如何理解，下手完成编程题目

很多同学面对编程题时会大脑空白，不知道第一行代码该怎么敲。其实，做编程题有点像是**在计算机里搭建一条数据加工流水线。**

我们可以直接以下面这道**简单加法**作为对象来讲解，带同学们去读懂题目了解题目。

简单加法

- **题目要求：** 编写一个程序，接收用户输入的两个整数，然后计算并输出它们的和。
- **输入格式：** 共两行，每行一个整数。
- **输出格式：** 一个整数，代表输入的两个整数的和。
- **输入示例：**

```
15
20
```

- **输出示例：**

```
35
```

第一步：题目要求

题目原文：编写一个程序，接收用户输入的两个整数，然后计算并输出它们的和。

任务：你要建一条流水线，专门处理“加法计算”。

明确要做什么：在脑海中明确核心。这题的任务：**把收到的两个数加起来。**

第二步：什么是输入格式？—— 用于接收原材料

输入格式：共两行，每行一个整数。

输入示例：15 20

- **输入到底是什么意思？**
  输入，就是评测系统发给你的**原材料。**在工厂里，你不能凭空变出产品，必须等待系统把数字喂给你的程序，然后你再根据原材料产出产品。
- **误区**：
  很多同学看到样例是 15 和 20，就会在代码里直接写 `int a = 15; int b = 20;`。**这是绝对错误的！**
  输入格式告诉你的是原材料的**形状和数量**（比如：这里会来两个整数），而不是**具体内容。**系统只是保证我会给你两个整数，但它真正在测试的时候，可能给的是 100 和 200，也可能是 -5 和 8。
- **我们该怎么做？**
  我们要提前准备好对应数量的收纳盒（声明变量），然后通过`cin`，等系统把数据塞进来。

第三步：什么是输出？—— 交付最终产品

输出格式：一个整数，代表输入的两个整数的和。
输出示例：35

- **输出到底是什么意思?**
  输出，就是你的工厂加工完毕后，通过传送带交还的**最终产品**。
- **输出格式**：
  这是很容易遗漏的地方，因为最终对于的结果正误判定非常死板，它**只**核对最终交出来的东西是不是它想要的。
- **如果你写了**：`cout << "计算结果是：" << a + b;`
- **结果判断**：**输出要求的是一个整数，代表输入的两个整数的和。**它只看到了一堆汉字和一个整数。它期望只看到一个纯粹的 整数。于是它就会给你打上一个叉。
- **我们该怎么做?**
  严格比对输出示例与输出格式。输出格式里写着：一个整数；并且在输出示例里只有数字 `35`，那你的程序就绝对不能多输出除了结果以外其他的东西。

第四步：输入/输出示例的作用 —— 你的试金石

> **输入示例**：15, 20
> **输出示例**：35

- 样例绝对不是让你照抄的答案，而是给你在草稿纸上**验算你自己逻辑**的工具。
- **验算过程**：我拿到了 15 和 20（输入），我们的大脑思考 15 + 20 （逻辑处理），我得出了 35（输出）。这和样例对上了！说明我的逻辑是通的。

第五步：将理解变成 C++ 代码

当在脑海里走完上面这个流水线后，写代码就只剩下纯粹的翻译工作了：

动作	思路	C++ 代码
准备盒子	我需要两个变量来装系统给的整数。	`int a, b;`
接收原料 (输入)	把系统给的数据吸进变量里。	`cin >> a >> b;`
加工计算	把两个盒子里的数字加起来。	`int sum = a + b;`
交付产品 (输出)	直接把结果按照题目要求的格式丢出去。	`cout << sum << endl;`

总结：

输入就是承诺给你的材料，**输出**是你承诺给它的产品。不要加东西，也不要对着样例照写。我们要写的是**通用的加工配方，不管拿进来的是什么大小的材料，最终给出的标准的成品**！