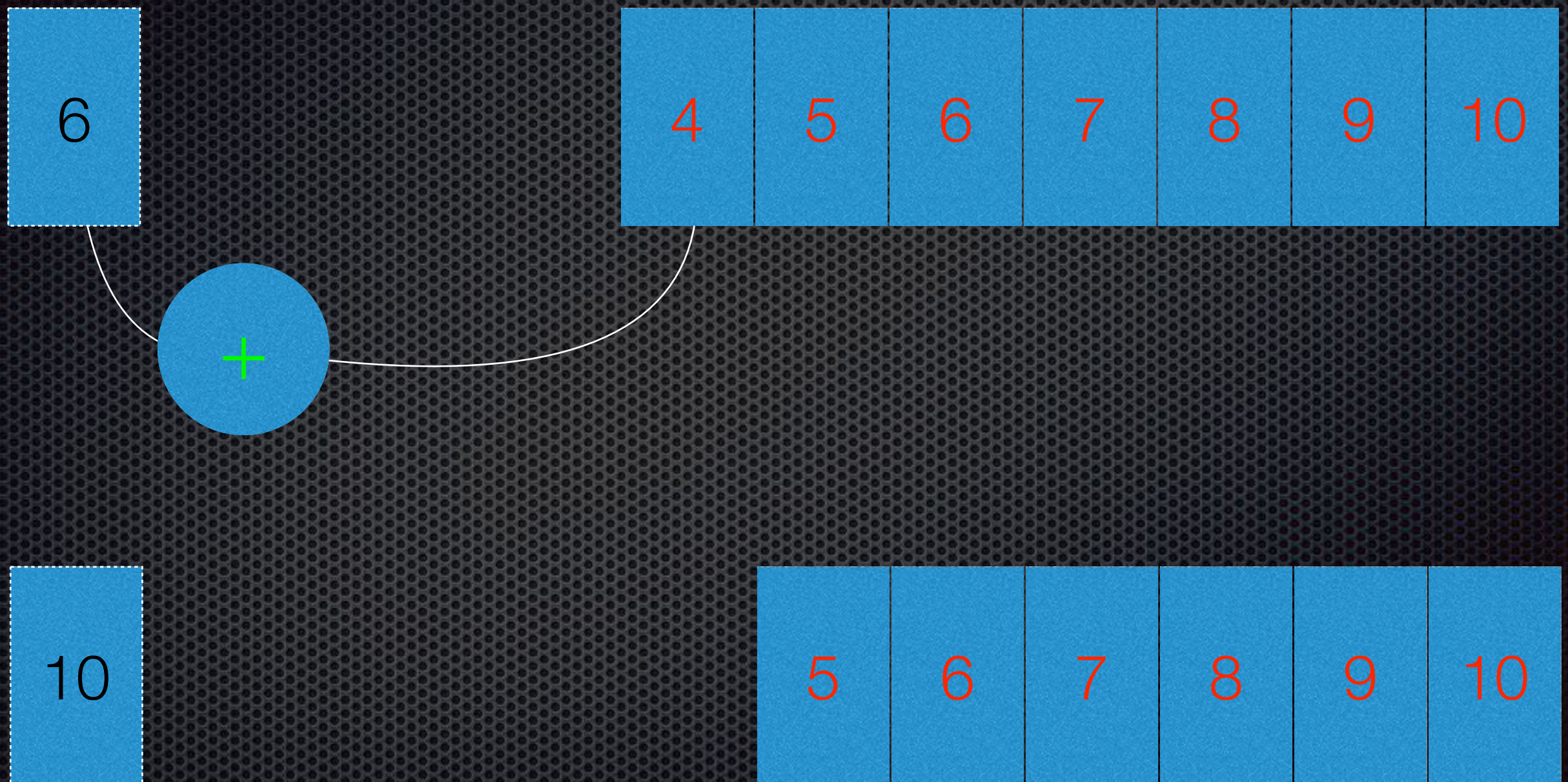
整理方法1 (续)



