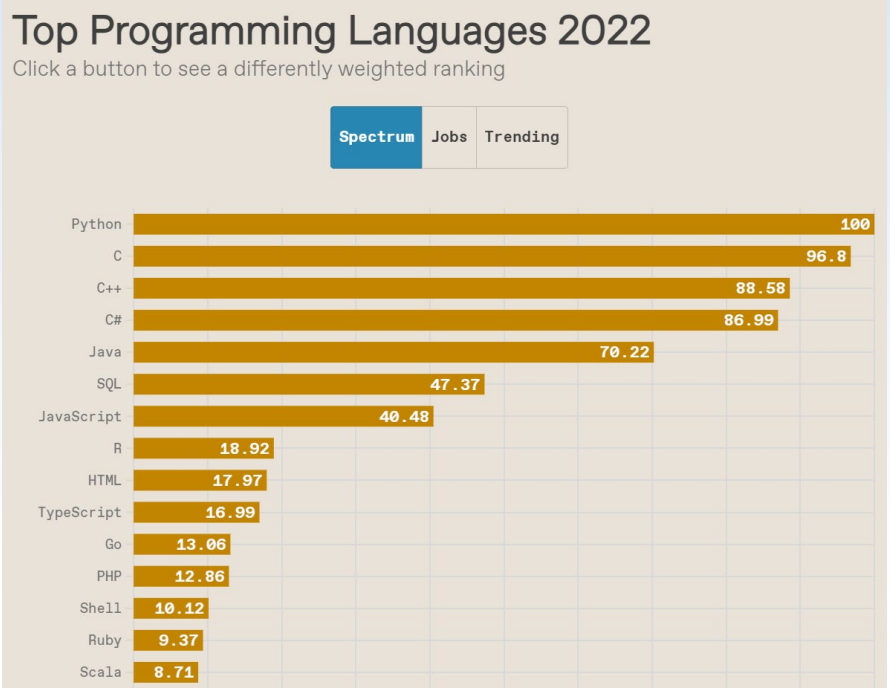
C语言系统级编程

荣文戈

w.rong@buaa.edu.cn

北京航空航天大学计算机学院

C语言一直是最流行的基础编程语言之一



IEEE Spectrum

Very Long Term History

To see the bigger picture, please find below the positions of the top 10 programming languages of many years back. Please note that these are *average* positions for a period of 12 months.

Programming Language	2021	2016	2011	2006	2001	1996	1991	1986
C	1	2	2	2	1	1	1	1
Python	2	5	6	8	25	25	-	-
Java	3	1	1	1	2	16	-	-
C++	4	3	3	3	3	2	2	6
C#	5	4	5	7	12	-	-	-
Visual Basic	6	13	-	-	-	-	-	-
JavaScript	7	7	10	9	9	20	-	-
PHP	8	6	4	4	10	-	-	-
Assembly language	9	11	-	-	-	-	-	-
SQL	10	-	-	-	37	-	-	-
Ada	29	28	17	16	18	7	5	2
Lisp	33	27	13	13	17	8	4	3
Pascal	268	75	15	17	15	5	3	7
(Visual) Basic	-	-	7	5	4	3	8	5

TIOBE Index

C语言广泛应用在航空航天等国民经济重要领域

为什么要开这门课？

基础不牢，地动山摇

先问一个基本概念

int a=1;	a叫做什么？	变量？
const int a=1;	a叫做什么？	常量？
const volatile int a=1;	a叫做什么？	？
int b[10]={0};	b叫做什么？	？

C语言里面真的有变量吗？常量到底有哪些？

先提几个问题

int a = 10;

printf(“a = %d\n”, a);
printf(“&a = %d\n”, &a);

&a返回值什么类型？

int *p = &a;

先提几个问题

```
int a[3] = {10};

printf("a[0] = %d\n", a[0]);

printf("a = %d\n", a);
printf("&a = %d\n", &a);
```

打印a和&a的两个语句打印出来的值一样吗？

```
p = a;
q = &a;
```

p和q该如何定义和声明？

先提几个问题

```
int a[3][4] = {10};

printf("a[0][0] = %d\n", a[0][0]);

printf("a = %d\n", a);
printf("&a = %d\n", &a);
printf("a[0] = %d\n", a[0]);
```

打印a、&a和a[0]的三个语句打印出来的值一样吗？

p = a; q = &a; r = a[0]， p、q、r应该如何定义和声明？

先提几个问题

```
int a[3][4][5][6] = {10};

printf("a[0][0][0][0] = %d\n", a[0][0][0][0]);

printf("a = %d\n", a);
printf("&a = %d\n", &a);
printf("a[0] = %d\n", a[0]);
printf("a[0][0] = %d\n", a[0][0]);
printf("a[0][0][0] = %d\n", a[0][0][0]);
```

打印a、&a、a[0]、a[0][0]和a[0][0][0]的五个语句打印出来的值一样吗？

p = a; q = &a; r = a[0]; s = a[0][0]; t = a[0][0][0]，p、q、r、s和t应该如何定义和声明？

先提几个问题

```
void func( int a[][4] )
{
   // do something
}

int main()
{
   int a[3][4];

   func(a);

   return 0;
}
```

第一维大小为什么没了？

先提几个问题

```
int a = 10;

printf("sizeof(a) = %d\n", sizeof(a));
printf("sizeof(a+1-1) = %d\n", sizeof(a+1-1));
```

sizeof(a)和sizeof(a+1-1)打印出来的值一样
背后计算大小的逻辑一样吗？

先提几个问题

```
int a[3] = {10};

printf("sizeof(a) = %d\n", sizeof(a));
printf("sizeof(a+1-1) = %d\n", sizeof(a+1-1));
```

