

第一部分 编程基础



亲爱的小朋友，欢迎来到编程的世界！编程跟我们平时学习的其他学科有很大的不同，我们在学习其他学科时，可能一本书、一支笔、一本作业本就足够了。但是学习编程，我们需要在计算机（日常称电脑）上操作，所以在正式学习编程之前，我们先要了解计算机的一些基础知识，比如计算机的发展历史、计算机由哪几部分组成、编程都有哪些步骤，等等。这一部分也会介绍一下有关 GESp 的知识，因为在 GESp 考试中也会有与认证考试本身有关的题目。在这一部分的最后，我们会带着大家一起编写一个简单的加法小程序，体验编程的流程。



第1章 GESP 介绍与二进制

作为本书的第1章，本章将介绍 GESP 的知识与二进制，你将了解到：

- ▶ 什么是 GESP，GESP 认证的语言包括哪几种，GESP 有几个级别。
- ▶ GESP 的考试频次和题目安排。
- ▶ 参加 GESP 认证考试有哪些好处。
- ▶ 二进制是怎样计数的。
- ▶ 怎样把一个二进制数转换成十进制数。
- ▶ 八进制与十六进制的知识。

1.1

GESP 介绍

既然这本书是关于 GESP 的，那么在我们正式学习编程之前，要搞明白两件事情，第一，GESP 是什么；第二，为什么要参加 GESP 认证。

1.1.1 什么是 GESP

GESP 的全称为 Grade Examination of Software Programming，即编程能力等级认证，是衡量大家计算机和编程能力的一个平台，是由中国计算机学会（即 CCF）主办的（如图 1-1 所示）。CCF 是一个非常权威的机构，我们现在熟知的全国青少年信息学奥林匹克竞赛（即 NOI），就是由 CCF 主办的。



编程能力等级认证
Grade Examination of Software Programming

图 1-1

GESP 是 CCF 于 2022 年刚刚推出的，是一个相对来说比较新的认证，所以现在可能很多家长还从来没有听说过。CCF 推出这个平台的目的是，提升青少年计算机和编程教育的水平，以及推广和普及青少年计算机和编程教育。

大家知道，现在咱们国家非常重视编程教育。现在有一个词非常流行，叫 AI（人工智能），即让电脑模仿人类来思考。AI 是新一轮科技革命和产业变革的重要驱动力量，是一门新的技术科学。咱们国家其实早在 2017 年的时候，就制定了《新一代人工智能发展规划》（如图 1-2 所示）。我们要去搞 AI，首先就要学会编程，所以编程将来是每个人必

备的技能。就像我们现在每个人都会玩手机和电脑一样，以后编程是每个人都会的，因为到时候 AI 是无处不在的。

GESP 是面向所有的中小学生的，即使是一年级的小朋友也可以参加 GESP 认证考试。而且就学习编程来讲，越是低年级的小朋友越有优势，因为低年级的小朋友空余时间相对来说多一些，可塑性也更强。

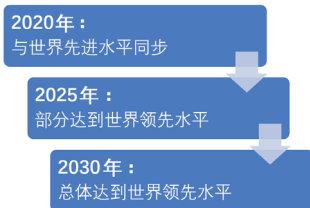


图 1-2

1.1.2 GESP 的语言和级别

GESP 考查的语言有三种，图形化编程（即 Scratch）、Python 及 C++（如图 1-3 所示）。GESP 一共分 8 个等级，但并不是每个等级都能选三种语言，其中一~四级可以选择 Scratch、Python 和 C++，而五~八级就只能选择 Python 和 C++（如图 1-4 所示）。



图 1-3

GESP 一~四级	GESP 五~八级
Scratch、Python 和 C++	Python 和 C++

图 1-4

GESP 规定，一级必考，不可以直接考二级，但是如果一级达到 90 分的话，二级是可以跳过去的。还有一些其他的跳级规则，大家可以到官网上去查询。

GESP 还规定，等级与语言是没有关联的，即同一个级别，无论是用什么语言（即 Scratch、Python 和 C++）考的，都是等价的，都是认可的。举个例子，GESP 规定，一级是必考的，只有考了一级才可以报考二级。那么，如果我一级考试用的是 Scratch，然后二级我想选择 Python 或者 C++，可以吗？答案是可以的，因为 Scratch 的一级跟 Python 或者 C++ 的一级是等价的。