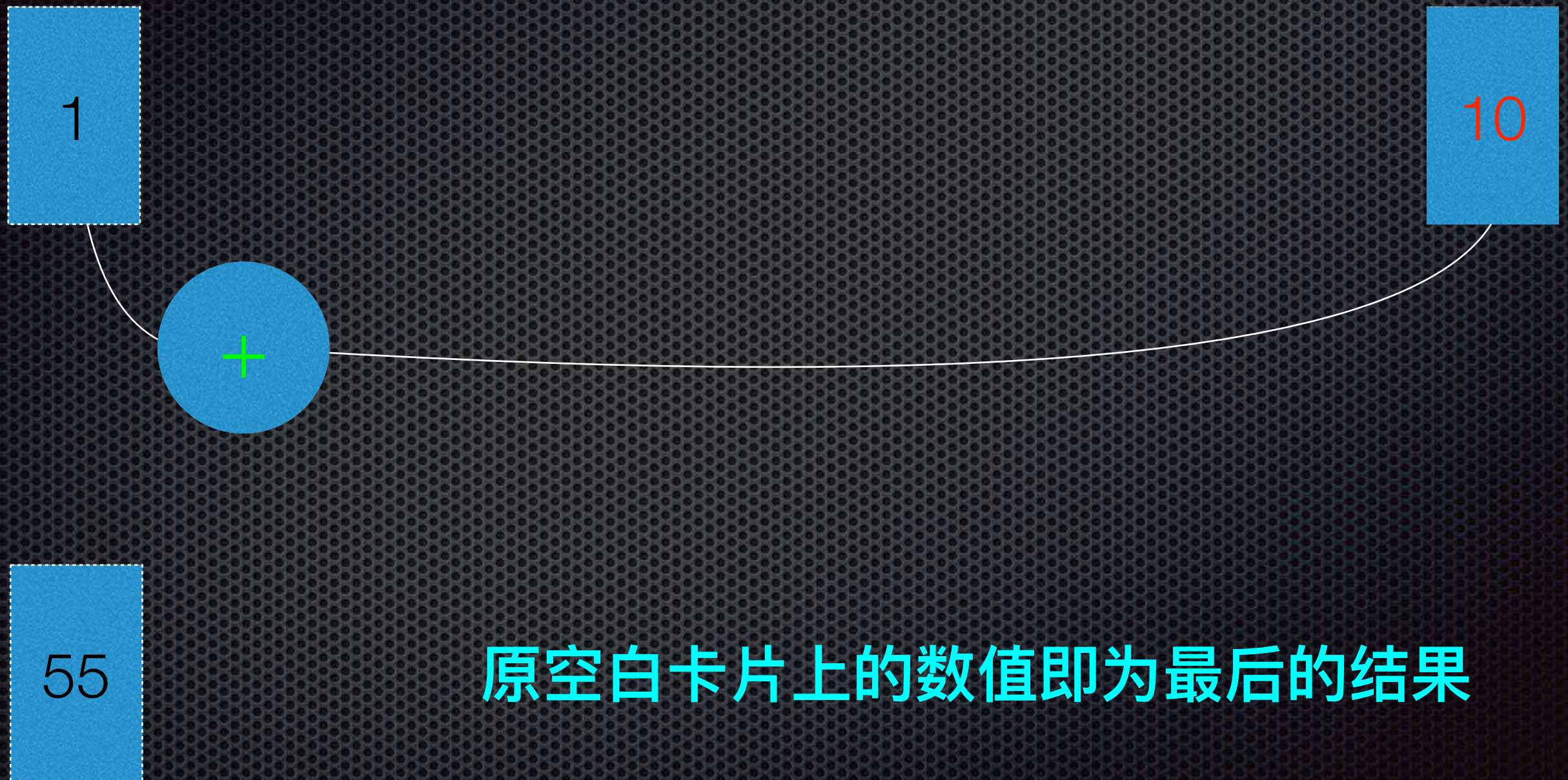
整理方法1 (续)



整理方法1 (续)



整理方法1 (续)



编制程序（方法1）

1. 在空白卡片 (x) 上写0
2. 取第一张卡片 (1)
3. 计算两张卡片上的数值的和
4. 擦除卡片x上的数值，并写入前一步的计算结果
5. 丢弃第一张卡片
6. 取当前的第一张卡片 (2)
7.
8. 报告卡片x上的值

方法2

- ❖ 执行主体的能力?
 - ❖ 我只会两个整数的加法和乘法，：)
- ❖ 合适的方法
 - ❖ $1+10$ 、 $2+9$ 、.....; $11 * 5$
 - ❖ 课堂作业，整理并描述你的计算过程