sizeof(a)和sizeof(a+1-1)打印出来的值不一样
是什么原因呢？

先提几个问题

```
int a = 10;

printf("sizeof(a++) = %d\n", sizeof(a++));
```

现在a等于几了？

先提几个问题

```
int a = 3;
a += a -= a * a;

printf("a = %d\n", a);
```

现在a等于几了？

a等于几都可以，这是未定义行为

最后再问一个问题

C语言中有对象（Object）吗？

有！！！

不仅有，而且是理解C语言的最核心基础概念

本课程内容简介

1、语法类知识：

控制结构（if、for、while、switch）、输入输出、文件等等

2、概念类知识：

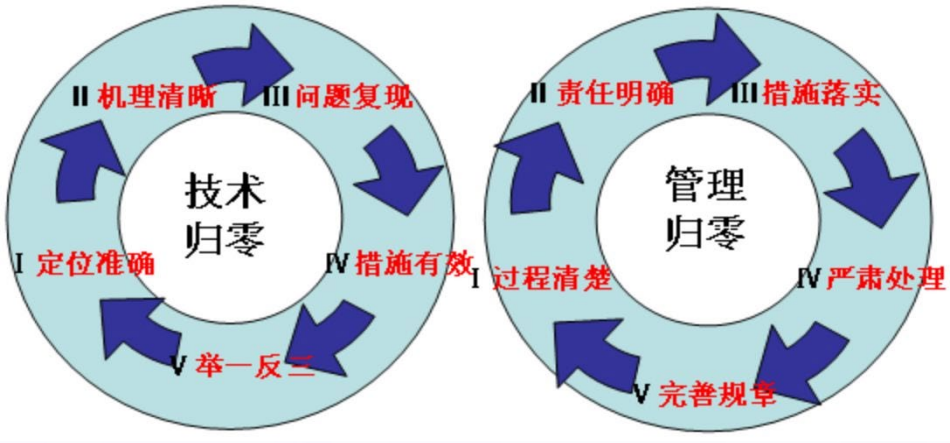
对象、左值、表达式、数组、指针、Name Space、Scope等等

3、策略类知识：

调试、函数构造、架构设计等等

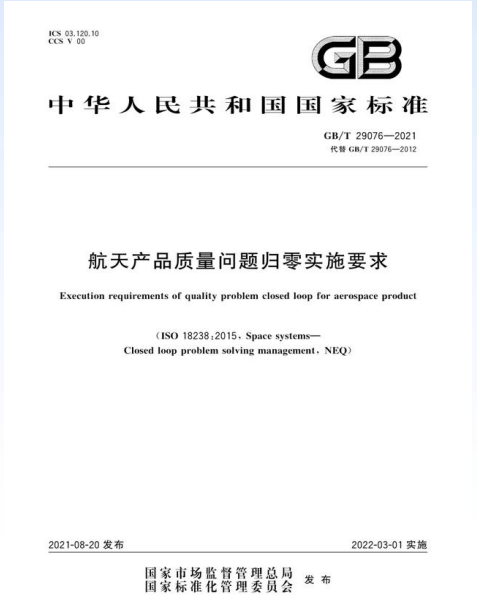
深入了解内存（系统级）

机理清晰是质量保证的核心之一



定位准确是前提
机理清楚是关键
问题复现是手段
措施有效是核心
举一反三是延伸

过程清楚是基础
责任明确是前提
措施落实是核心
严肃处理是手段
完善规章是结果



ICS 03.120.10
CCS V 00

GB

中华人民共和国国家标准

GB/T 29076—2021
代替 GB/T 29076—2012

航天产品质量问题归零实施要求

Execution requirements of quality problem closed loop for aerospace product

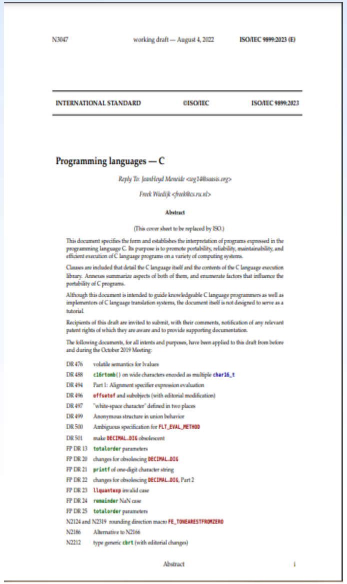
(ISO 18238:2015, Space systems—
Closed loop problem solving management, NEQ)

2021-08-20 发布　　2022-03-01 实施

国家市场监督管理总局
国家标准化管理委员会　发布

该标准是2021年8月20日发布，
2022年3月1日实施的一项中国国家标准。

本课程参考书



N3047 working draft — August 4, 2022 ISO/IEC 9899:2023 (E)

INTERNATIONAL STANDARD ©ISO/IEC ISO/IEC 9899:2023

Programming languages — C

Reply To: JeanHeyd Meneide <wg14@soasis.org>

Freek Wiedijk <freek@cs.ru.nl>

Abstract

(This cover sheet to be replaced by ISO.)

This document specifies the form and establishes the interpretation of programs expressed in the programming language C. Its purpose is to promote portability, reliability, maintainability, and efficient execution of C language programs on a variety of computing systems.

Clauses are included that detail the C language itself and the contents of the C language execution library. Annexes summarize aspects of both of them, and enumerate factors that influence the portability of C programs.

Although this document is intended to guide knowledgeable C language programmers as well as implementors of C language translation systems, the document itself is not designed to serve as a tutorial.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

The following documents, for all intents and purposes, have been applied to this draft from before and during the October 2019 Meeting:

DR 476	volatile semantics for lvalues
DR 488	c16rtomb() on wide characters encoded as multiple char16_t
DR 494	Part 1: Alignment specifier expression evaluation
DR 496	offsetof and subobjects (with editorial modification)
DR 497	"white-space character" defined in two places
DR 499	Anonymous structure in union behavior
DR 500	Ambiguous specification for FLT_EVAL_METHOD
DR 501	make DECIMAL_DIG obsolescent
FP DR 13	totalorder parameters
FP DR 20	changes for obsolescing DECIMAL_DIG
FP DR 21	printf of one-digit character string
FP DR 22	changes for obsolescing DECIMAL_DIG, Part 2
FP DR 23	llquantexp invalid case
FP DR 24	remainder NaN case
FP DR 25	totalorder parameters
N2124 and N2319	rounding direction macro FE_TONEARESTFROMZERO
N2186	Alternative to N2166
N2212	type generic cbrt (with editorial changes)

