本节内容

IEEE 754 浮点数

表示范围
&
几种特殊状态

本节总览

- IEEE 754 浮点数的表示范围
  - 规格化浮点数的表示范围
  - 超出规格化浮点数表示范围怎么办?
  - 几种特殊状态：阶码全0、阶码全1

IEEE 754：规格化浮点数、特殊状态的浮点数

规格化浮点数解读方法：
- 符号：0正1负
- 阶码：真值 = 按无符号整数解读阶码，再减偏置值
- 尾数：小数点前面隐含一个 1

IEEE 754 标准规定：

仅当阶码不全为0、也不全为1时，表示这是一个“规格化浮点数”

阶码全为0、全为1 留作特殊用途，需要按照特殊的方式去解读真值

1 bit	8 bit	23 bit
符号	阶码	尾数

float：32位单精度浮点数

float 阶码偏置值 127

1 bit	11 bit	52 bit
符号	阶码	尾数

double：64位双精度浮点数

double 阶码偏置值 1023

特殊状态的浮点数（阶码全0 或 全1）

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
正零	0	全0	全0	+0	0	全0	全0	+0
负零	1	全0	全0	−0	1	全0	全0	−0
非规格化正数	0	全0	$f \neq 0$	$2^{-126}(0.f)$	0	全0	$f \neq 0$	$2^{-1022}(0.f)$
非规格化负数	1	全0	$f \neq 0$	$-2^{-126}(0.f)$	1	全0	$f \neq 0$	$-2^{-1022}(0.f)$
正无穷大	0	全1	全0	+∞	0	全1	全0	+∞
负无穷大	1	全1	全0	-∞	1	全1	全0	-∞
无定义数（非数）	0或1	全1	$f \neq 0$	NaN	0或1	全1	$f \neq 0$	NaN

阶码全为0、全为1 留作特殊用途，需要按照特殊的方式去解读真值

IEEE 754：规格化浮点数的表示范围

⚠注意：规格化浮点数的阶码不全为0、也不全为1

单精度 float 型

1 bit	8 bit	23 bit
符号 s	阶码 e	尾数 f

尾数 f → 规格化尾数 → 1.XXXXXXXXXXXXXXXXXXXXXXX

移码转真值：先按"无符号整数"解读阶码二进制串，再减掉偏置值，得到真值

先按无符号数解读阶码e，则取值范围为 1~254

减掉偏置值 127

阶码的真值取指范围为 -126 ~ 127

IEEE 754：规格化浮点数的表示范围

⚠注意：规格化浮点数的阶码不全为0、也不全为1

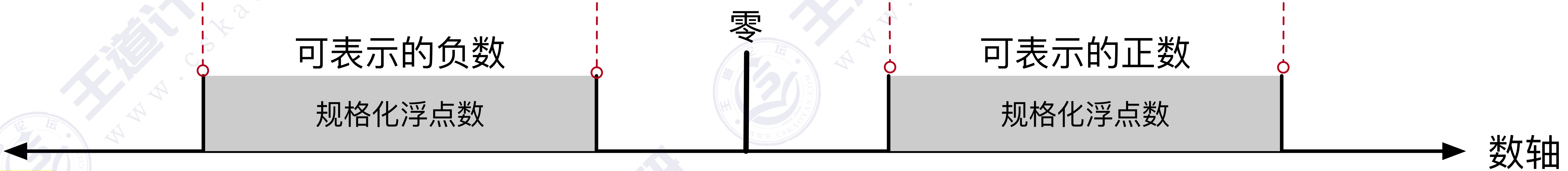
单精度 float 型

1 bit	8 bit	23 bit
符号 s	阶码 e	尾数 f

尾数 f → 规格化尾数 → 1.XXXXXXXXXXXXXXXXXXXXXXX

阶码的真值取指范围为 -126 ~ 127

【符,阶,尾】	1,11111110,11…11	1,00000001,00…00	0,00000001,00…00	0,11111110,11…11
二进制"科学计数法"	$-1.11111111111111111111111\times 2^{127}$	-1.0×2^{-126}	$+1.0\times 2^{-126}$	$+1.11111111111111111111111\times 2^{127}$
十进制→	$-(2-2^{-23})\times 2^{127}$	-1.0×2^{-126}	$+1.0\times 2^{-126}$	$+(2-2^{-23})\times 2^{127}$

