初赛（一）

翁思源

信息学奥赛

wengsiyuan40@gmail.com

2026 年 8 月 22 日

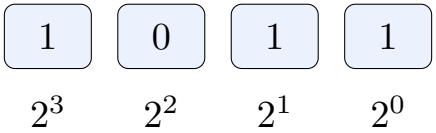
什么是 k 进制？

一句话理解

k 进制就是“逢 k 进一”，每一位只能使用 $0 \sim k-1$。

进制	可用数字	常见写法
二进制	$0, 1$	$(1011)_2$
八进制	$0 \sim 7$	$(27)_8$
十进制	$0 \sim 9$	$(35)_{10}$
十六进制	$0 \sim 9, A \sim F$	$(2AF)_{16}$

位权：同一个数字，位置不同，价值不同



$$(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$$

k 进制整数的展开规律

从最右边开始，位权依次是 $k^0, k^1, k^2, \ldots$；**每一位数字乘自己的位权，再全部相加。**

任意进制整数转十进制：按位展开

例：$(2A7)_{16}$ 转成十进制

十六进制中 $A = 10$：

$$(2A7)_{16} = 2 \times 16^2 + 10 \times 16^1 + 7 \times 16^0 = 512 + 160 + 7 = 679$$

考试易错点

- 最右位从 0 次方开始，不是从 1 次方开始；
- A, B, C, D, E, F 分别表示 $10, 11, 12, 13, 14, 15$。

十进制整数转 k 进制：除 k 取余

每轮做两件事

1. 余数是当前得到的一位；
2. 商继续除以 k，直到商为 0。

读数方向

第一次得到的是**最低位**，所以余数必须**从下往上读**。

$$\text{十进制整数} \xrightarrow{\text{不断除 } k \text{ 取余}} k \text{ 进制整数}$$

演示：把 $(45)_{10}$ 转成二进制

被除数	商	余数
$45 \div 2$	22	1
$22 \div 2$	11	0
$11 \div 2$	5	1
$5 \div 2$	2	1
$2 \div 2$	1	0
$1 \div 2$	0	1

从下往上读余数

$$(45)_{10} = (101101)_2$$

二、八、十六进制之间：分组最快

转换	原因	分组方法
二进制 $\leftrightarrow$ 八进制	$8 = 2^3$	每 3 位一组
二进制 $\leftrightarrow$ 十六进制	$16 = 2^4$	每 4 位一组

例

$$(11010110)_2 = (11\ 010\ 110)_2 = (326)_8$$

$$(11010110)_2 = (1101\ 0110)_2 = (D6)_{16}$$

整数部分从小数点向左分组，不足位在**最左边**补 0。

小数部分也有位权

$$101 \cdot \boxed{1}\ \boxed{1}$$
$$2^{-1}\quad 2^{-2}$$

$$(101.11)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = 5.75$$

规律

小数点右边第一位的位权是 k^{-1}，然后是 $k^{-2}, k^{-3}, \ldots$。

十进制小数转 k 进制：乘 k 取整

每轮步骤

1. 小数部分乘 k；
2. 取乘积的整数部分，作为下一位；
3. 保留新的小数部分，继续乘 k。

读数方向

与整数转换相反：第一次得到的是**小数点后的第一位**，所以按**从上往下**的顺序读。

演示：把 $(0.625)_{10}$ 转成二进制

计算	整数部分	新的小数部分
$0.625 \times 2 = 1.25$	1	0.25
$0.25 \times 2 = 0.5$	0	0.5
$0.5 \times 2 = 1.0$	1	0

从上往下读整数部分

$$(0.625)_{10} = (0.101)_2$$

检查

$2^{-1} + 2^{-3} = 0.5 + 0.125 = 0.625$。

带小数的数：整数、小数分开处理

把 $(13.625)_{10}$ 转成二进制

$$13 = (1101)_2, \qquad 0.625 = (0.101)_2$$

合并得到：

$$(13.625)_{10} = (1101.101)_2$$

不能把整个数一直“除 2 取余”