Abstract i

ISO/IEC JTC1/WG14 C语言标准文本

2023年4月2日发布的Working Draft

https://www.open-std.org/jtc1/sc22/wg14/www/docs/n3096.pdf

一切疑惑都可以从这个标准中找到答案

希望这是一次有趣的体验

学习周期和考核方法

学习时长1-9周（16学时，1学分）

考核方法：考试（开卷，带什么资料都行）+

随堂测验（上课的时候带张草稿纸）

欢迎任何有兴趣的同学来旁听！！！

内存基本概念简单回顾

bit（比特）
状态0/1
二进制

high-order bit　　low-order bit

8个bit拼在一起，组成一个byte（字节）
存储范围00000000~11111111

二进制转十进制

0~255

1 byte一定等于8 bit？

1个byte一定等于8个bit吗？

<limits.h>定义了一个宏CHAR_BIT

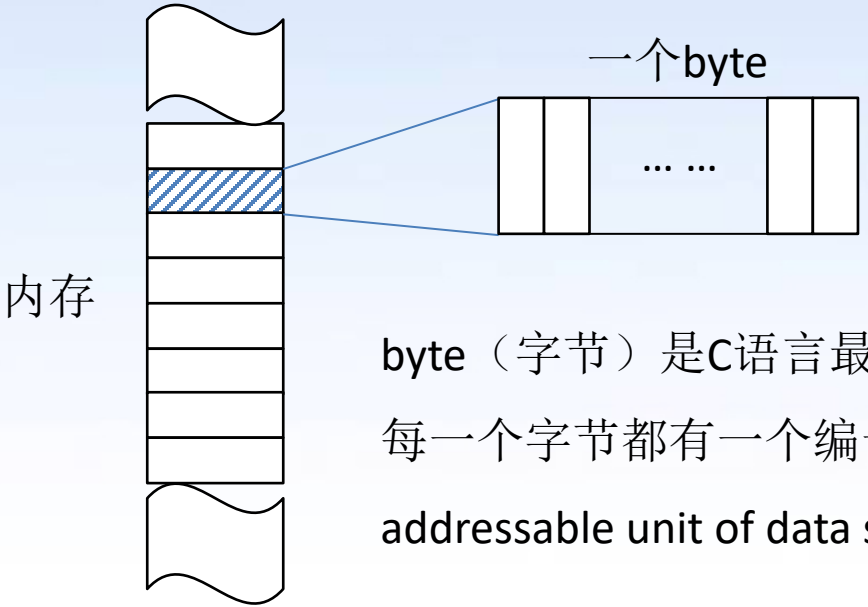
C语言中规定一个byte等于CHAR_BIT个bit

A byte contains CHAR_BIT bits

C语言的标准规定CHAR_BIT>=8

1个Byte包括至少（而不是正好）8个bit

内存长什么样？



byte（字节）是C语言最重要的基础存储单元

每一个字节都有一个编号，也就是地址

addressable unit of data storage…

内存就是一系列byte顺序排列在一起组成的存储结构

我们机器里的内存有多少个这样的byte呢？

我们说8/16/32/64位机到底是什么意思？

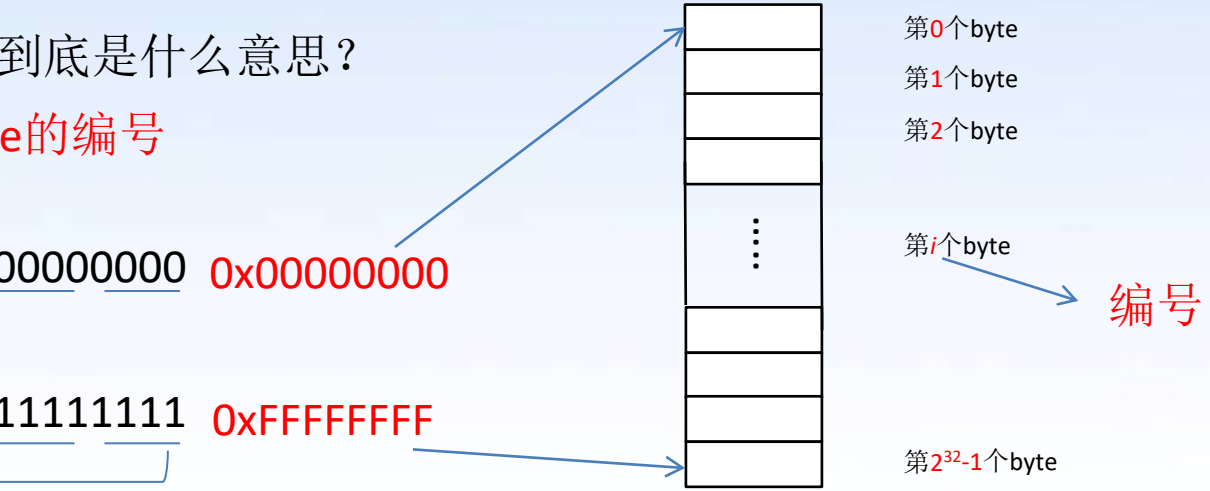
32位bit来表示byte的编号

00000000000000000000000000000000 0x00000000

……

11111111111111111111111111111111 0xFFFFFFFF

32位



可以编号的byte个数一共可以有2^{32}个，$2^{32}=2^2*2^{10}*2^{10}*2^{10}$

4G内存，其中1K=1024Byte，1M=1024K, 1G=1024M

测试题

1、32位机最多能访问4G内存的原因是什么？

2、对64位机器来说，最多能访问的byte数量有多少？

答案：

1、32位机最多能访问的内存byte总数为2^{32}个，因此:

$$2^{32} \div 1024 = 2^{22}\text{K}，\ \text{1K=1024个字节}$$

$$2^{22}\text{K} \div 1024 = 2^{12}\text{M}，\ \text{1M=1024K字节}$$

$$2^{12}\text{M} \div 1024 = 2^{2}\text{G} = 4\text{G}，\ \text{1G=1024M字节}$$

2、64位机最多能访问的byte总数为2^{64}个（理论上）

C语言中最重要的概念：对象

1、对象的概念

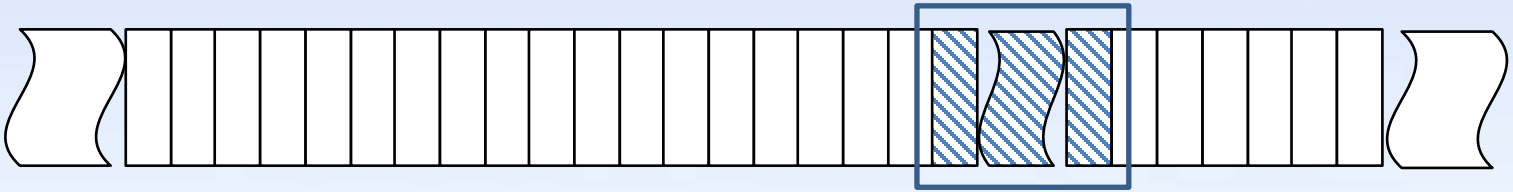
2、如何刻画对象的属性

3、理解对象类型
（非数组 vs. 数组类型，Referenced Type vs Pointer Type）

4、理解指针和数组也是一种普通的对象类型

本课程核心概念就是对象（Object）

对象是什么



region of data storage

objects are composed of contiguous sequences of one or more bytes

对象是怎么分配出来的呢？

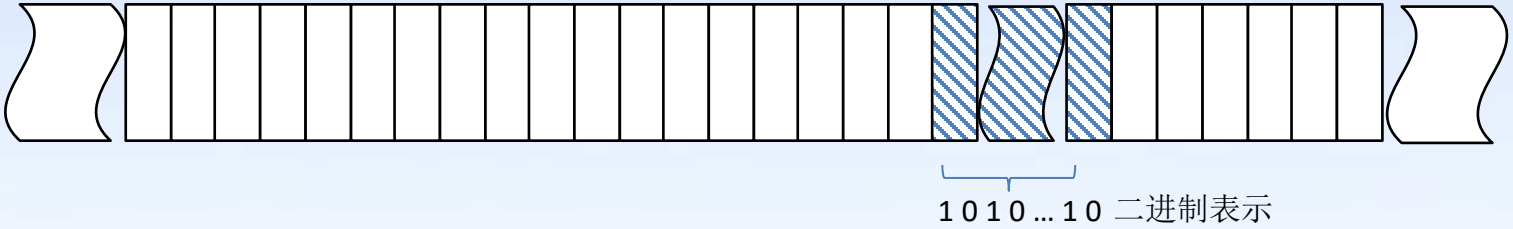
1、对象声明，例如int a;

2、内存管理函数，例如malloc(4);

3、声明String Literal，”hello”;

4、声明Compound Literal，(int){1}, (int[2]){1,2};

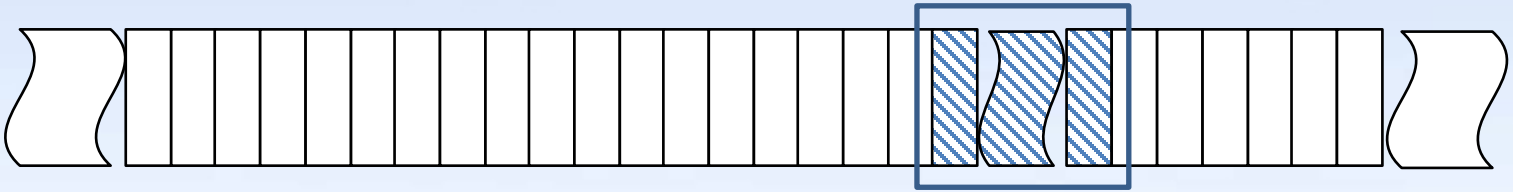
对象表示（Object Representation）



这个对象各个bit组成的二进制串就是这个对象的对象表示(Object Representation)