既然这样，可能有些小朋友要问了，我可不可以先去学 Scratch，等到过一段时间再转成 C++？这么做理论上是可以的，但是浪费了一些时间，效率不高。笔者曾听说过一个案例，有个小孩之前学的 Java，已经很厉害了，都拿到全国的奖了，但因为后面没有赛道，又从头来考 C++ 的 GESP 一级。Scratch 也是一样，到了五~八级，就没有 Scratch 了，这时如果你要继续考级，就要转成 Python 或者 C++。虽然 Scratch 前面的等级 GESP 是认可的，但是 Scratch 的知识对于 Python 或者 C++ 来说，却没有多少借鉴作用，到头来你还是要从最基本的学起。

那么，我可不可以先学 Python，然后再转成 C++，或者干脆就不转呢？这个要看你后续的发展道路。如果后续你想参加 NOI，那么你不得不转成 C++，因为 C++ 是 NOI 唯一指定的语言。再者，Python 和 C++ 虽然都是高级语言，但是 C++ 是编译型语言，执行效率更高，而且就学习难易程度而言，笔者并不认为 Python 比 C++ 简单多少，为什么非得绕这么一个弯子呢？

1.1.3 GESP 的考试频次和题目安排

GESP 一年考试四次,分别是在三月、六月、九月和十二月。考试的形式是线下机考,就是说它是一个线下的考试,要到一个指定的考点,而不是在自己家里上网考试。考试时间是 120 分钟,满分是 100 分,60 分及格。单选题 15 道,每题 2 分,一共 30 分。判断题 10 道,每题 2 分,一共 20 分。编程题 2 道,每题 25 分,一共 50 分。单选题和判断题大部分考的都是概念,大家千万不要小看这些概念题,它们加起来一共也有 50 分的。

1.1.4 为什么要参加 GESP 认证考试

接下来我们就要讲一讲为什么要参加 GESP 认证考试。首先,是为了让你对你的计算机和编程能力有个全面的了解。大家如果平时自己学习的话,往往可能学习了半年或者一年,但并不知道自己的真实水平。就跟大家在学校里学习语文和数学一样,如果你不参加任何考试的话,是不知道自己的水平的。如果不知道自己的水平,那么学了一年半载以后,到底要不要继续学呢?要学的话,每天或者每周安排多少时间来学?这些你都无从确定。

笔者就遇到过这样一位小朋友,他自学了一段时间编程,自己感觉学得很好,结果我让他做了一套模拟试卷,发现原来还有很多不会的。所以考试本身是为了了解你当前的真实水平。

参加 GESP 考试,除了可以了解自己的编程水平,还有一个实实在在的好处,就是可以增加大家进入省市重点中学以及将来进入名校的机会。由于咱们国家现在对编程技能越来越重视,很多省市的重点中学都把获得 GESP 证书(如图 1-5 所示)作为招生条件之一。如果你后续继续去参加 NOI 并获奖的话,很多高校,如北大、清华等名校,也会向你抛来橄榄枝。所以,学习编程是实现弯道超车的一个好机会。



图 1-5

1.2 二进制

我们知道,计算机也称电脑。什么叫电脑呢?因为计算机是由很多电路组成的。电路通常只有两个状态,接通与断开,这两种状态正好可以用 1 和 0 来表示,所以在计算机里,数据的存储和运算采用的都是二进制(如图 1-6 所示)。我们要学习编程,也必须先了解二进制。



图 1-6

1.2.1 感受二进制

我们首先来感受一下什么是二进制。

我们先看一看十进制是怎样计数的。十进制是逢十进一，也就是说当你在数数的时候，从零开始数，零、一、二、三、四、五、六、七、八、九，每个数都用一个符号表示，即 0、1、2、3、4、5、6、7、8、9。九后面是什么呢？是“十”。这个时候需要进位，要用两个符号来表示“十”，变成了 10，右边第一个符号（称作个位数）变成了 0，进过去的这一位（称作十位数）变成了 1。这就是十进制的计数原理。

从 10 开始，继续往后数，11、12、13，直到 19，下一个数又要进一位，于是十位数变成了 2，个位数又变成了 0，于是就变成了 20。十进制就是这样，逢十进一。