整数部分用“除基取余”，小数部分用“乘基取整”，最后在小数点处合并。

有些小数不会有限结束

例：$(0.1)_{10}$ 转二进制

乘 2 后的小数部分会不断变化，无法在有限位内恰好变成 0，因此得到循环或无限展开。

考试处理原则

- 题目给“保留若干位”，就算到指定精度；
- 题目问“精确相等”，要警惕它可能不能有限表示；
- 不要看到没有结束就误以为计算错了。

真题：CSP-J 2021 第一轮

第 7 题

二进制数 101.11 对应的十进制数是（　）。

- A. 6.5
- B. 5.5
- C. 5.75
- D. 5.25

来源：CSP-J 2021 第一轮第 7 题

真题解析：别漏掉负指数位权

$$(101.11)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$
$$= 4 + 0 + 1 + 0.5 + 0.25 = 5.75$$

答案

C

检查点

小数点右边不是 $2^1, 2^2$，而是 $2^{-1}, 2^{-2}$。

真题：CSP-J 2022 第一轮

第 13 题

八进制数 $(32.1)_8$ 转换成十进制数是（　）。

A. 24.125
C. 26.125
B. 24.250
D. 26.250

来源：CSP-J 2022 第一轮第 13 题

真题解析：八进制小数同样按位权展开

$$(32.1)_8 = 3 \times 8^1 + 2 \times 8^0 + 1 \times 8^{-1}$$
$$= 24 + 2 + \frac{1}{8} = 26.125$$

答案

C

易错原因

把 $(32)_8$ 看成十进制 32，或把小数第一位误当成 $1/10$。

真题：CSP-J 2024 第一轮

第 2 题

计算 $(14)_8 - (1010)_2$，再乘 $(D)_{16}$，最后减去 $(1101)_2$，结果是（　）。

A. 13
B. 14
C. 15
D. 16

来源：CSP-J 2024 第一轮第 2 题

真题解析：先统一成十进制

$$(14)_8 = 12, \quad (1010)_2 = 10, \quad (D)_{16} = 13, \quad (1101)_2 = 13$$

$$(12 - 10) \times 13 - 13 = 2 \times 13 - 13 = 13$$

答案

A

做题顺序

先在每个数旁边写出十进制值，再按照括号和乘减顺序计算。

真题：NOIP 2018 普及组初赛

下列四个不同进制的数中，与其他三项数值不同的是（　）。

- A $(269)_{16}$
- B $(617)_{10}$
- C $(1151)_{8}$
- D $(1001101011)_{2}$

来源：NOIP 2018 普及组初赛

真题解析：全部转成同一进制比较

$$
\begin{aligned}
(269)_{16} &= 2\times 256 + 6\times 16 + 9 = 617,\\
(1151)_{8} &= 1\times 512 + 1\times 64 + 5\times 8 + 1 = 617,\\
(1001101011)_{2} &= 512 + 64 + 32 + 8 + 2 + 1 = 619.
\end{aligned}
$$

答案

D

技巧

遇到“哪一个不同”，不必两两比较；统一转成十进制最稳。

先问位数：同一个数有不同长度的表示

固定字长

计算机通常用固定数量的二进制位保存一个整数，例如 8 位、16 位或 32 位。

正数 5

8 位：00000101　　16 位：0000000000000101

为什么必须先看位数？

符号位、取反结果、补码范围都依赖总位数。没有位数，就不能随意补多少个前导位。

三种编码先看共同点

情况	原码、反码、补码	处理
正数	三者完全相同	最高位为 0
负数	三者不同	最高位为 1

8 位表示 $+5$

$$\text{原码} = \text{反码} = \text{补码} = 00000101$$

学习重点

真正需要分步骤处理的是**负数**。

负数的原码、反码、补码怎么求？

以 8 位的 -5 为例

1. 先写 $+5$：　00000101
2. **原码**：符号位改为 1：　10000101
3. **反码**：符号位不变，其余位取反：　11111010
4. **补码**：反码加 1：　11111011