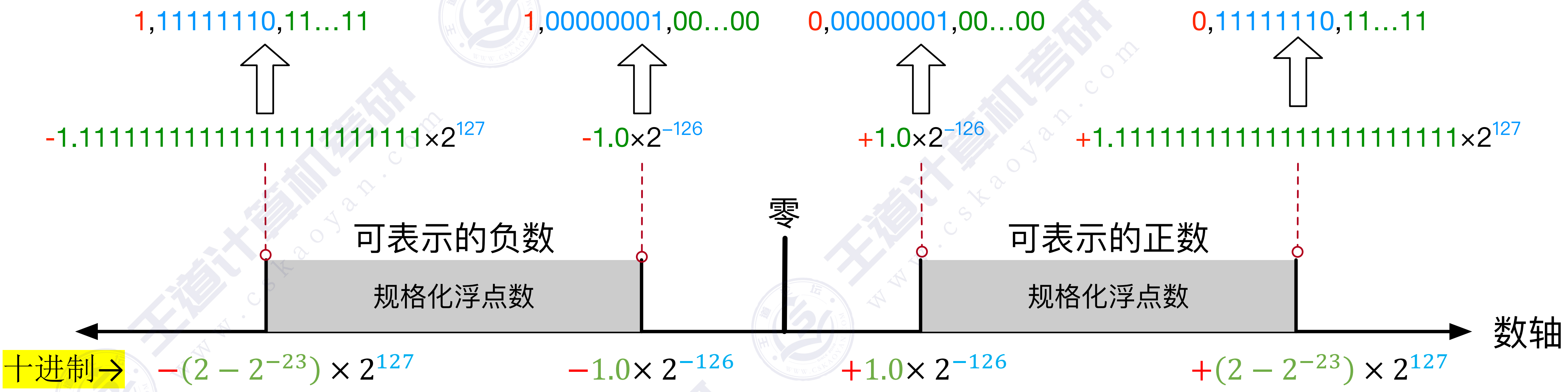


【真题训练】2018年真题_14

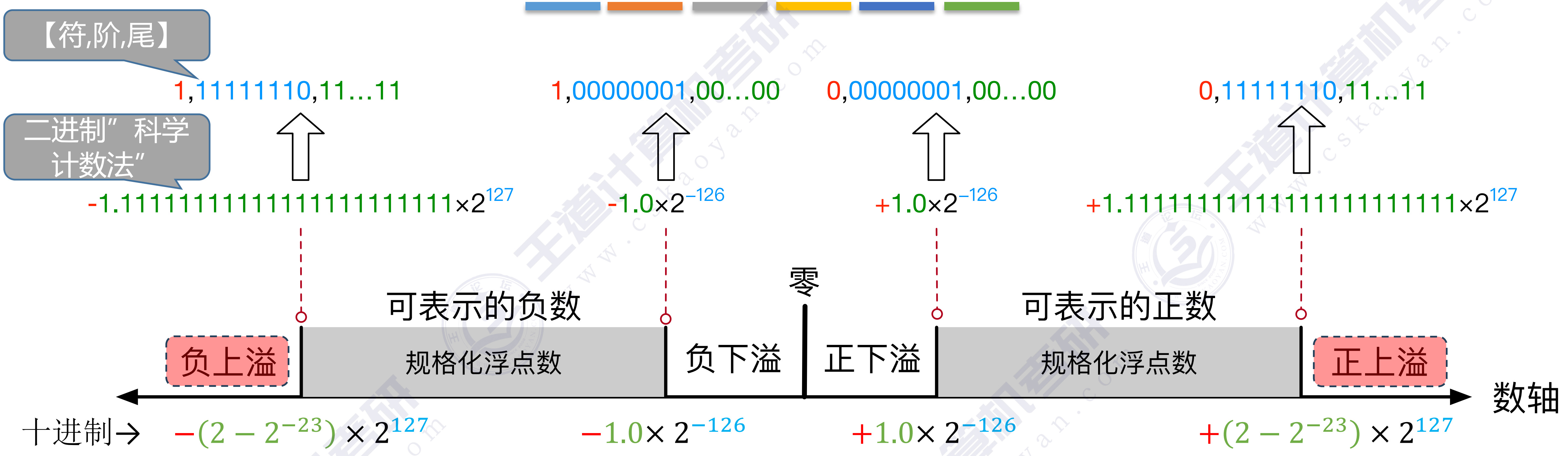
【2018真题_14】

14. IEEE 754 单精度浮点格式表示的数中，最小的规格化正数是（　　）。

$\boxed{\text{A. } 1.0\times2^{-126}}$　　B. 1.0×2^{-127}　　C. 1.0×2^{-128}　　D. 1.0×2^{-149}



IEEE 754：浮点数的上溢（Overflow）



运算结果大于最大规格化正数时称为正上溢，小于绝对值最大的规格化负数时称为负上溢。
正上溢、负上溢 统称上溢（Overflow），也会翻译为"溢出"

浮点数上溢（溢出）的处理：
① 浮点数运算部件将运算**结果设为设置为 +∞ 或 -∞**
② 设置浮点数溢出异常标志位（例如：x86会将浮点运算单元FPU的OE标志位 Overflow Exception 置为1）
注： IEEE 754 规定，默认不响应浮点数溢出异常，不中断程序；除非程序员手动开启此类异常响应

IEEE 754：无穷大的表示

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
正无穷大	0	全1	全0	$+\infty$	0	全1	全0	$+\infty$
负无穷大	1	全1	全0	$-\infty$	1	全1	全0	$-\infty$

float ⟹ $+\infty$　0 11111111 00000000000000000000000　0x7F800000

$-\infty$　1 11111111 00000000000000000000000　0xFF800000

double ⟹ ？

【程序示例】两个最大规格化正浮点数相加发生上溢

```c
#include <stdio.h>
#include <float.h>
#include <math.h>

int main() {
    // float 类型最大有限值（规格化正数）
    float max_float = FLT_MAX;
    printf("最大规格化正浮点数: %e\n", max_float);

    // 两个最大规格化正浮点数相加
    float result = max_float + max_float;
    printf("两个最大规格化正浮点数相加的结果: %e\n", result);

    // 检测是否为无穷大
    if (isinf(result)) {
        printf("结果为无穷大（发生了上溢）\n");
    } else {
        printf("结果是有限数\n");
    }

    return 0;
}
```