理解对表示之前，我们需要了解对象如何刻画

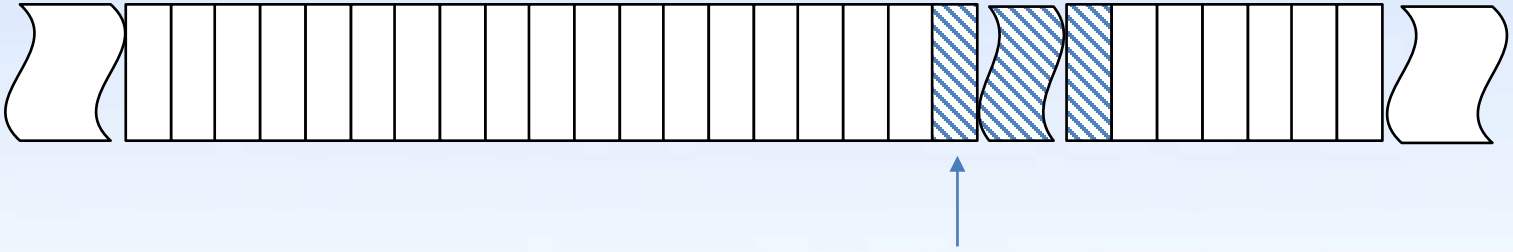
对象如何刻画？



Object = <Address, Object Type, Size, Name, Value, Value Type, Alignment>

首地址、对象类型、大小、名称、值、值类型、对齐要求

刻画对象的属性：首地址

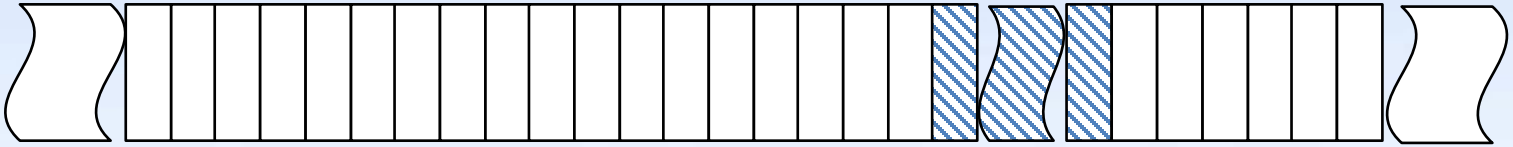


对象所占用的连续字节中lowest字节的编号

分配对象地址需要有对齐（Alignment）的要求

对象的地址是系统分配的

刻画对象的属性：对象类型



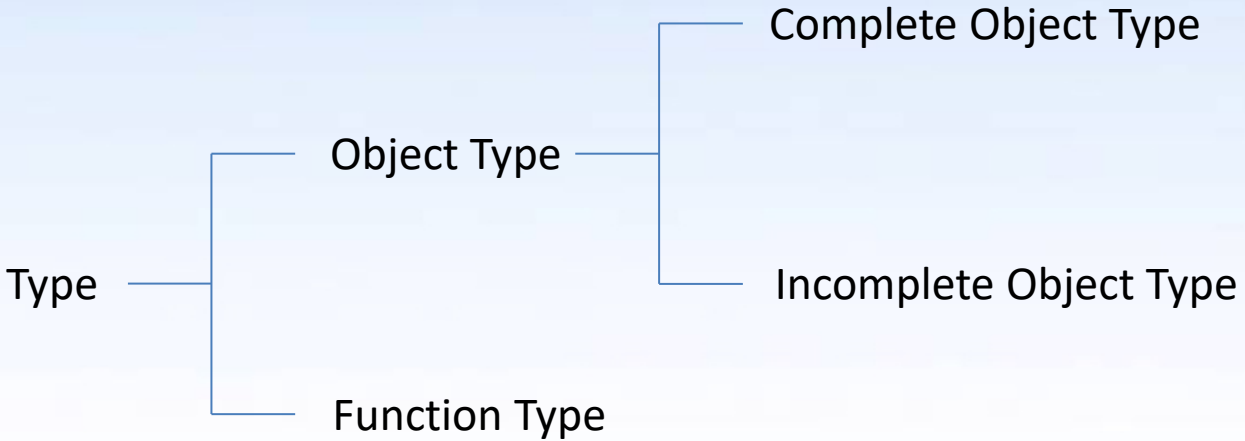
当我们试图理解对象，我们需要了解这个对象的对象类型

对象类型主要包括：完全对象类型和不完全对象类型

常见的完全对象类型，例如int, short, long, float, double ...

