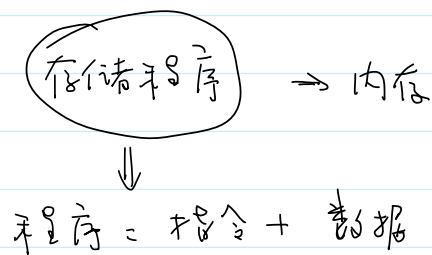
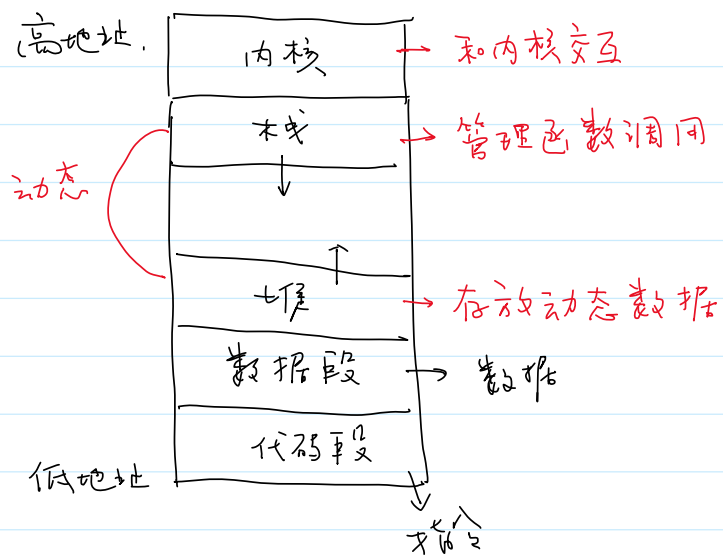


# 虚拟内存空间

2024年4月23日 9:39



变量要绑定一个值。

三要素。

变量名：引用绑定的值。

类型。

① 限定了值的范围。 { 编码  
内存的大小 → sizeof

② 值能够进行的操作

值

int a = 10;

# 输入输出模型

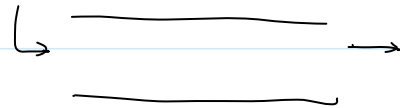
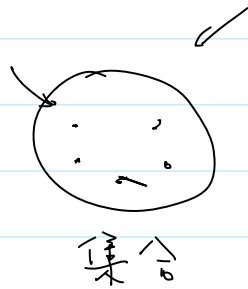
2024年4月23日 10:16

CPU. 内存, 高速缓存, TLB

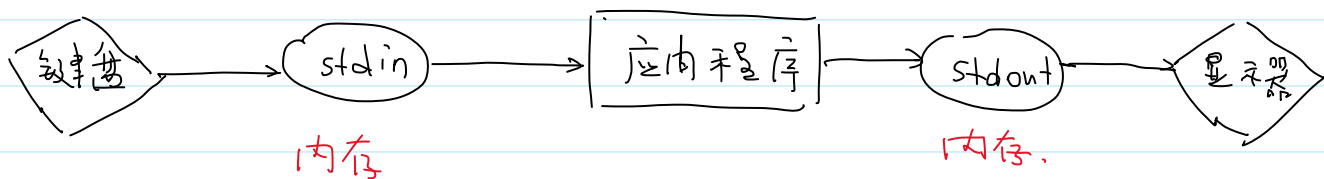
CPU. 外部设备, DMA

内存, 外部设备, 缓存.

缓冲区  $\Rightarrow$  内存.



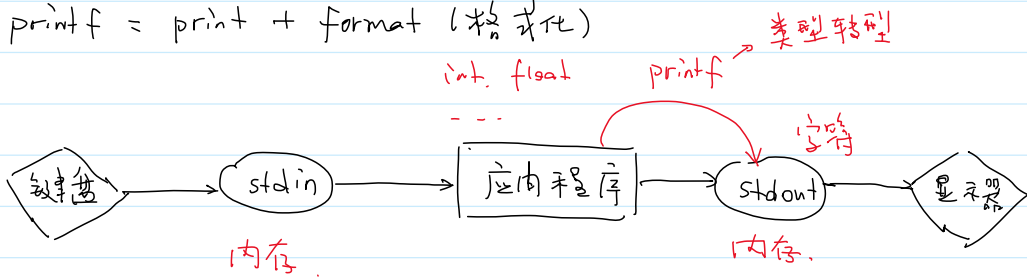
FIFO



# printf

2024年4月23日 10:51

printf = print + format (格式化)