0,11111110,11…11

+∞ 0,11111111,00…00

程序员可手动检查浮点数运算结果是否溢出，也可选择忽视溢出

英汉 · 汉英词典

infinity | BrE ɪn'fɪnɪti, AmE ɪn'fɪnədi |

noun

① uncountable (boundlessness) 无限 wúxiàn

② uncountable (infinite distance) 无限远的距离 wúxiàn yuǎn de jùlí

- into infinity
  至无限远

③ uncountable Photography 无限远聚焦区 wúxiàn yuǎn jùjiāoqū

④ uncountable Mathematics 无穷大 wúqióngdà

- to infinity
  直至无穷

⑤ countable (huge, endless amount) 无限大的量 wúxiàndà de liàng

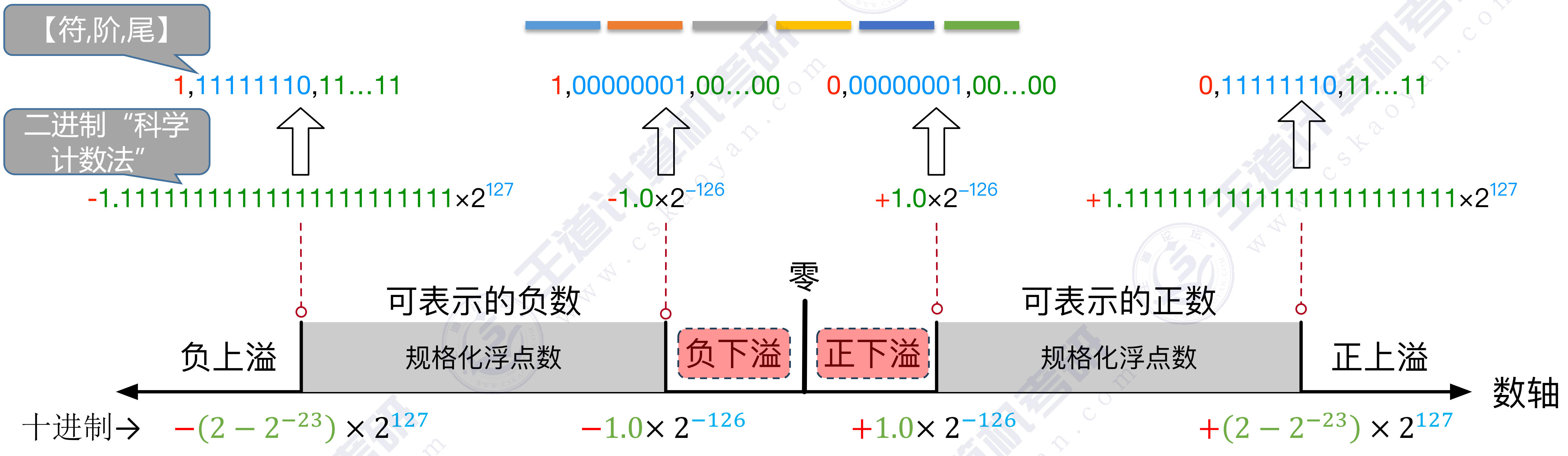
- an infinity of sth;
  数不清的某物

$+(2-2^{-23})\times 2^{127}$

运行结果

```
最大规格化正浮点数: 3.402823e+38
两个最大规格化正浮点数相加的结果: inf
结果为无穷大（发生了上溢）
```

IEEE 754： 浮点数的下溢（Underflow）



若浮点数运算结果在 0 至绝对值最小的规格化正数之间时称为正下溢，在 0 至绝对值最小规格化负数之间时称为负下溢。正下溢和负下溢统称下溢（Underflow）

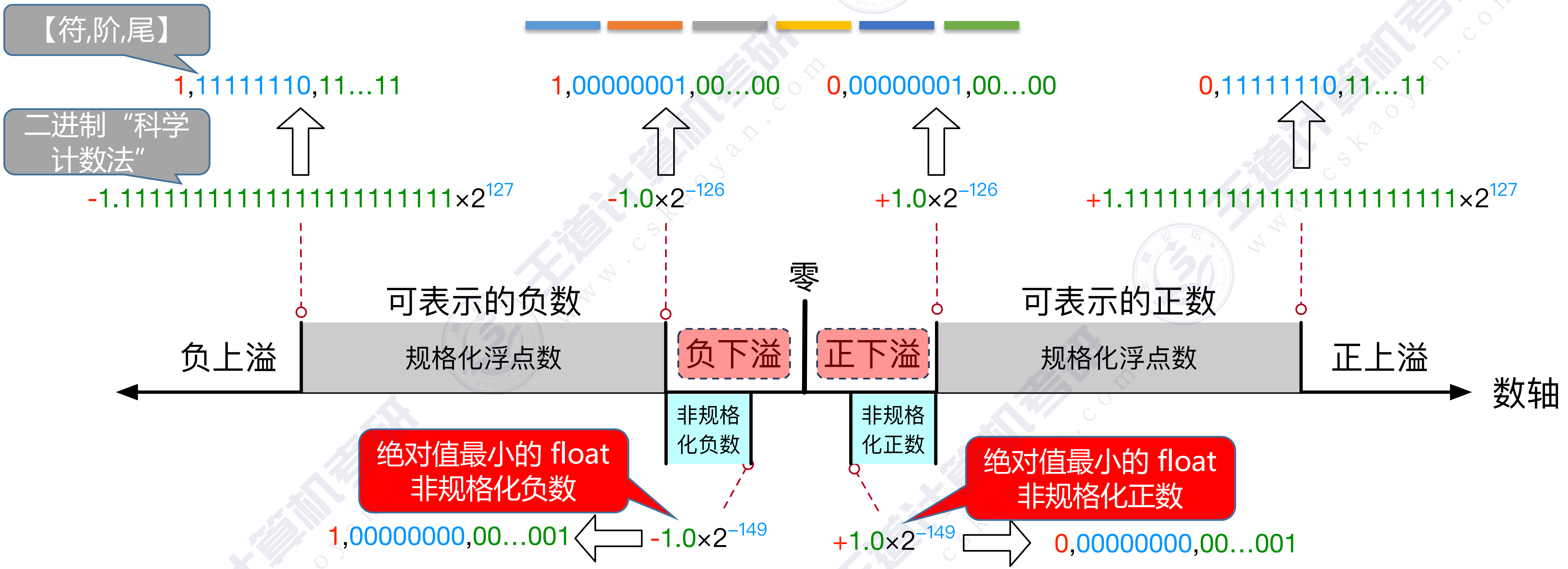
浮点数运算结果下溢时的处理：

① 若结果落入非规格化区间 → 用非规格化浮点数存储；若结果太小（真值逼近于0）→ 按机器零存储

② 若下溢至机器零，设置浮点数下溢异常标志位（例如：x86会将FPU的UE标志位 Underflow Exception 置1）