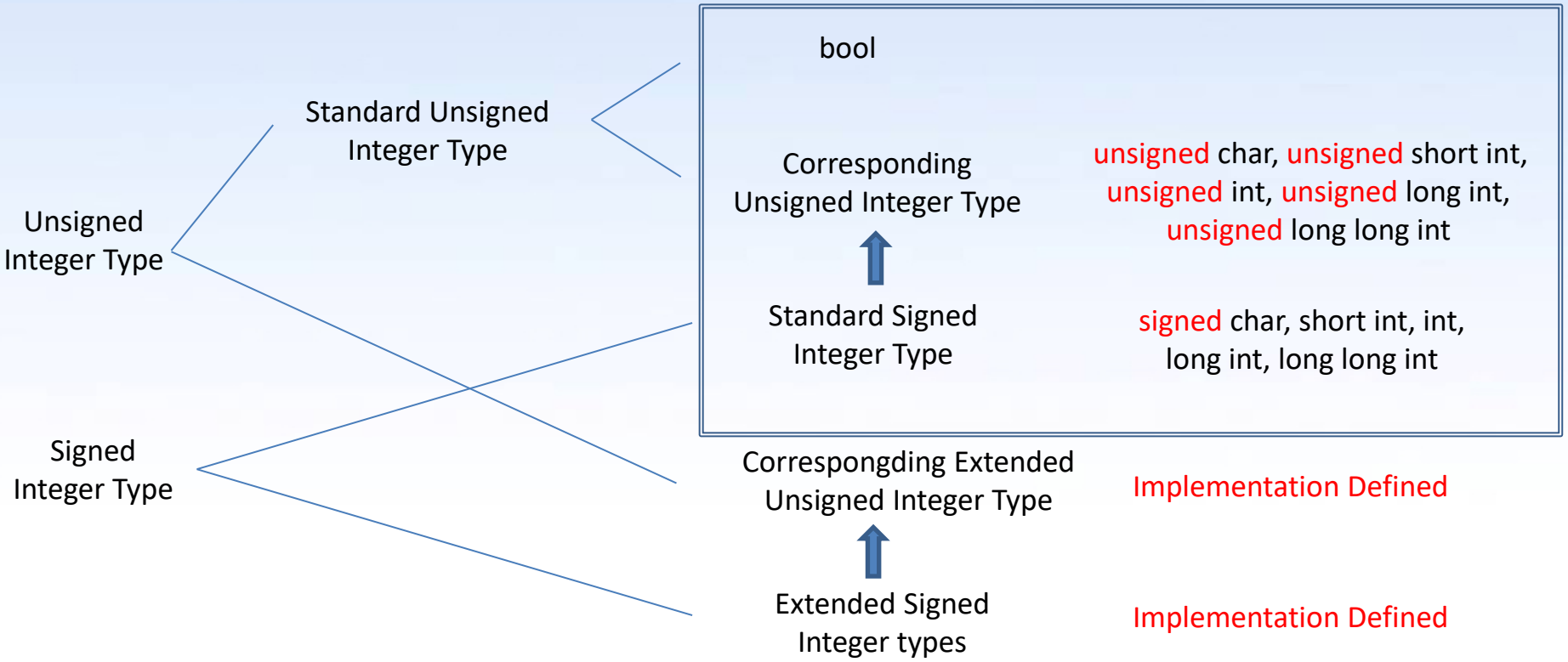
Type（类型）

Type(类型)是用来理解一个对象或者函数返回值语义的重要依据

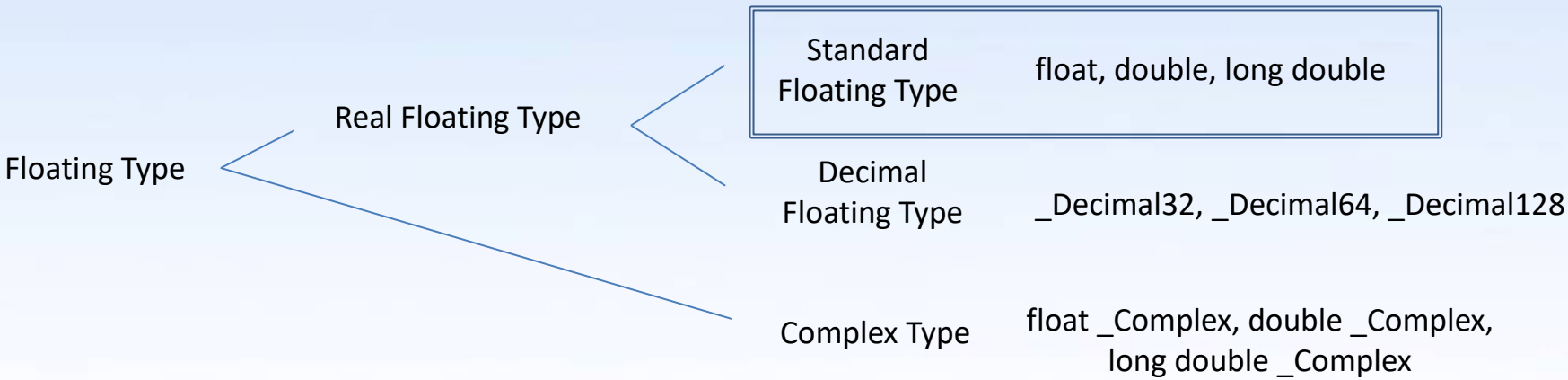


Object: Region of data storage in the execution environment, the contents of which can represent values

Object Type: Integer Related

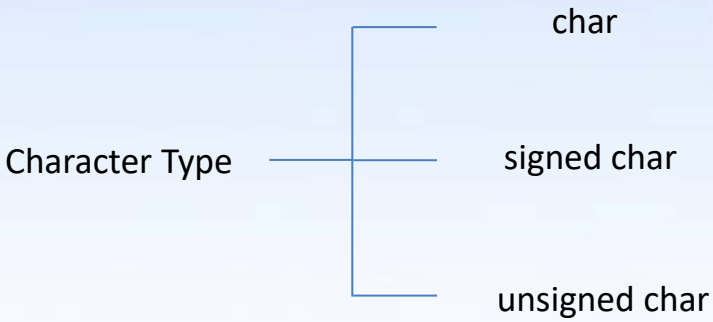


Object Type: Floating Related



Complex types are a conditional feature that implementations need not support

Object Type: Character Related



平时最常用的char，到底是unsigned还是signed不同平台表现不同

对于char类型来说，只用于存储ASCII字符比较安全

Object Type: Enumerated Type

enum WeekDay {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday};

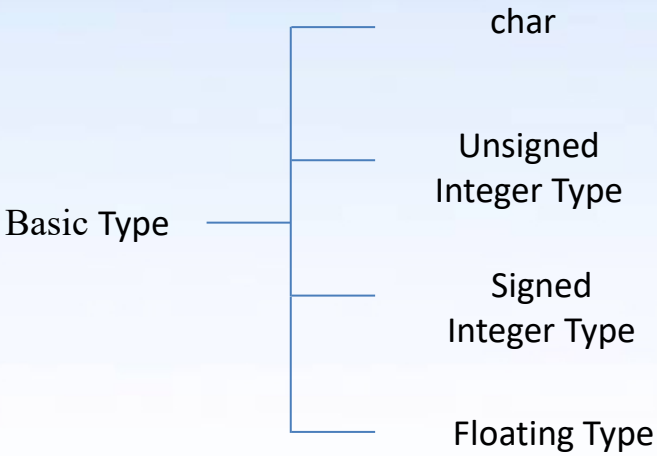
enum WeekDay d = Monday;

printf("%d\n", Thursday);

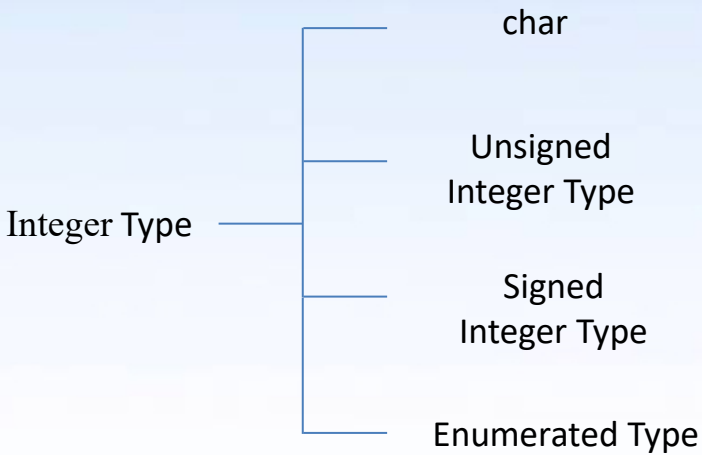
结果为3

All enumerations have an underlying type.
The underlying type is either char or a standard or extended signed or unsigned integer type

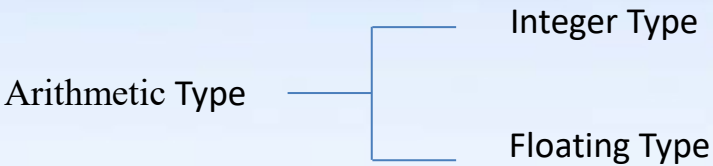
Object Type: Basic Type



Object Type: Integer Type



Object Type: Arithmetic Type



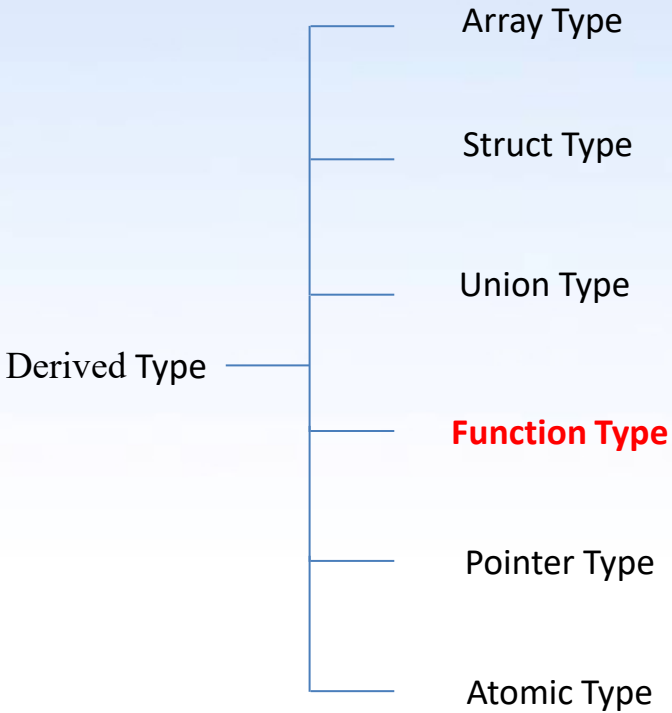
Object Type: Basic Type

Arithmetic Type	Integer Type	char		
		Signed Integer Type	Standard Signed Integer Type	singed char signed short int signed int signed long int signed long long
			bit-precise Signed Integer Type	_BitInt(N)
			Extended Signed Integer Type	Implementation Defined
		Unsigned Integer Type	Standard Unsigned Integer Type	bool unsigned char unsigned short int unsigned int unsigned long int unsigned long long
			bit-precise Unsigned Integer Type	unsigned _BitInt(N)
			Extended Unsigned Integer Type	Implementation Defined
		Enumerated Type		
	Floating Type	Real Floating Type	Standard Floating Type	float double long double
			Decimal Floating Type	_Decimal32 _Decimal64 _Decimal128
		Complex Type		float _Complex double _Complex long double _Complex