那么二进制呢？二进制，简单来讲，就是逢二进一。在十进制里，因为逢十进一，所以是没有表示“十”这个数的单个符号的，“十”这个数要用两个符号来表示。同样的道理，在二进制里，因为逢二进一，所以是没有表示“二”这个数的单个符号的，只有“0”和“1”两个符号（如图 1-7 所示）。

1010011101000101

图 1-7

我们试着用二进制来数数。零还是 0，一还是 1，接下来应该是二，但是二进制逢二进一，没有表示“二”的单个符号，于是进一位，同时第一位数变成 0，于是变成了 10。但这个时候，10 不能读成“shi”，我们直接把每个符号按顺序读出来，10 读成“yi ling”，它表示数值二。10 后面是 11，表示的数值是三。

11 后面再加 1，第一位变成二了，必须向第二位进 1 并且自己变成 0，这时第二位也变成了二，于是向第三位进 1，同时第二位变成 0，所以最后就是 100。100 表示的数值就是四。这个过程可以用竖式进行，如图 1-8 所示，图中两个很小的“1”表示进位。

$$\begin{array}{r} 11 \\ + \quad 11 \\ \hline 100 \end{array}$$

图 1-8

然后是 101、110、111、1000，相应的数值为五、六、七、八，等等。

1.2.2 数码和基数

接下来让我们学习进制中的一些术语。

数码：数制中表示基本数值大小的不同数字符号。例如：

- 十进制有 10 个数码：0、1、2、3、4、5、6、7、8、9。

- 二进制有 2 个数码: 0、1。

基数: 数制所使用的数码的个数。例如, 二进制的基数为 2; 十进制的基数为 10。

拓展

上面的结论可以推广到 N 进制, 即 N 进制有 N 个数码, N 进制的基数为 N 。例如, 八进制有 8 个数码, 分别为 0、1、2、3、4、5、6、7, 基数为 8。十六进制应该有 16 个数码, 基数为 16, 可是阿拉伯数字一共只有 10 个符号, 从 0 到 9, 如何表示 16 个数码呢? 这时, 我们就需要引入新的符号。十六进制中, 我们用 a、b、c、d、e、f (或者 A、B、C、D、E、F) 表示 10、11、12、13、14、15。

1.2.3 二进制表示

接下来, 问题出现了。虽然在计算机内部存储的都是二进制, 但毕竟二进制不利于阅读, 就像我们前面看到的那样, 二进制里的 111, 其实是十进制的 7, 所以我们在写代码的时候, 绝大部分情形下仍然使用十进制。那么, 对于一个数, 比如 11, 我们如何知道它是十进制的“十一”, 还是二进制的“一一”(即数值 3)呢?

方法是, 当我们使用二进制表示一个数的时候, 前面必须加上前缀 0b 或者 0B (如图 1-9 所示)。所以, 11 表示的是“十一”, 0b11 或者 0B11 表示的是“三”。而且, 使用二进制表示一个数时, 通常都需要很多位, 比如 10 000 这个十进制数, 用二进制表示有 14 位, 所以我们一般会每 8 位 (也有的地方是每 4 位) 加一个空格, 不足 8 位的前面补 0, 所以 10 000 用二进制表示时就是 0b00100111 00010000。如何把一个十进制数转换成二进制数, 将在 GESP 三级教程中讲解。

0b10100111, 0B01000101

图 1-9

1.2.4 二进制转十进制

那么, 当我们看到一个二进制数, 尤其是看到一个很长的二进制数时, 我们怎么知道它表示的数值是多少呢? 的确, 对于一个很长的二进制数, 你是无法一眼看出它表示的数值的, 但我们又的确需要知道它的大小, 这就需要把它转换成十进制数。这时, 我们就需要了解一个新的概念——位权。

简单来说, 位权就是一个数中, 某一位表示的数值的大小。我们仍然以十进制数为例, 我们来看 6352 这个数。这个数有 4 位, 分别为 6、3、5、2, 其中 2 就表示 2, 但是 6、3、5 呢? 它们就表示 6、3、5 吗? 如果这里的 6、3、5 仅仅表示 6、3、5 的话, 那么 $6352=6+3+5+2=16$, 显然这是不对的。