注： IEEE 754 规定，默认不响应浮点数下溢异常，不中断程序；除非程序员手动开启此类异常响应

IEEE 754：浮点数的下溢（Underflow）



浮点数运算结果下溢时的处理：

① 若结果落入非规格化区间 → 用非规格化浮点数存储；若结果太小（真值逼近于0）→ 按机器零存储

② 若下溢至机器零，设置浮点数下溢异常标志位（例如：x86会将FPU的UE标志位 Underflow Exception 置1）

注： IEEE 754 规定，默认不响应浮点数下溢异常，不中断程序；除非程序员手动开启此类异常响应

IEEE 754：真值0 的表示

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
正零	0	全0	全0	+0	0	全0	全0	+0
负零	1	全0	全0	−0	1	全0	全0	−0

float ⇒ +0 0 00000000 00000000000000000000000 0x00000000

-0 1 00000000 00000000000000000000000 0x80000000

double ⇒ ？

IEEE 754：非规格化浮点数的表示

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
非规格化正数	0	全0	$f \neq 0$	$2^{-126}(0.f)$	0	全0	$f \neq 0$	$2^{-1022}(0.f)$
非规格化负数	1	全0	$f \neq 0$	$-2^{-126}(0.f)$	1	全0	$f \neq 0$	$-2^{-1022}(0.f)$

不可以将阶码解读为 0 -127= -127，而是固定解读为阶码的最小真值 -126

尾数按照 **0.xxxxxx...xxx** 解读。即 **小数点前隐含0，而不是隐含1**

float 非规格化正数示例 ⟹ 0 00000000 01000000000000000000000 ⟹ 真值 = 0.01 × 2^{-126} → = 2^{-128}