// 格式串  
printf("i = %d, j = %d, x = %f, y = %f\n", i, j, x, y);  
return 0;

Microsoft Visual Studio 调试

i = 10, j = 20, x = 43.289200, y = 5527.000000

作用: 打印格式串中的内容, 并用后面表达式的值替换转换说明

格式串 { 普通字符: 原样输出  
          转换说明: 占位符,

↓  
格式: ① %m.pX  
      ② %~~m~~.pX  
          ↓  
      右边添空格.

X: 类型如何转换字符数据  
d. decimal  
f. float

m: 最小字段宽度.

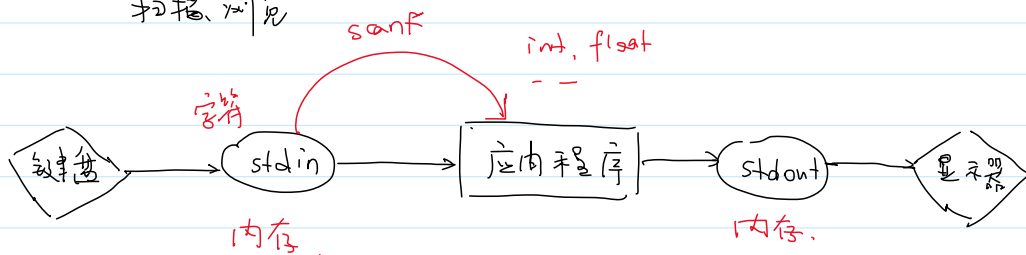
p: 精度 (precision)  
%d: 最少显示数字的个数  
%f: 显示小数的位数

```
< int i = 40;  
float x = 839.21f;  
  
printf("|%d|%5d|%5d|%5.0d|\n", i, i, i, i);  
printf("|%f|%10f|%10.2f|%-10.2f|\n", x, x, x, x);  
Microsoft Visual Studio 调试  
|40| 40|40 | 040|  
|839.210022|839.210022| 839.21|839.21 |
```

转换说明 { 类型转换. X  
              -m.p : 控制输出格式

scanf = scan + format

↓  
扫描/浏览



```

8 int main(void) {
9     int i, j;
10    float x, y;
11
12    scanf("%d%d%f", &i, &j, &x, &y);
13    return 0;
14 }
15

```

Handwritten annotations on the code: '格式串' (format string) points to the format string in line 12, and '位置' (position) points to the addresses in line 12.

名称	值
i	100
j	200
x	3.14000010
y	5.67000008

Terminal output: 100 200 3.14 5.67 \n

空格  
H, ', \n, \v  
%d: 忽略前置的空白字符。  
匹配一个十进制的整数。

%f: 忽略前置的空白字符  
匹配一个浮点数。

原理, 从左到右, 依次匹配格式串中的每一项。

stdin 中的字符

如果有一项匹配不成功, scanf 会立刻返回。

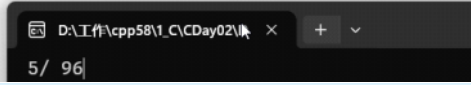
返回值: 匹配成功的转换说明的个数!