Arithmetic Type（算术类型）除了Enumerated Type，统称为Basic Type

注意bool类型，char类型

Derived Type



Derived Type可以递归地构造

These methods of constructing derived types can be applied recursively.

Object Type: Array Type

1、Element Type (T)：需要是Object Type

2、Number of Elements (N)

Array of T，例如：假设T是int类型，N是5，则该对象类型为int[5]

递归地构造

Array of T，例如：假设T是int[5]类型，N是3，则该对象类型为int[3][5]

数组类型要从一维的角度去理解

int[5]真的是对象类型吗？

sizeof(5) vs. sizeof 5

sizeof(int) vs. sizeof int

sizeof(int[5]) vs. sizeof int[5]

sizeof用括号引导的只能是对象类型

Object Type: Pointer Type

给定任何一个Type (T)，都有对应的一个指针类型Pointer to T

1、Object Type

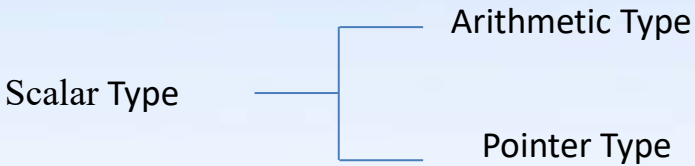
2、Function Type

} **Referenced Type** ⟺ **Pointer Type**

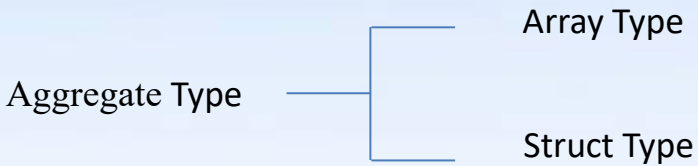
int	int*
int[5]	int(*)[5]
int*	int**

任何指针类型也是一个Object Type，把指针类型当作普通Object Type看待

Scalar Type



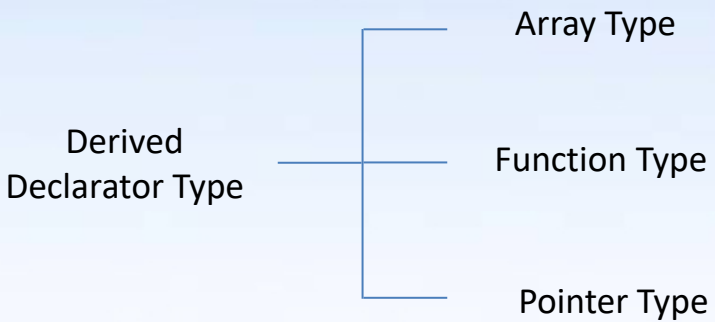
Aggregate Type



聚合类型不包括联合体，因为联合体类型任一时刻只能包括一个成员类型

Note that aggregate type does not include union type because an object with union type can only contain one member at a time

Derived Declarator Type



Incomplete Object Type

缺少必要信息确定Object大小的Type

1、缺少元素个数的数组，如extern int a[];

2、Variable Length Array（Length还未确定的时候）

3、包含Incomplete Object的struct/union

4、void

Object Type: Qualified Type

1、const

2、volatile

3、restrict

unqualified/qualified version是两种不同的Type，但大小，表示值，对齐方式一样

4、_Atomic

_Atomic(type-name)和type-name大小，表示值，对齐方式不一样

Atomic types are a conditional feature that implementations need **not** support

分配对象

在C语言中，分配一段内存（对象）的方法有4种：

1、对象声明
2、使用malloc分配
3、字符串
4、Compound Literal

我们先来看通过对象声明的方法

基本概念：对象声明（Declaration）

在C语言中，声明一个对象具有两个基本要素：
1、对象类型 Object Type
2、对象名称（标示符） Identifier

对象类型约定了这段内存的类型和大小
对象名称则是这块内存的别名，用于识别定位这块内存

对象类型进一步分为：
1、非数组对象类型
2、数组对象类型

对象声明可以形式化成*T O*

非数组对象类型及声明

包括int, short, float, double, char, 结构体/联合体等等

声明方式如下：

对象类型为int, 对象名称为a	int a;
对象类型为float, 对象名称为b	float b;
对象类型为double, 对象名称为c	double c;

任何对象类型分配的内存大小可以通过sizeof这个操作符来获取，例如：

sizeof(int), sizeof(float), sizeof(double)

数组对象类型

数组对象类型是多个同样的对象类型组成的一种一维构造类型
包含两要素：1、元素个数，2、元素的对象类型

1、数组对象类型的元素可以是非数组对象类型

int[2]包括2个元素，每个元素对象类型是int
char[3]包括3个元素， 每个元素对象类型是char

任何数组对象类型分配的内存也可以通过sizeof这个操作符来获取

sizeof(int[2]), sizeof(char[3])

数组对象类型

数组对象类型是多个同样的对象类型组成的一种**一维派生类型**
包含两要素：1、元素个数，2、元素的对象类型

2、数组对象类型的元素也可以是其他数组对象类型

float[2][3]包括2个元素，每个元素对象类型是float[3]
double[2][3][4]包括2个元素，每个元素对象类型是double[3][4]

仍然可以通过sizeof这个操作符来获取对象类型的大小

sizeof(float[2][3]), sizeof(double[2][3][4])

注意：C语言数组对象类型其实都是一维的

数组对象类型的声明

数组对象类型申明方式如下：

对象类型为int[2], 对象名称为a　　int a[2];
对象类型为float[2][3], 对象名称为b　　float b[2][3];
对象类型为double[2][3][4], 对象名称为c　　double c[2][3][4];

需要能正确的识别出数组对象声明中的对象类型和对象名称

指针对象类型

1、指针对象类型是一个非数组对象类型

2、任何一种对象类型，都有指向其的一个指针对象类型，反之亦然

int、float、double

int[2]、float[2][3]、double[2][3][4]

int*、float*、double*

int(*)[2]、float(*)[2][3]、double(*)[2][3][4]

int对象类型（Referenced Type）对应的指针对象类型是int*（Pointer Type)

int[2]对象类型（Referenced Type）对应的指针对象类型是int(*)[2]（Pointer Type）

指针对象类型

指针对象类型也有相应的指向其的另一个指针对象类型

int*、float*、double*

int(*)[2]、float(*)[2][3]、double(*)[2][3][4]

int**、float**、double**

int(**)[2]、float(**)[2][3]、double(**)[2][3][4]

指向int*对象类型的指针对象类型是int**

指向int(*)[2]对象类型的指针对象类型是int(**)[2]