绝对值最小的 float 非规格化正数 ⟹ 0 00000000 00000000000000000000001

真值 = 0. 00000000000000000000001 × 2^{-126} → = 2^{-149}

绝对值最大的 float 非规格化正数 ⟹ 0 00000000 11111111111111111111111

真值 = 0. 11111111111111111111111 × 2^{-126} → = $(1-2^{-23}) \times 2^{-126}$ = $2^{-126} - 2^{-149}$

IEEE 754：非规格化浮点数的表示

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
非规格化正数	0	全0	$f \neq 0$	$2^{-126}(0.f)$	0	全0	$f \neq 0$	$2^{-1022}(0.f)$
非规格化负数	1	全0	$f \neq 0$	$-2^{-126}(0.f)$	1	全0	$f \neq 0$	$-2^{-1022}(0.f)$

不可以将阶码解读为 0 -127= -127，而是固定解读为阶码的最小真值 -126

尾数按照 **0.xxxxxx...xxx** 解读。即 **小数点前隐含0，而不是隐含1**

float 非规格化负数示例 ⟹ 1 00000000 01000000000000000000000 ⟹ 真值 = - 0.01 × 2^{-126} → = - 2^{-128}

绝对值最小的 float 非规格化负数 ⟹ 1 00000000 00000000000000000000001

真值 = - 0. 00000000000000000000001 × 2^{-126} → = - 2^{-149}

绝对值最大的 float 非规格化负数 ⟹ 1 00000000 11111111111111111111111

真值 = - 0. 11111111111111111111111 × 2^{-126} → = - $(1-2^{-23}) \times 2^{-126}$ = - $(2^{-126} - 2^{-149})$

【真题训练】2023年真题_14

【2023真题_14】

14. 已知 float 型变量用 IEEE 754 单精度浮点数格式表示。若 float 型变量 x 的机器数为 8020 0000H，则 x 的值是（　）。

$\boxed{\text{A. } -2^{-128}}$　　B. -1.01×2^{-127}　　C. -1.01×2^{-126}　　D. 非数（NaN）

0x80200000 = 1000 0000 0010 0000 0000 0000 0000 0000 0000

= 1,00000000,01000000000000000000000

↑ 阶码全0　　↑ 尾数不全为0

非规格化浮点数真值 = $-\ 0.01 \times 2^{-126} = -\ 2^{-128}$

IEEE 754：非数（NaN）的表示

值的类型	单精度（32位）				双精度（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
无定义数（非数）	0或1	全1	$f \neq 0$	NaN	0或1	全1	$f \neq 0$	NaN

在 IEEE 754 浮点数标准中：**NaN** 代表“不是一个数”（Not a Number）

float NaN示例 ➡ 0 11111111 10000000001110000000001

sqrt 函数用于计算一个数的平方根
#include <math.h>

运算结果为 NaN 的典型例子：

- 0除以0　　0.0f / 0.0f　　// NaN
- 负数开根号　　sqrt(-1.0f)　　// NaN
- 无穷减无穷　　∞ - ∞　　// NaN

特别地：

- 非零数值除以0　　1.0f / 0.0f;　　// +∞
　　-1.0f / 0.0f;　　// -∞

IEEE 754 规定，非零数值除以0，结果是无穷，0除以0结果是NaN

【程序示例】运算结果为 NaN、∞ 的例子

```c
#include <stdio.h>
#include <math.h>

int main() {
    float a = 0.0f;
    float b = 0.0f;
    float result = a / b;

    printf("结果是: %f\n", result);

    if (isnan(result)) {
        printf("结果为 NaN\n");
    } else {
        printf("结果不是 NaN\n");
    }

    return 0;
}
```