格式串 { 普通字符, 精确匹配  
空白字符, ' ', '\t', '\v', '\n' 匹配任意个空白字符。  
转换说明,

%d: 忽略前置的空白字符。  
匹配一个十进制的整数

%f: 忽略前置的空白字符  
匹配一个浮点数。

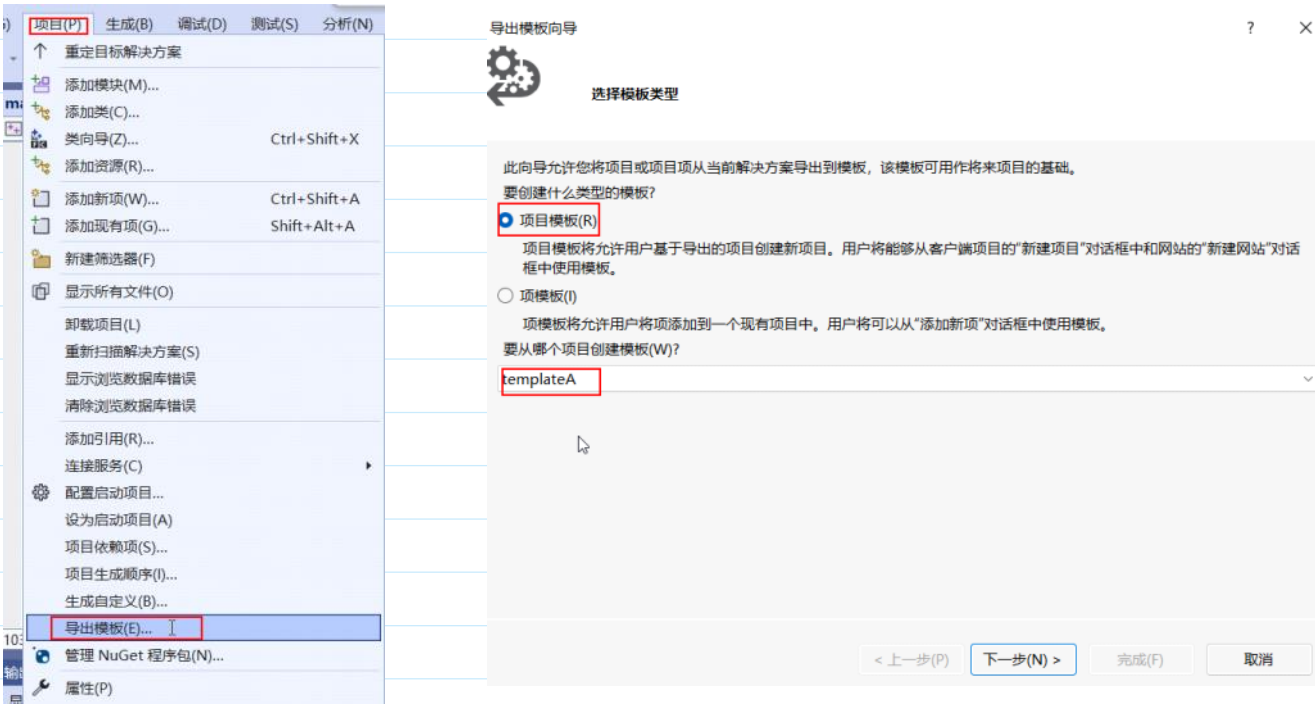
```
int i, j;  
scanf("%d/%d", &i, &j);
```