指针对象类型

指针对象类型也可以用来构造数组类型，作为数组对象中元素的对象类型

int*[2]、float*[2][3]、double*[2][3][4]

int*[2]包括2个元素，每个元素对象类型是int*
float*[2][3]包括2个元素，每个元素对象类型是float*[3]
double*[2][3][4]包括2个元素，每个元素对象类型是double*[3][4]

提示：把int*, float*, double*当作一个整体来看待

把int*当作一个整体，比如看作PINT

int*[2]  PINT[2]

PINT[2]包括2个元素，每个元素类型是PINT

指针对象类型

指针对象类型构成的数组对象类型也有其对应的指针对象类型

int*[2]、float*[2][3]、double*[2][3][4]  int*(*)[2]、float*(*)[2][3]、double*(*)[2][3][4]

提示：把int*, float*, double*当作一个整体来看待

把int*当作一个整体，比如看作PINT

int*[2]  PINT[2]

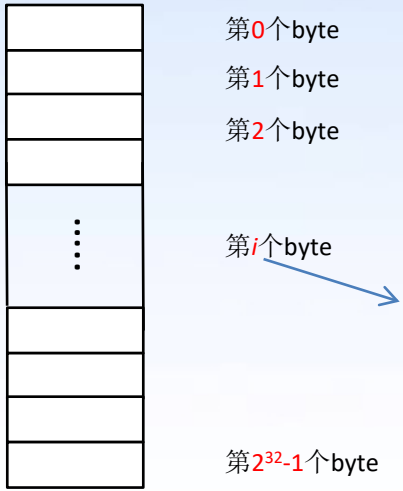
PINT[2]  PINT(*)[2]

指针对象申明

对象类型是int*，对象名是a	int* a
对象类型是float*，对象名是b	float* b
对象类型是double*，对象名是c	double* c
对象类型是int(*)[2]，对象名是d	int (*d)[2]
对象类型是float(*)[2][3]，对象名是e	float (*e)[2][3]
对象类型是double(*)[2][3][4]，对象名是f	double (*f)[2][3][4]
对象类型是int*[2]，对象名是g	int* g[2]
对象类型是float*[2][3]，对象名是h	float* h[2][3]
对象类型是double*[2][3][4]，对象名是i	double* i[2][3][4]

理解sizeof(int*), sizeof(int**), sizeof(int(*)[2])

指针对象类型的取值范围



32位机内存

任何指针对象的值代表内存中一个字节的编号

取值范围就是[0 ~ 最大可寻址的字节编号]

对32位机器而言，可以寻址的字节编号范围为

[0 ~ 2^{32}-1]

转换成16进制表示

[0x00000000 ~ 0xFFFFFFFF]

NULL

基本概念：对象类型总结

假设对象类型Object_Type (Obj_T)，对象名Name (N)

1、对象申明可以形式化定义为Obj_T N;

2、对象类型Obj_T对应的指针类型可以形式化记为Obj_T*

注1：具体对象申明语法和对象类型有关，例如：
int a; vs. int b[10];

注2： Obj_T*里面*的位置跟语法有关，例如：
int*; vs. int(*)[10];

基本概念：对象类型总结

Variable Category	Variable Type	Variable Name	Variable Declaration	Element Type	Corresponding Pointer Type	Size
Non-Array Variables	int	a	int a;	N/A	int*	4
	char	b	char b;	N/A	char*	1
	float	c	float c;	N/A	float*	4
	int*	d	int* d;	N/A	int**	4
	char*	e	char* e;	N/A	char**	4
	float*	f	float* f;	N/A	float**	4
Array Variables	int[2]	g	int g[2];	int	int(*)[2]	8
	char[2][3]	h	char h[2][3];	char[3]	char(*)[2][3]	6
	float[2][3][4]	i	float i[2][3][4];	float[3][4]	float(*)[2][3][4]	96
	int*[2]	j	int* j[2];	int*	int*(*)[2]	8
	char*[2][3]	k	char* k[2][3];	char*[3]	char*(*)[2][3]	24
	float*[2][3][4]	l	float* l[2][3][4];	float*[3][4]	float*(*)[2][3][4]	96

思考题

1、int**[3] 对象类型包括几个元素，每个元素是什么对象类型？

2、指向int**[3]对象类型的指针对象类型是什么？

3、如何声明一个对象，对象名为a，对象类型为int**[3]

答案（提示：int**也是一个对象类型，将int**当作一个整体来看）：
1、int**[3]数对象类型是一个一维数组类型，包括3个元素，其中每个元素的对象类型是int**

2、int**(*)[3]

3、int** a[3]（建议）或int **a[3]

思考题

1、32位机里面，int*对象类型的取值需要几个字节来存储？

2、64位机里面，int*的取值范围是多大？需要几个字节来存储？

答案

1、32位机里面，int*的取值范围是[0x00000000, 0xFFFFFFFF]，因此4个字节来存储就够了，所以sizeof(int*)=4

2、64位机里面，理论上可以访问的字节有2^{64}个，因此字节编号范围为[0x0000000000000000, 0xFFFFFFFFFFFFFFFF]，需要8个字节来存储，因此sizeof(int*)=8

sizeof(int*)可以判断你的编译器是8/16/32/64位

了解一下typedef

C语言提供了一个typedef的方法来为对象类型提供别名

Original Type	Target Type	Typedef Declaration
char	INT1	typedef char INT1;
short	INT2	typedef short INT2;
int	INT4	typedef int INT4;
int*	PINT	typedef int* PINT;
char*	PCHAR	typedef char* PCHAR;
int(*)[2]	PAINT	typedef int (*PAINT)[2];
int**	PPINT	typedef int** PPINT;
int[2]	AINT	typedef int AINT[2];
int[2][3]	AAINT	typedef int AAINT[2][3];

1、typedef char INT1;
将char定义一个别名INT1
char c; vs. INT1 c;

2、typedef int AINT[2];
注意定义数组别名的语法
int a[2]; vs. AINT a;

int[3]真的是对象类型吗？

Original: int[3], Target: VINT

typedef int VINT[3];

通过typedef新定义出来的VINT是不是一个合法的对象类型？

```
VINT e;
VINT* p = &e;
int (*q)[3] = &e;
```

VINT是一种对象类型

如何将int[2][3]定义为VVINT类型？

方法1

Original : int[2][3], Target: VVINT

typedef int VVINT[2][3];

int g[2][3]; 等价于 VVINT g;
VVINT* p = &g;
int (*q)[2][3] = &g;

方法2

利用刚才定义的VINT的方法
1、typedef int VINT[3];
2、typedef VINT VVINT[2];

int g[2][3]; 等价于 VVINT g;
VVINT* p = &g;
int (*q)[2][3] = &g;

方法2更清晰看出如何用一维的视角去看待“高维”数组

int* vs. PINT

typedef int* PINT;

1、int* p, q;

2、PINT r, s;

1、p是int*类型，q是int类型

2、r和s都是PINT类型，也就是int*类型

int* p vs. int *p：空格放在哪更好？

int *p;
int* p;

1、int *p和int* p是一样的
都定义了一个int*类型的对象，对象名为p

int *p, q;
int* p, q;

2、int *p, q和int* p, q是一样的
定义了一个int*类型的对象，对象名为p，同时定义了一个int类型的对象，对象名为q

int *p, *q;

3、int *p, *q才是定义了两个int*类型的对象，对象名分别为p和q

int* p;
int* q;