我们仔细分析一下， $6352=6000+300+50+2$ ，所以这里的 5 表示 50，即 5×10^1 ，3 表示 300，即 3×10^2 ，6 表示 6000，即 6×10^3 。这里， 10^1 就表示 5 所在的位置的位权， 10^2 就表示 3 所在的位置的位权， 10^3 就表示 6 所在的位置的位权。我们有这样的结论，从右边开始数，第 n 位的位权为 10^{n-1} ，如图 1-10 所示。

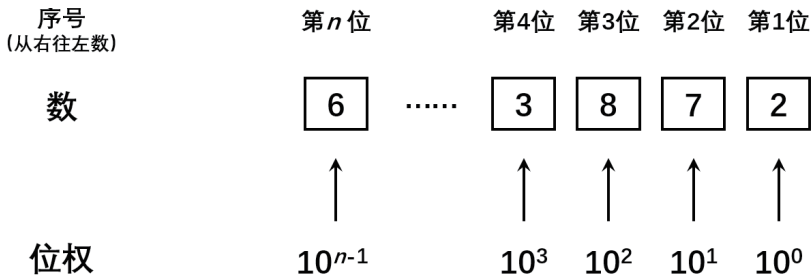


图 1-10

提示

- (1) 10^n 表示 n 个 10 相乘。
- (2) 任何一个非零数的 0 次方等于 1，所以 $10^0=1$ 。

所以一个十进制数表示的值的大小，等于它的每个位上的数乘以这个位的位权，然后求和。这个方法称为位权展开求和法。

上述理论和求和方法同样适用于二进制数。对于二进制数，从右边开始数，第 n 位的位权为 2^{n-1} ，如图 1-11 所示。

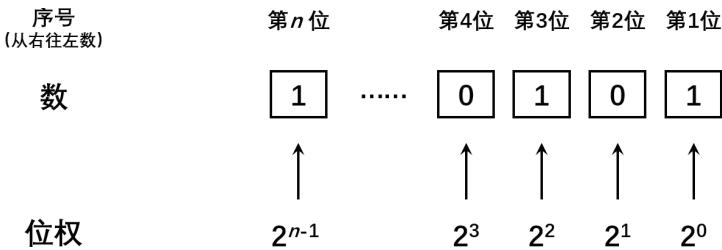


图 1-11

对于一个二进制数，从右往左，位权依次为 2^0 、 2^1 、 2^2 、 2^3 、 2^4 ，等等。

由于同学们可能还没有学过幂运算，下面列出 $2^0\sim2^{10}$ 的值：

$2^0=1$	$2^1=2$	$2^2=4$	$2^3=8$
$2^4=16$	$2^5=32$	$2^6=64$	$2^7=128$
$2^8=256$	$2^9=512$	$2^{10}=1024$	

大家并不需要把这些数全部背下来，需要的时候用计算器算一下，或者在草稿纸上算一下就可以了。

在进行转换时，我们可以把每个位的位权标注在数字上面，然后把每个位上的数乘以

这个位的位权，再求和，如图 1-12 所示。

$$\begin{array}{cccc} 8 & 4 & 2 & 1 \\ 0b0110 & = & 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 1 & = & 6 \end{array}$$

图 1-12

由于 1 乘以任何数等于这个数本身，0 乘以任何数为 0，所以只需把数字为 1 的位上的位权相加即得到这个二进制数表示的数值，数字为 0 的不需要加（数字为 0 的位权也不用标），如图 1-13 所示。

$$\begin{array}{ccccccc} 128 & 32 & 16 & 4 & 1 \\ 0b10110101 & = & 128 + 32 + 16 + 4 + 1 & = & 181 \end{array}$$

图 1-13



【课堂练习】

请算出下面这两个二进制数表示的数值。

(1) 0b00001101

(2) 0b00110010

解答（请自己在每个 1 上面标上位权）：

(1) $0b00001101 = 8 + 4 + 1 = 13$

(2) $0b00110010 = 32 + 16 + 2 = 50$



【例题】

比较下面两个数的大小：

1001, 0b1001

分析：第一个数没有任何前缀，是十进制数，第二个数有 0b 前缀，是二进制数。两个不同进制的数，是无法直接看出大小的，必须转换成相同的进制。就目前而言，十进制数转换成二进制数还没有学到，所以我们将二进制数转换成十进制数。