# 如何创建模板

2024年4月23日 14:31

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3
4  /*****
5  *                                COMMENT                                *
6  *****/
7
8  int main(void) {
9
10     return 0;
11 }
```



# 整数的编码

2024年4月23日 15:01

无符号整数 (二进制编码)

$b_{n-1} \dots b_1 b_0$  的值为  $b_{n-1}2^{n-1} + \dots + b_12^1 + b_02^0$ ,

Q: 如果一个无符号整数的内存表示为  $1010\_1010_{(2)}$ , 那么它的值是多少?

$$128 + 32 + 8 + 2 = 170$$

有符号整数 (补码)

二进制表示  $b_{n-1} \dots b_1 b_0$  的值为  $-b_{n-1}2^{n-1} + \dots + b_12^1 + b_02^0$ .

Q: 如果一个有符号整数的二进制表示为  $1010\_1010_{(2)}$ , 那么它的值是多少?

$$-128 + 32 + 8 + 2 = -86$$

两个性质:

A.  $\underbrace{11\dots1}_{n \text{ times}}_{(2)} \rightarrow -1$

$$-1 \times 2^{n-1} + 2^{n-2} + \dots + 2^1 + 2^0 = -1$$

B.  $x + (-x) = \underbrace{100\dots0}_{n \text{ times}}_{(2)}$

有符号整数  $11010100_{(2)}$ , 求它相反数的二进制表示.  
 $00101100_{(2)}$

Why: 为什么计算机采用补码存储有符号整数?

用加法器来做减法运算

$$a - b = a + (-b)$$

$$14 - 6 = 8$$

原码:

0000 1110

$\oplus 1000 0110$  (-6)

反码:

0000 1110

$\oplus 1111 1001$  (-6)



$$\begin{array}{r} \textcircled{+} \quad 10000110 \quad (-6) \\ \hline 10010100 \quad (-20) \end{array}$$

$$\begin{array}{r} \textcircled{+} \quad 11111001 \quad (-6) \\ \textcircled{-} \quad 10000011 \quad (7) \\ \hline \end{array}$$

补码:

$$\begin{array}{r} 00001110 \\ \textcircled{+} \quad 11111010 \quad (-6) \\ \hline \textcircled{-} \quad 10000100 \quad (8) \end{array}$$

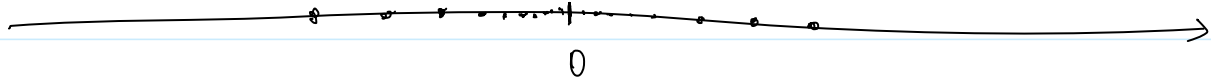
# 浮点数

2024年4月23日

15:43

编码: IEEE 754

→ 浮点数是不精确



float → %f

double → %lf

编码: ASCII (1字节, 低7位, 128个)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr
0	0	000	NUL (null)	32	20	040	Space	64	40	100	@	96	60	140	`
1	1	001	SOH (start of heading)	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX (start of text)	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX (end of text)	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT (end of transmission)	36	24	044	\$	68	44	104	D	100	64	144	d
5	5	005	ENQ (enquiry)	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK (acknowledge)	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL (bell)	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS (backspace)	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB (horizontal tab)	41	29	051	)	73	49	111	I	105	69	151	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	013	VT (vertical tab)	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	015	CR (carriage return)	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	016	SO (shift out)	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	017	SI (shift in)	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE (data link escape)	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1 (device control 1)	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2 (device control 2)	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3 (device control 3)	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4 (device control 4)	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN (synchronous idle)	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB (end of trans. block)	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN (cancel)	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM (end of medium)	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB (substitute)	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC (escape)	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS (file separator)	60	3C	074	<	92	5C	134	\	124	7C	174	
29	1D	035	GS (group separator)	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS (record separator)	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US (unit separator)	63	3F	077	?	95	5F	137	_	127	7F	177	DE

类型: ①限定值的取值  $\begin{cases} \rightarrow \text{编码, (ASCII)} \\ \rightarrow \text{内存大小, (t)} \end{cases}$

② 限定值能进行的操作

## 一. 表示值.

'a', 'A'

转义序列: ① 字符转义序列

Name	Escape Sequence	Name	Escape Sequence
Alert (bell)	\a	Vertical tab	\v
Backspace	\b	Backslash	\\
Form feed	\f	Question mark	\?
New line	\n	Single quote	\'
Carriage return	\r	Double quote	\"
Horizontal tab	\t		

## ② 数字转又序列

八进制, (最多3个) '\0'

十六进制, '\xABC'

## 二、支持哪些操作。

C语言将 char 当作一个字节的整数来处理。

<ctype.h>