口诀必须带条件

负数补码：**先写原码，符号位不动，其余取反，再加** 1。

为什么计算机更喜欢补码？

原码的问题

- 加法与减法需要额外判断符号；
- 有 $+0$ 和 -0 两种零。

补码的好处

- 加减法可以统一为二进制加法；
- 零只有一种表示；
- 能多表示一个最小负数。

本节要求

初赛先掌握“如何编码、如何还原、范围为何不对称”，不展开硬件电路。

从补码还原负数：取反加一

8 位补码 11111011 表示多少？

1. 最高位是 1，说明它表示负数；
2. 全部位取反：00000100；
3. 再加 1：$00000101 = 5$；
4. 加上负号，得到 -5。

注意

这是“补码求绝对值”的逆向步骤；不要只把最高位改成负号。

从补码还原负数：无符号值减 2^n

更快的统一公式

对一个 n 位补码：若最高位是 1，先按无符号二进制算出 U，则

$$\text{有符号值} = U - 2^n.$$

8 位补码 10101011

$$U = 128 + 32 + 8 + 2 + 1 = 171$$

$$171 - 2^8 = 171 - 256 = -85$$

选哪种方法？

“取反加一”更直观；“减 2^n”通常更快。会一种即可，最好能互相检查。

n 位补码能表示多大范围？

结论

$$-2^{n-1} \leq x \leq 2^{n-1} - 1$$

位数	补码范围
8 位	$-128 \sim 127$
16 位	$-32768 \sim 32767$
32 位	$-2147483648 \sim 2147483647$

为什么负数多一个？

补码只有一个 0，原本“负零”的位置被用来表示最小负数。

真题：NOIP 2017 普及组初赛

在 8 位二进制补码中，10101011 表示的十进制数是（　）。

A. 43
B. -85
C. -43
D. -84

来源：NOIP 2017 普及组初赛

真题解析：两种方法都得到 -85

取反加一

$$10101011 \to 01010100$$

$$01010100 + 1 = 01010101 = 85$$

因此是 -85。

减 2^8

$$(10101011)_2 = 171$$

$$171 - 256 = -85$$

答案

B

真题：CSP-J 2024 第一轮

32 位 `int` 类型的存储范围是（　）。

A. $[-2147483647,\ 2147483647]$
B. $[-2147483647,\ 2147483648]$
C. $[-2147483648,\ 2147483647]$
D. $[-2147483648,\ 2147483648]$

来源：CSP-J 2024 第一轮第 1 题；题干明确采用 32 位补码

真题解析：一位符号，范围不对称

$$-2^{32-1} \leq x \leq 2^{32-1} - 1$$

$$-2147483648 \leq x \leq 2147483647$$

答案

C

严谨提醒

这里能直接写出范围，是因为题目已经说明“32 位补码”。实际 C++ 环境中，类型宽度应以题目或环境说明为准。

位运算：把整数看成一排开关

运算	C++ 符号	直观含义
按位与	&	两位都是 1 才是 1
按位或	|	至少一位是 1 就是 1
按位异或	^	两位不同才是 1
按位取反	~	每一位 $0 \leftrightarrow 1$
左移、右移	<<,>>	整体移动二进制位

“按位”的意思

先把两个数写成等长二进制，再让同一列的两位单独计算。

三种双目位运算的真值表

a	b	$a \,\&\, b$	$a \mid b$	$a \oplus b$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