判断是否为NaN

程序输出：
结果是：nan
结果为 NaN

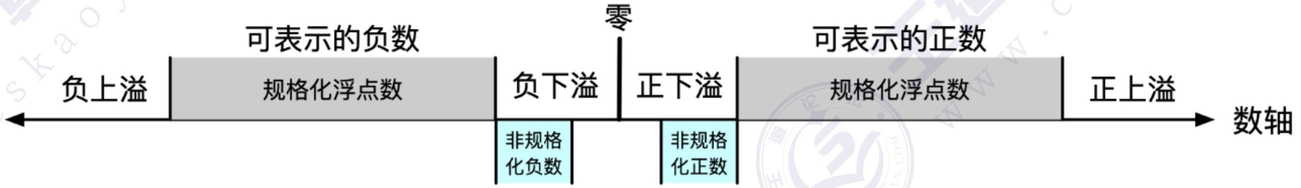

```c
#include <stdio.h>
#include <math.h>

int main() {
    float a = 1.0f;
    float b = 0.0f;
    float result = a / b;

    printf("结果是: %f\n", result);

    if (isinf(result)) {
        printf("结果为无穷大\n");
    } else {
        printf("结果是有限数\n");
    }

    return 0;
}
```

判断是否为 ∞

程序输出：
结果是：inf
结果为 无穷大

IEEE 754 浮点数的表示范围

- 浮点数的表示范围
  - 要点
    - 规格化浮点数
      - 特点：阶码不全为0、不全为1 — 阶码真值 = 按无符号数解读阶码 - 偏置值
      - 尾数的小数点前隐含1
    - 非规格化浮点数
      - 特点：阶码全0，尾数不全为0 — 阶码真值 = 1 - 偏置值
      - 尾数小数点前隐含0
    - 其他特殊状态 — ±0、±∞、NaN
  - 命题：推演各特殊值