4、建议int*，因为int*是一个对象类型，因此多个int*对象申明，建议一行一个

思考题

```
typedef int* PINT;
PINT p;

PINT* q = &p;
```

标示符q对应那块内存，实际是什么对象类型？

答案

```
int** q = &p;
```

思考题

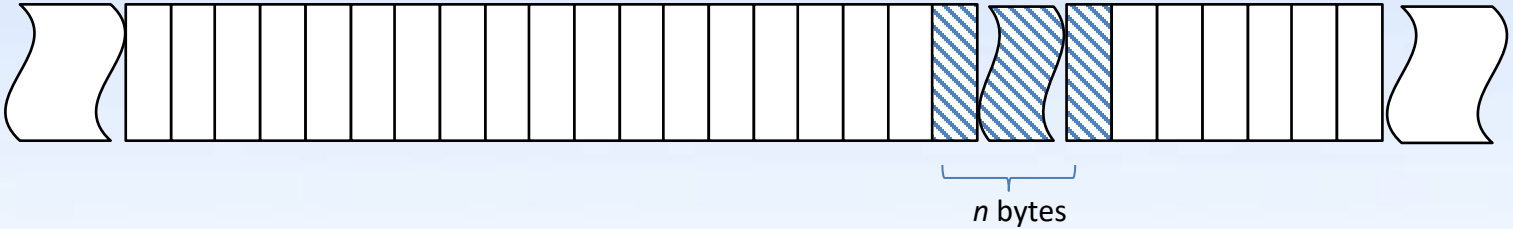
1、char[3][5]对象类型包括几个元素，每个元素是什么对象类型？

2、sizeof(int[7][8])=？

答案：
1、char[3][5]对象类型是一个一维数组对象类型，包括3个元素，其中每个元素的对象类型是char[5]

2、sizeof(int[7][[8])=7*sizeof(int[8])=7*8*sizeof(int)=7*8*4=224字节

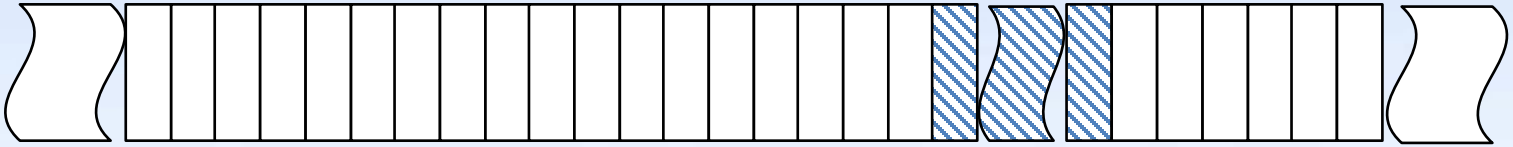
刻画对象的属性：对象大小



对象占用的连续字节个数

对象的大小和对象的类型相关，sizeof(Obj_T)可以获得这个对象的大小

刻画对象的属性：对象名字



如果一个对象有名字，即named object，系统通过对象名称可以直接定位这块内存

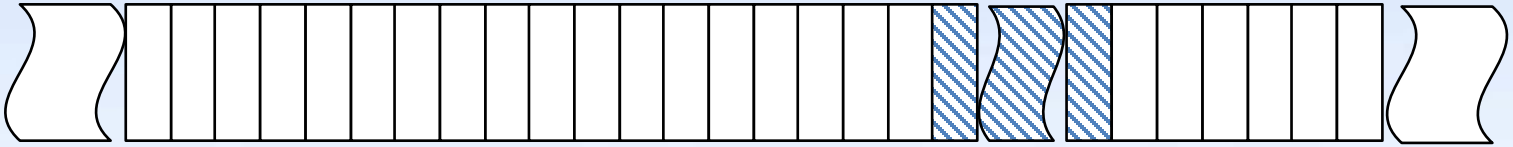
例如int a; (*T O*)

对象的标识符(*O*)是这块内存对象的名字

标识符(*O*)就可以用来直接定位到这个对象(designate object)

如果一个对象没有名字，即unnamed object，需要通过其他机制来定位这块内存

刻画对象的属性：对象对齐要求

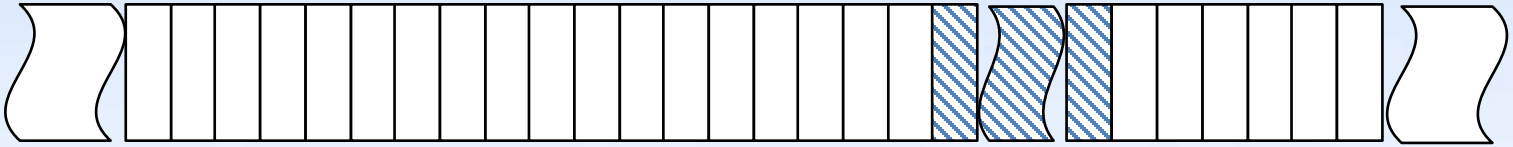


对齐要求一般是要求对象的首地址能被对齐要求整除

对象的对齐要求和对象的类型相关， _Alignof(Obj_T)可以获得这个对象类型的对齐要求

对象的对齐要求可以修改

刻画对象的属性：值



<Value, Value Type>

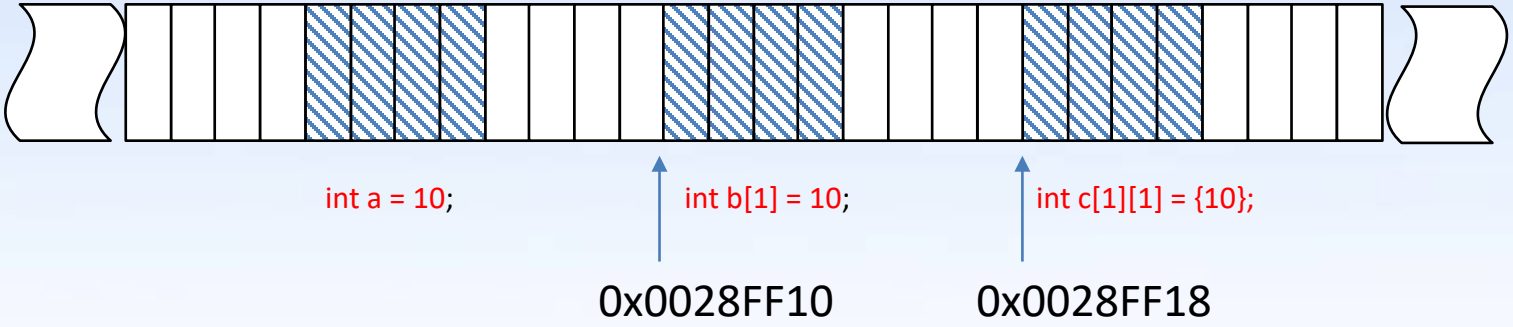
对象都有值（Value），且这个值和值的类型（Value Type）和对象类型有关

1、非数组对象类型，值类型=对象类型，对象表示按规则转换成Value

2、数组对象类型，<第一个元素的首地址，元素类型对应的pointer type>

int a[2]，该对象类型是int[2]，所以这块内存的值就是<第一个元素的首地址，int*>

刻画对象的属性：值的示例



对象a、b、和c对应的对象的32为0/1串是一样的

对象a对应的对象的值: <10, int>

对象b对应的对象的值: <0x0028FF10, int*>

对象c对应的对象的值: <0x0028FF18, int(*)[1]>