一句话记忆

与：**全 1 才 1**；或：**有 1 就 1**；异或：**不同才 1**。

演示：先右对齐，再逐列计算

计算 $13 \,\&\, 11$

$$13 = (1101)_2, \qquad 11 = (1011)_2$$

$$\begin{array}{r} 1101 \\ \underline{1011}\ \& \\ 1001 \end{array}$$

所以 $13 \,\&\, 11 = (1001)_2 = 9$。

常见错误

二进制位数不同时，应该在左侧补 0 后右对齐，不能从左边对齐。

异或为什么经常出现？

三个重要性质

$$x \oplus x = 0, \qquad x \oplus 0 = x$$

$$a \oplus b = b \oplus a, qquad(a \oplus b) \oplus c = a \oplus (b \oplus c)$$

直观解释

一个开关翻转两次会回到原状态，所以同一个数异或两次可以“消掉”。

别混淆

异或不是乘方。C++ 中乘方不能写成 ^。

按位取反：必须先确定总位数

4 位二进制中

$$\sim(0101) = 1010$$

8 位二进制中

$$\sim(00000101) = 11111010$$

结论

只写“~ 101”而不说明位宽，会隐藏前面还有多少位需要取反。手算题通常会给定位数或数据类型。

左移与右移：只先讨论非负数

左移 k 位

$$x \ll k \approx x \times 2^k$$

例：$5 \ll 2$：

$$00101 \to 10100 = 20$$

右移 k 位

$$x \gg k = \left\lfloor \frac{x}{2^k} \right\rfloor$$

例：$20 \gg 2$：

$$10100 \to 00101 = 5$$

边界说明

上述乘除关系用于非负数且没有溢出的情形。负数移位和溢出问题要结合语言规则与题目条件判断。

按位运算与逻辑运算不是一回事

类型	符号	计算对象
按位与、或	&,|	每一个二进制位
逻辑与、或	&&,||	条件的真与假

例

$6 \,\&\, 3 = (110)_2 \,\&\, (011)_2 = (010)_2 = 2$，而 `6 && 3` 的结果是真，即 1。

审题

初赛老题有时把“按位与”写成“逻辑与”，应结合符号、二进制对齐和选项判断题意。

位掩码：检查第 k 位是不是 1

构造只在第 k 位为 1 的掩码

$$1 \ll k$$

检查从右往左第 k 位（最低位编号为 0）

$$(x \gg k)\ \&\ 1$$

结果为 1，说明该位是 1；结果为 0，说明该位是 0。

下标意识

最低位是第 0 位，因此第 k 位的位权是 2^k。

位掩码：打开、关闭、翻转某一位

目标	表达式	依赖的性质
第 k 位置 1	$x \mid (1 \ll k)$	有 1 就为 1
第 k 位置 0	$x \,\&\, \sim(1 \ll k)$	与 0 得 0
第 k 位翻转	$x \oplus (1 \ll k)$	与 1 异或会翻转

学习要求

先理解“掩码只有目标位是 1”，再记表达式；死背符号很容易混。

真题：CSP-J 2019 第一轮

计算下面两个二进制数的按位与（原题表述为“逻辑与”）。

11 1011 1001 0111　　01 0110 1110 1011

A. 01 0010 1000 1011
B. 01 0010 1001 0011
C. 01 0010 1000 0001
D. 01 0010 1000 0011

来源：CSP-J 2019 第一轮第 2 题

真题解析：同一列必须同时为 1

$$\begin{array}{r} 11\ 1011\ 1001\ 0111 \\ \underline{01\ 0110\ 1110\ 1011}\ \& \\ 01\ 0010\ 1000\ 0011 \end{array}$$

答案

D

快速检查

与运算不可能把原来某一列的 0 变成 1，所以结果中的每个 1 必须在两个数的对应位都出现。

真题：CSP-J 2024 第一轮

以下哪个序列对应数字 $0 \sim 7$ 的四位格雷码？

A. $0000, 0001, 0011, 0010, 0110, 0111, 0101, 1000$

B. $0000, 0001, 0011, 0010, 0110, 0111, 0100, 0101$

C. $0000, 0001, 0011, 0010, 0100, 0101, 0111, 0110$

D. $0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100$

判断规则

相邻两个编码必须**恰好只有一位不同**。

来源：CSP-J 2024 第一轮第 4 题；以正式试卷为准

真题解析：逐对做异或，结果必须只有一个 1

对选项 D 的相邻编码检查：

$$0000 \to 0001 \to 0011 \to 0010 \to 0110 \to 0111 \to 0101 \to 0100$$

每一步只改变一位。例如：

$$0010 \oplus 0110 = 0100$$

结果中只有一个 1，说明只改变了一位。

答案

D

位运算联系

两个编码异或后，哪一位是 1，就表示那一位发生了变化。

CSP-S 2023 阅读程序第一题：先看主干

```cpp
unsigned short f(unsigned short x) {
    x ^= x << 6;
    x ^= x >> 8;
    return x;
}

int main() {
    unsigned short x;
    cin >> x;
    unsigned short y = f(x);
    cout << y << endl;
    return 0;
}
```