方法3

- 执行主体

- 我可比前两个厉害多了，除法都会，：））

- 合适的方法

- $(\text{首项} + \text{尾项}) * \text{项数} / 2$

- 课堂作业，整理并描述你的计算过程

借助计算机完成某个任务

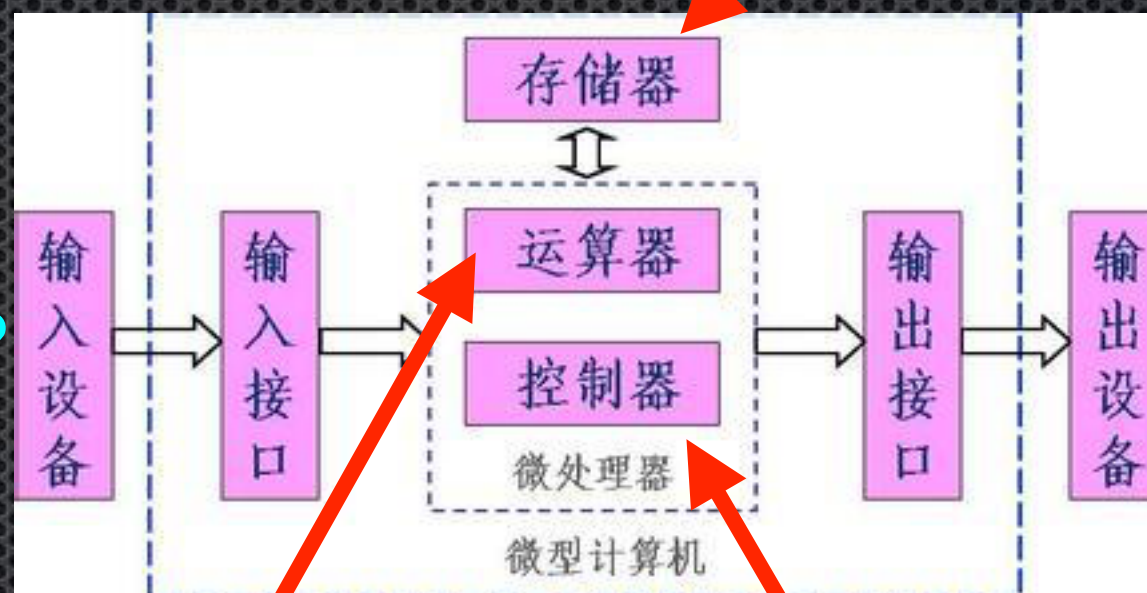
- ❖ 计算机：程序的执行者 你懂的，我懂吗？！
- ❖ 如何让计算机知道你要它做什么，怎么做？
- ❖ 如何告诉计算机在执行程序时需要处理的内容？
- ❖ 如何规范和限制计算机的操作？（能够明白你的指示）
- ❖ 如何得到你想要的结果？ 知己更要知彼！

计算机是个啥玩艺?



你要的都在我这，：)

你想怎么样?



我就这样!

必须听我的!!

不会别的，就是会算，就是能算!

有关计算机的部分基本概念

- ❖ 二进制
- ❖ 算术运算和逻辑运算
- ❖ 数据表示
- ❖ 存储器（一堆盒子）
- ❖ 顺序执行
- ❖ 软件体系结构
 - ❖ 低级语言、高级语言

问题

- 1. 如何表示数据?
- 2. 如何存放数据?
- 3. 如何进行计算?
- 4. 如何完成数据变更?
- 5. 如何告诉我计算的结果?
- 6. 如何将这些编制成计算机程序?
- 7. 如何运行计算机程序?
- 8. 天啊~计算机死了, 结果不对, …… , : (