```
// 大小写转换函数
int tolower(int c);
int toupper(int c);
```

→ 扩展了字符类型支持的操作。

<https://zh.cppreference.com/w/c/string/byte>

ASCII 值			字符	isctrl	isprint	isspace	isblank	isgraph	ispunct	isalnum	isalpha	isupper	islower	isdigit	isxdigit
十进制	十六进制	八进制		isctrl	isprint	isspace	isblank	isgraph	ispunct	isalnum	isalpha	isupper	islower	isdigit	isxdigit
0-8	\x0-\x8	\0-\10	控制码 (NUL 等)	≠0	0	0	0	0	0	0	0	0	0	0	0
9	\x9	\11	制表符 (\t)	≠0	0	≠0	≠0	0	0	0	0	0	0	0	0
10-13	\xA-\xD	\12-\15	空白符 (\n, \v, \f, \r)	≠0	0	≠0	0	0	0	0	0	0	0	0	0
14-31	\xE-\x1F	\16-\37	控制码	≠0	0	0	0	0	0	0	0	0	0	0	0
32	\x20	\40	空格	0	≠0	≠0	≠0	0	0	0	0	0	0	0	0
33-47	\x21-\x2F	\57	!"#\$%&'()*+,-./	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
48-57	\x30-\x39	\71	0123456789	0	≠0	0	0	≠0	0	≠0	0	0	0	≠0	≠0
58-64	\x3A-\x40	\100	;<=>?@	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
65-70	\x41-\x46	\101-\106	A B C D E F	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	≠0
71-90	\x47-\x5A	\107-\132	G H I J K L M N O P Q R S T U V W X Y Z	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	0
91-96	\x5B-\x5F	\133-\140	[ \ ] ^ _ `	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
97-102	\x61-\x66	\141-\146	a b c d e f	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	≠0
103-122	\x67-\x7A	\147-\172	g h i j k l m n o p q r s t u v w x y z	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	0
123-126	\x7B-\x7E	\173-\176	{   } ~	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
127	\x7F	\177	退格符 (DEL)	≠0	0	0	0	0	0	0	0	0	0	0	0

## 三、读/写 (和用户交互)

printf + %c.

scanf + %c.

└→ 匹配一个字符。(不会忽略空白字符)

└→ 匹配一个字符, (不会忽略空白字符)

Q: 跳过空白字符, 读取下一个非空白字符.

" %c "

```
putchar(c);  
c = getchar();
```

语言. ① 基本语法  
② 惯用法 (成语)  
③ 设计模式.









↗ 跳过这一行剩余的字符

```
// idiom  
while (getchar() != '\n') /* skips rest of line */  
(i)
```

## 类型转换

2024年4月23日 16:59

### 隐式转换

 C + C	130	int
 C + S	165	int
 S + S	200	int
 S + i	1100	int
 i + i	11000	long
 l + ll	1010000	_int64
 ll + f	1000003.12	float
 f + d	8.8100001049041747	double

A. 整数提升 (int)

B. 值的表示范围: 小 → 大  
int → long → long long → float → double → long double

C. 同一整数转换等级.

有符号 → 无符号

```
int i = -1;  
unsigned int u = 100;
```

```
if (u > i) {  
    printf("Greater!\n");  
} else {  
    printf("Less!\n");  
}
```

```
D:\工作\cpp58\1_C\CDay02\D  X + v  
Less!
```



教训: 千万不要将无符号整数和有符号整数混合运算,

### 强制转换

1. 我们可以使用强制类型转换计算浮点数的小数部分, 如:

```
float f, frac_part;  
I  
frac_part = f - (int) f;
```

2. 使用强制类型转换显示表明肯定会发生的转换。

```
i = (int) f; /* f is converted to int */
```

3. 使用强制类型转换进行我们需要的类型转换。

```
float quotient;  
int dividend, divisor;  
  
/* What's the difference between next two expressions? */  
quotient = dividend / divisor;  
quotient = (float) dividend / divisor;
```

4. 有时候, 我们还可以使用强制类型转换来避免溢出。

```
long long millisPerDay = 24 * 60 * 60 * 1000;  
long long nanosPerDay = 24 * 60 * 60 * 1000 * 1000 * 1000; /* overflow */  
  
printf("%lld\n", nanosPerDay / millisPerDay);
```

# 定义别名

2024年4月23日 17:29

格式: typedef 类型 别名;

How

Why?

① 可读性增强.

② 可移植性增强 (代码)

# sizeof运算符

2024年4月23日 17:38

作用：计算某一类型的值，在内存中所占字节的长度。

sizeof(c1 + c2)	4	unsigned int
sizeof(c1)	1	unsigned int
sizeof c1	1	unsigned int
sizeof(int)	4	unsigned int
sizeof(long long int)	8	unsigned int