题目条件

输入 x 是不超过 65535 的自然数。核心考点是：**异或、移位、16 位截断**。

来源：CSP-S 2023 第一轮阅读程序第 1 题；正式题目与答案见历届试题索引

先读懂两条复合赋值语句

复合赋值展开

$$\texttt{x ^= x << 6} \iff \texttt{x = x ^(x << 6)}$$

$$\texttt{x ^= x >> 8} \iff \texttt{x = x ^(x >> 8)}$$

本题最容易漏掉的细节

x 是 16 位的 unsigned short。每次结果重新存回 x 时，只保留低 16 位。

草稿纸建议

全程写成 4 位十六进制数：0x0000 到 0xFFFF，比写 16 个二进制位更快。

代入演算（一）：65535 与 1

$x = 65535 = \texttt{0xFFFF}$

$$\texttt{0xFFFF} \texttt{ << } 6 \text{ 的低 } 16 \text{ 位} = \texttt{0xFFC0}$$

$$\texttt{0xFFFF} \oplus \texttt{0xFFC0} = \texttt{0x003F} = 63$$

此时右移 8 位为 0，所以最终仍为 63。

$x = 1 = \texttt{0x0001}$

$$\texttt{0x0001} \oplus \texttt{0x0040} = \texttt{0x0041} = 65$$

$65 >> 8 = 0$，最终输出 65，不是 64。

代入演算（二）：512 与 64

$x = 512 = 0\text{x}0200$

$$0\text{x}0200 \oplus 0\text{x}8000 = 0\text{x}8200$$

$$0\text{x}8200 \oplus 0\text{x}0082 = 0\text{x}8282 = 33410$$

$x = 64 = 0\text{x}0040$，执行完第一条异或后

$$0\text{x}0040 \oplus 0\text{x}1000 = 0\text{x}1040 = 4160$$

对应选择题

第 20 题选 B；第 21 题选 D。

CSP-S 2023：逐项结论

题号	判断依据	答案
16	两步变换都不会把非零 16 位数变成 0	T
17	参数改成 32 位后，中间高位不再立刻截掉	F
18	0xFFFF 最终变成 0x003F	T
19	输入 1 时输出 65，不是 64	F
20	输入 512 时输出 33410	B
21	输入 64，执行完第 5 行后为 4160	D

第 17 题为什么会变？

32 位参数会暂时保留左移产生的高位；下一次右移时，这些高位可能重新影响低 16 位。

CSP-J 2022 阅读程序第一题：程序在做什么？

```cpp
int main() {
    unsigned short x, y;
    cin >> x >> y;
    x = (x | x << 2) & 0x33;
    x = (x | x << 1) & 0x55;
    y = (y | y << 2) & 0x33;
    y = (y | y << 1) & 0x55;
    unsigned short z = x | y << 1;
    cout << z << endl;
    return 0;
}
```

题目条件

$0 \leq x, y \leq 15$。程序把 x, y 的 4 个二进制位**交错放入**结果 z。

来源：CSP-J 2022 第一轮阅读程序第 1 题

两个掩码：把二进制位逐渐“拉开”

掩码的二进制形状

$$\texttt{0x33} = 0011\ 0011, \qquad \texttt{0x55} = 0101\ 0101$$

z 的位编号	7	6	5	4	3	2	1	0
最终来源	y_3	x_3	y_2	x_2	y_1	x_1	y_0	x_0

一句话概括

x 被放到偶数位 $0,2,4,6$；y 先做同样处理，再左移 1 位，放到奇数位 $1,3,5,7$。

样例推演：输入 2 2

先处理 $x = 2 = (0010)_2$

$$(0010 \mid 1000)\ \&\ 0011\ 0011 = 0000\ 0010$$

$$(0000\ 0010 \mid 0000\ 0100)\ \&\ 0101\ 0101 = 0000\ 0100 = 4$$

y 的处理完全相同，也得到 4。

合并

$$z = x \mid (y << 1) = 4 \mid 8 = 12$$

所以“输出 10”和“输出 59”都错误。

选择题推演：输入 13 8

$x = 13 = (1101)_2$

拉开后：

$$x = 0101\ 0001 = 81$$

即原来的 4 位分别来到第 $6, 4, 2, 0$ 位。

$y = 8 = (1000)_2$

拉开后：

$$y = 0100\ 0000 = 64$$

再左移 1 位得到 128。

最终结果

$$z = 81 \mid 128 = 209$$

第 21 题选 B。

CSP-J 2022：逐项结论

题号	判断依据	答案
16	改为有符号 short，本题数值范围仍装得下	T
17	char 可能只有 $-128 \sim 127$，而结果可达 209	F
18	输入不同时当然可能得到非零结果	F
19	输入 2 2 时输出 12，不是 10	F
20	输入 2 2 时输出 12，不是 59	F
21	输入 13 8 时输出 209	B

类型题的判断方法

不要只看“能不能编译”，还要检查中间值与最终值是否超出新类型的表示范围。

CSP-J 2021 阅读程序第一题：两个经典位运算

```
int n;
int a[1000];
int f(int x) {
    int ret = 0;
    for (; x; x &= x - 1) ret++;
    return ret;
}
int g(int x) {
    return x & -x;
}
// main: cout << f(a[i]) + g(a[i]);
```