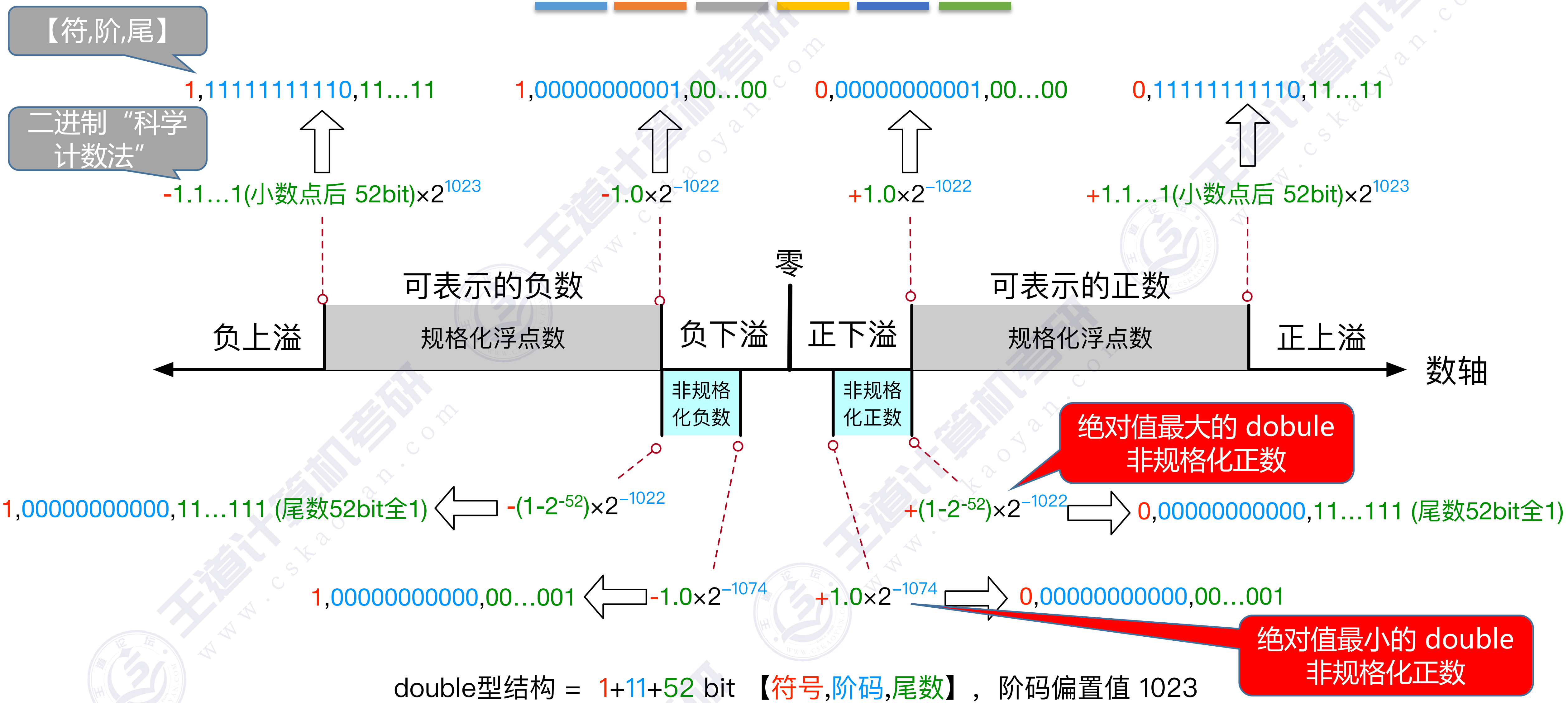


- 阶码全0、全1 的解释

值的类型	单精度 float（32位）				双精度 double（64位）			
	符号	阶码	尾数	值	符号	阶码	尾数	值
正零	0	全0	全0	+0	0	全0	全0	+0
负零	1	全0	全0	−0	1	全0	全0	−0
非规格化正数	0	全0	$f \neq 0$	$2^{-126}(0.f)$	0	全0	$f \neq 0$	$2^{-1022}(0.f)$
非规格化负数	1	全0	$f \neq 0$	$-2^{-126}(0.f)$	1	全0	$f \neq 0$	$-2^{-1022}(0.f)$
正无穷大	0	全1	全0	+∞	0	全1	全0	+∞
负无穷大	1	全1	全0	-∞	1	全1	全0	-∞
无定义数（非数）	0或1	全1	$f \neq 0$	NaN	0或1	全1	$f \neq 0$	NaN

- 超出规格化浮点数表示范围如何处理？
  - 上溢（溢出）的处理
    - ①将结果设置为 +∞ 或 -∞（视符号位而定）
    - ② 设置浮点数溢出异常标志位（IEEE 754 默认不响应浮点数溢出异常，不中断程序）
  - 下溢的处理
    - ① 若结果落入非规格化区间 → 用非规格化浮点数存储；若结果太小（真值逼近于0）→ 按机器零存储
    - ② 若下溢至机器零，设置浮点数下溢异常标志位（IEEE 754 默认不响应浮点数下溢异常，不中断程序）

【课后训练】推演 double 的表示范围



double型结构 = 1+11+52 bit 【符号,阶码,尾数】， 阶码偏置值 1023

【课后训练】写出 double 的±0、±∞、NaN

double型结构 = 1+11+52 bit 【符号,阶码,尾数】，阶码偏置值 1023

double ±0 ⟹	+0	0 00000000000 000……000 （52bit 0）	0x0000000000000000
	-0	1 00000000000 000……000 （52bit 0）	0x8000000000000000
double ± ∞ ⟹	+∞	0 11111111111 000……000 （52bit 0）	0x7FF0000000000000
	−∞	1 11111111111 000……000 （52bit 0）	0xFFF0000000000000
double NaN ⟹		0 11111111111 XXX……XXX （52bit不全为 0）	
		1 11111111111 XXX……XXX （52bit不全为 0）	