程序设计

- 程序设计语言

- 第1, 2, 3, 4, 5个问题

- 编译器

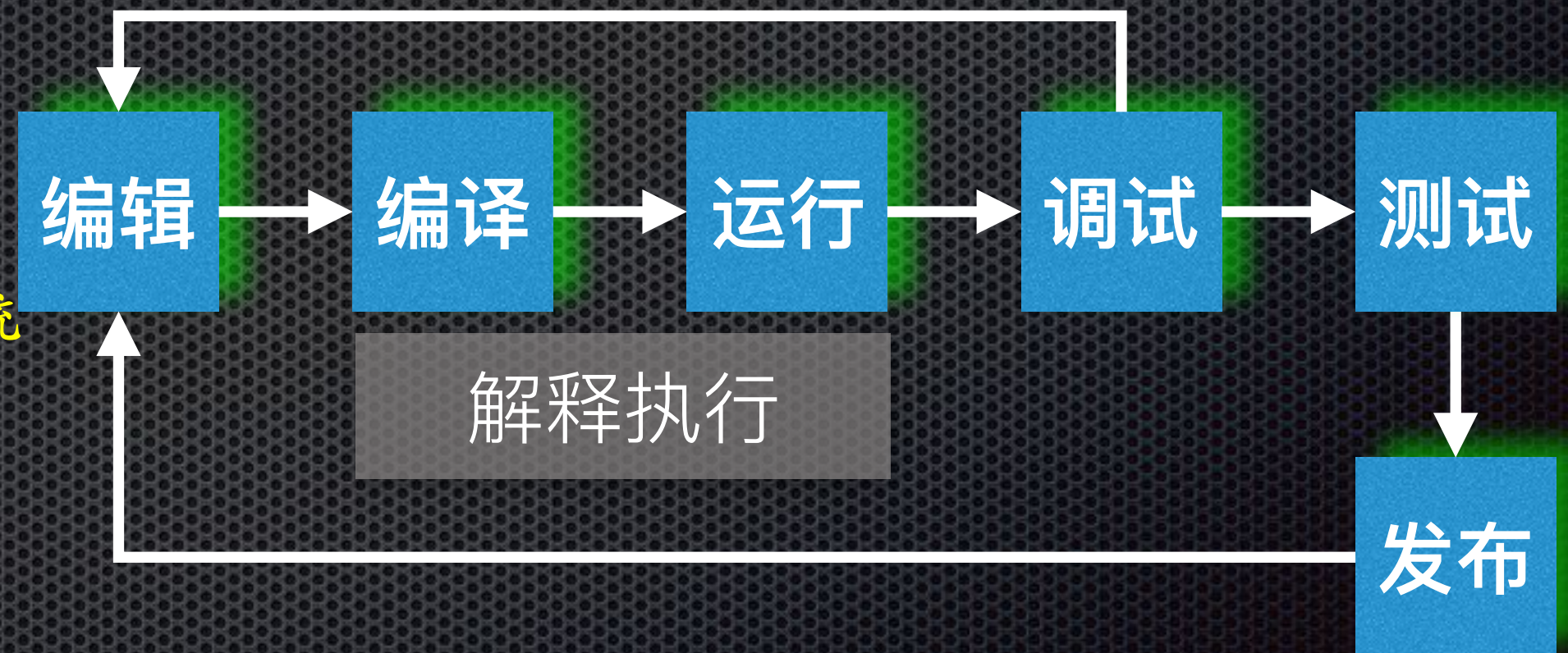
- 第6个问题

- 计算机&操作系统

- 第7个问题

- 调试器






















- 第8个问题



程序设计语言

- ❖ 多，真多
- ❖ 新的语言接二连三的出现
- ❖ 2017年编程语言排行榜 (IEEE)



Language Rank	Types	Spectrum Ranking
1. Python	 	100.0
2. C	  	99.7
3. Java	  	99.5
4. C++	  	97.1
5. C#	  	87.7
6. R		87.7
7. JavaScript	 	85.6
8. PHP		81.2
9. Go	 	75.1
10. Swift	 	73.7

C语言

为什么会是C语言?

- ✧ ken & dmr
- ✧ 游戏的动力
- ✧ PDP-7(利器)
- ✧ unix & C Language
- ✧ open source
- ✧ Gartner, 计算机程序设计艺术



能力培养

- ❖ 分析问题
- ❖ 构造算法（符合计算思维）
- ❖ 编写代码（不要把语言当方法）
- ❖ 调试程序（不要把技巧当技术）

重点是解决问题的思路

学习步骤

- ❖ 从看到编写
- ❖ 由简单到复杂
- ❖ 切忌死背死抠语法
 - ❖ 用不好的可以先不用
 - ❖ 重点是解决问题

学习方法

- 多实践、多调试
- 从程序的运行过程中，体会.....

关于

- ✧ 关于课程

- ✧ 关于实验

- ✧ 关于学习

- ✧ 关于考试

- ✧ 关于团队

- ✧ 计算应用技术研究所、创明工作室、……

- ✧ 考研还是去工作（四年后）