先给两个函数起名字

$f(x)$：统计二进制中 1 的个数；$g(x)$：取最低位的 1 所代表的权值。

来源：CSP-J 2021 第一轮阅读程序第 1 题

函数 f：每轮清掉最低位的一个 1

核心性质

$$x \,\&\, (x-1)$$

会把 x 最低位的那个 1 变成 0，更高位保持不变。

$x = 12 = (1100)_2$

$$1100 \to 1000 \to 0000$$

循环执行 2 次，所以 $f(12) = 2$。

因此

`ret` 增加了几次，就说明原二进制中有几个 1。

函数 g：只保留最低位的一个 1

补码关系

$$-x = \sim x + 1$$

因此 $x \,\&\, (-x)$ 会清掉其它位，只留下最低位的 1。

几个例子

$$g(10) = g((1010)_2) = (0010)_2 = 2$$

$$g(16) = g((10000)_2) = 16$$

常见名称

这个值通常写成 `lowbit(x)`，等于 x 中最低位 1 的位权。

样例逐项代入：最后一个数最容易算错

x	2	11	9	16	10
$f(x)$	1	3	2	1	2
$g(x)$	2	1	1	16	2
$f(x)+g(x)$	3	4	3	17	4

第 18 题

题目声称最后一个输出是 5，实际为 4，所以判断为 F。

边界与语法判断：不要只顾着算二进制

数组边界

`int a[1000]` 的合法下标为 $0 \sim 999$。当 $n = 1001$ 时会访问 `a[1000]`，发生越界。

是否会死循环？

$x = 0$ 时循环一次也不执行；在题目采用的 32 位补码语境中，负数也只有有限个二进制位，每轮仍会清掉一个 1。

函数位置

若把 g 的定义移到 `main` 后面，又没有提前声明，调用 `g(a[i])` 时编译器还不认识它，无法正常编译。

大数代入与最终答案

$x = 511998$

二进制中有 16 个 1，且最低位的 1 的权值为 2：

$$f(x) + g(x) = 16 + 2 = 18$$

$x = -65536$

32 位补码中有 16 个 1，`lowbit` 为 65536，结果 65552。

$x = 2147483647$

二进制为 31 个连续的 1：

$$f(x) + g(x) = 31 + 1 = 32$$

答案汇总

16–21：F，F，F，T，F，B。

最小单位：bit；常用单位：Byte

bit（比特、位）

只能取 0 或 1，是信息存储的基本二进制单位。

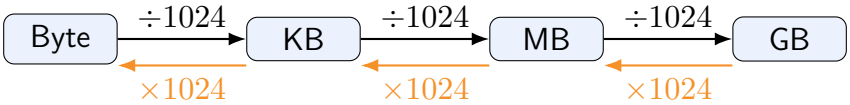
Byte（字节，B）

$$1\ \text{Byte} = 8\ \text{bit}$$

大小写很重要

小写 b 常表示 bit，大写 B 表示 Byte。8 Mb 与 8 MB 不是同一个容量。

初赛常用容量换算链



常见约定

初赛题常给出 1 KB = 1024 B、1 MB = 1024 KB。

以题面为准

现实标准中还会出现十进制单位与 KiB、MiB。考试若明确换算关系，必须按题面给出的关系计算。

容量计算统一套路：先算 bit，再换 Byte

原始图像的常用公式

$$\text{总 bit 数} = \text{宽} \times \text{高} \times \text{每像素 bit 数}$$

$$\text{总 Byte 数} = \frac{\text{总 bit 数}}{8}$$

数组同理

若题目明确一个整数占 4 Byte，那么 n 个整数占 $4n$ Byte。

默认不计什么？

若题目只给分辨率和色深，通常按未压缩像素数据计算；文件头、压缩率等只有题目说明时才计入。

真题：CSP-J 2024 第一轮

记 $1\ \text{KB} = 1024\ \text{Byte}$，$1\ \text{MB} = 1024\ \text{KB}$，那么 $1\ \text{MB}$ 是多少 bit？