使用前面的方法， $0b1001 = 8 + 1 = 9$ ，所以 $1001 > 0b1001$ 。



【课堂练习】

请比较下面两个数的大小：

1010, 0b1110

解答：把第二个数转换成十进制数， $0b1110 = 14$ ，所以 $1010 > 0b1110$ 。

1.2.5 常见的二进制数

我们已经学会了如何把二进制数转换成十进制数，但是把十进制数转换成二进制数，则相对来说有点难，表 1-1 列出了一些比较小的十进制数（以及几个特殊的十进制数）对

应的二进制表示。

表 1-1 常用的十进制数对应的二进制表示

十进制	二进制	十进制	二进制
0	0b0	6	0b110
1	0b1	7	0b111
2	0b10	8	0b1000
3	0b11	16	0b10000
4	0b100	32	0b100000
5	0b101	64	0b1000000

我们发现，从 8 开始，十进制数每乘以 2，二进制表示里末尾就多了一个 0。实际上，这正是二进制的本质，正如十进制里，末尾每多一个 0，这个数就变成了原来的数的 10 倍，在二进制里，末尾每多一个 0，这个数就变成了原来的数的 2 倍。这个规律同样可以扩展到 N 进制。

1.3 八进制和十六进制

我们前面在学习数码和基数的时候提到，八进制有 8 个数码，用 0 ~ 7 表示，十六进制有 16 个数码，用 0 ~ 9, a、b、c、d、e、f (或者 A、B、C、D、E、F) 来表示。除此以外，在书写八进制数的时候，我们必须以 0 打头，而在书写十六进制数的时候，必须以 0x 或者 0X 打头。



【真题解析】

判断下列说法是否正确：

C++ 表达式 010+100+001 的值为 111。

解析：如果你不知道以 0 打头的数为八进制，这条题目肯定会认为是正确的。但是现在，我们已经知道 010 是八进制数了，我们就要来计算一下。010 转成十进制数怎么转呢？方法跟二进制数转十进制数是一样的，用位权展开求和法。 $010 = 1 \times 8^1 + 0 \times 8^0 = 8$ ，所以 $010+100+001 = 109$ ，所以本题错误。

课后作业

- 小格去报名参加 CCF 组织的 GESP 认证考试的第一级，那么他可以选择的认证语言有几种？（ ）
A. 1 B. 2 C. 3 D. 4
- 比较下列两组数的大小：
(1) 111 ○ 0b1110
(2) 15 ○ 0b1111



延伸阅读：二进制数是一类特殊的数吗

在讲解二进制数的表示方法的时候，我向大家提了一个问题：当你看到一个数，比如 11，你怎么知道它是十进制数十一，还是二进制数三呢？结果一个小朋友反问我，二进制里有“三”吗？

这是一个有趣的问题。确实，二进制里只有 0、1、10、11、100、101 等，是没有读音为“san”的数的，那么难道这个问题本身不对吗？

这其实是因为我们平时的表述不够严密。我们平时习惯说十进制数和二进制数，但严格来讲，这种说法是不对的。数就是数，是没有所谓的十进制数和二进制数的，十进制和二进制，只是数的不同表示方法而已。就像我们在唱票时，采用画“正”字的方法来计数，也只是数的一种不同的表示方法（如图 1-14 所示，一个“正”字表示 5，图中表示的数为 18）。它们表示的数，还是那些数。

正正正下

图 1-14

所以上述问题，“11 是十进制数十一，还是二进制数三”，这样的问法是不严密的。我们永远看不到数，我们能看到的只是数的表示方法。所以上述问题严格来讲，应该这样问：当你看到“11”这样的符号时，你怎么知道它是一个十进制表示（此时表示的数值为“十一”，这里写成了汉字，因为如果写成 11，又变成表示方法了），还是一个二进制表示（此时表示的数值为三）呢？

但是，如果我们每次看到一个二进制表示的数，比如，0b111，都说这是一个“7 的二进制表示”，就显得很啰嗦，没有“二进制数 7”来得简洁。所以，在日常交流中，我们仍然会直接说“这是一个十进制数”“那是一个二进制数”，尽管这种表述并不严密。