A. $1\,000\,000$
B. $1\,048\,576$
C. $8\,000\,000$
D. $8\,388\,608$

来源：CSP-J 2024 第一轮第 5 题

真题解析：Byte 转 bit 还要乘 8

$$1\ \text{MB} = 2^{20}\ \text{Byte}$$

$$2^{20} \times 8 = 2^{23} = 8\,388\,608\ \text{bit}$$

答案

D

易错点

$1\,048\,576$ 是 Byte 数，不是 bit 数。

真题：CSP-J 2020 第一轮

一幅 2048×1024 像素、32 位真彩色的未压缩图像，需要多少存储空间？

A. 16 MB
C. 8 MB
B. 4 MB
D. 2 MB

来源：CSP-J 2020 第一轮第 4 题

真题解析：像素数乘每像素位数

$$2048 \times 1024 \times 32 \text{ bit}$$

$$= 2048 \times 1024 \times 4 \text{ Byte} = 8 \times 1024 \times 1024 \text{ Byte} = 8 \text{ MB}$$

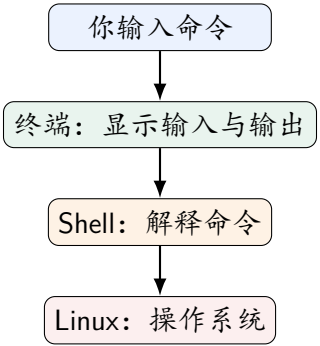
答案

C

快速思考

32 bit $= 4$ Byte，所以每个像素占 4 Byte，先除以 8 再计算更省草稿。

Linux、终端、Shell 分别是什么？



初赛关注点

认识常见命令的作用、相对路径和绝对路径、编译与运行的基本命令。

路径符号：先知道自己在哪里

符号	含义	例子
/	根目录；也用于分隔目录	/home/alice
.	当前目录	./main
..	上一级目录	cd ..
~	当前用户的家目录	cd ~

绝对路径与相对路径

以 / 开头的是绝对路径；不以 / 开头时，通常从当前目录出发寻找。

查看与切换目录：pwd、ls、cd

```
pwd
ls
ls -l
cd /home/alice
cd ..
cd ~
```

	错误写法（没有空格）	正确写法（有空格）
返回上一级	cd..	cd ..
查看详细列表	ls-l	ls -l

空格把“命令”和“参数”分开

cd 是命令，.. 是参数；ls 是命令，-l 是选项。

创建文件与目录：touch、mkdir

```
mkdir code
touch main.cpp
mkdir -p a/b/c
```

名称联想

mkdir = make directory；看到“创建目录”优先想到它。

复制、移动与删除：cp、mv、rm

```
cp a.cpp b.cpp
mv old.cpp new.cpp
rm a.out
rm -r old_dir
```

特别小心 rm

命令行删除通常不经过回收站。做题只需认识命令；实际操作前必须确认目标路径。

在 Linux 中编译并运行 C++ 程序

```
g++ main.cpp -o main
./main
```

逐段理解

- g++: C++ 编译器驱动程序;
- main.cpp: 源代码文件;
- -o main: 把输出文件命名为 main;
- ./main: . 表示当前目录。

文件权限：读、写、执行

字母	英文	含义
r	read	读取
w	write	写入、修改
x	execute	执行

常见命令

`chmod +x main`：给文件增加执行权限。

区分

“存在可执行文件”与“拥有执行权限”是两件事；但竞赛环境中由 g++ 生成的程序通常已经可执行。

Linux 命令高频混淆表

命令	做什么	不做什么
ls	列出目录内容	不切换目录
cd	切换工作目录	不复制文件
cp	复制文件或目录	不改变当前目录
mkdir	创建目录	不创建普通源文件
mv	移动或重命名	不保留原副本

真题：CSP-S 2021 第一轮

在 Linux 终端中，用于列出当前目录下文件和子目录的命令是（　）。

A. `ls`
B. `cd`
C. `cp`
D. 以上都对

答案与辨析

A　`ls` 是 list；`cd` 切换目录；`cp` 复制文件。

来源：CSP-S 2021 第一轮第 1 题

真题：CSP-S 2022 第一轮

在 Linux 系统中，用于切换工作目录的命令是（　）。

A. `ls`
B. `cd`
C. `cp`
D. 以上都对

答案与辨析

B　`cd` 是 change directory 的缩写。

来源：CSP-S 2022 第一轮第 1 题

真题：CSP-S 2023 第一轮

在 Linux 系统中，用于创建目录的命令是（　）。

A. `newdir`
B. `mkdir`
C. `create`
D. `mkfold`

答案与辨析

B　`mkdir` 是 make directory 的缩写。

来源：CSP-S 2023 第一轮第 1 题