投票 最多可选 1 项

设置

你课前你学习过《程序设计基础CAP》吗？

A 学过，学得还可以

B 学过，没学明白

C 没学过

提交

投票 最多可选 1 项

本课程的课程进度文档看了吗？
第一次课的课前MOOC自学内容（常量与变量，标准数学函数；数据的键盘输入，数据的屏幕输出）课前预习了吗？

A 看了文档，也预习了

B 看了文档，但没有预习

C 没看文档，但在C精髓MOOC中学习过了

D 没看文档，也没有预习

提交

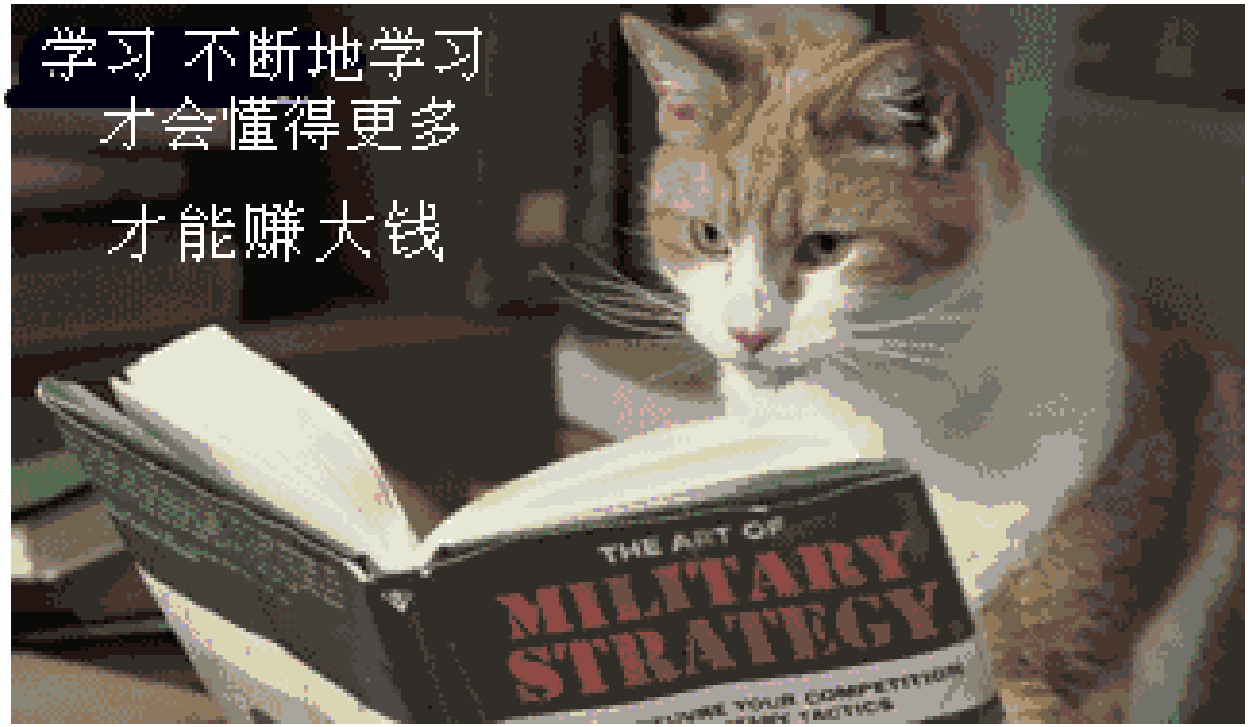
设置

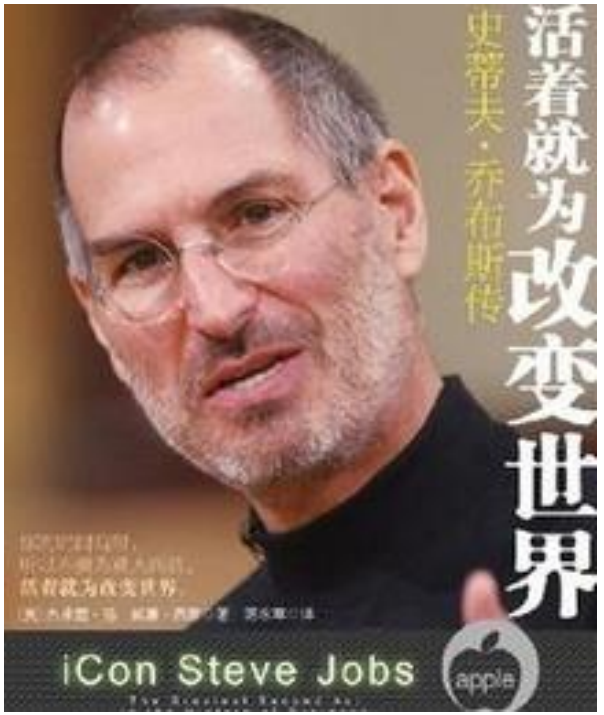
我的讲课风格

- 优点：
  - 自学能看懂的MOOC或教材的基础内容，课上不会再重复
  - 根据学生的接受程度，适当补充课外内容，如算法和数据结构，指针和内存管理，游戏程序设计等
- 缺点：
  - 不会重复基础内容，希望讲基础的同学慎选
  - 讲课进度和语速较快，估计跟不上的同学慎选
  - 要求课前必须自学MOOC指定内容，不想课前自学的同学慎选

学编程主要学什么？

* 每个人都应该学习给电脑编写程序的技术，因为这一过程能够教你如何去思考！—史蒂夫·乔布斯





如何学好编程？

- 师傅领进门，出徒在个人
- 从**实践**中学习
- 从**案例**中学习
- 从**错误**中学习
- 从**互联网**上学习
- 从**同学**中学习
- **持续写**代码
- **先思考，后提问**

勤学苦练

数据的基本输入输出

（教材2.1~2.4，3.2~3.3节）

哈尔滨工业大学

苏小红

sxh@hit.edu.cn

认识第一个C语言程序

* 1：向屏幕输出"Hello world!"
* 2：分两行输出"Hello world!"

```
#include  <stdio.h>
int main()
{
    printf("Hello world\n");
    return 0; //返回0值表示程序运行成功
}
```

* C语言是一种很小的语言，若不使用外部库，几乎什么也干不了
* #include告诉编译器要使用哪些外部代码
* 计算机运行程序时需要一些方法判断程序是否运行成功

能否来点高大上的？

认识第一个C语言程序

* 3：输出带颜色的"Hello world!"
* windows操作系统下system() 函数
* 功 能： 发出一个DOS命令
  * #include <stdlib.h>
* 用 法： int system(char *command);
  * system("color 1E"); //改变控制台的前景色和背景
    * 0=黑色 1=蓝色 2=绿色 3=湖蓝色 4=红色 5=紫色 6=黄色 7=白色 8=灰色 9=淡蓝色 A=淡绿色 B=淡浅绿色 C=淡红色 D=淡紫色 E=淡黄色 F=亮白色
  * system("pause"); //冻结屏幕，观察程序执行结果
  * system("worldstart.mp4"); //播放视频

认识第一个C语言程序

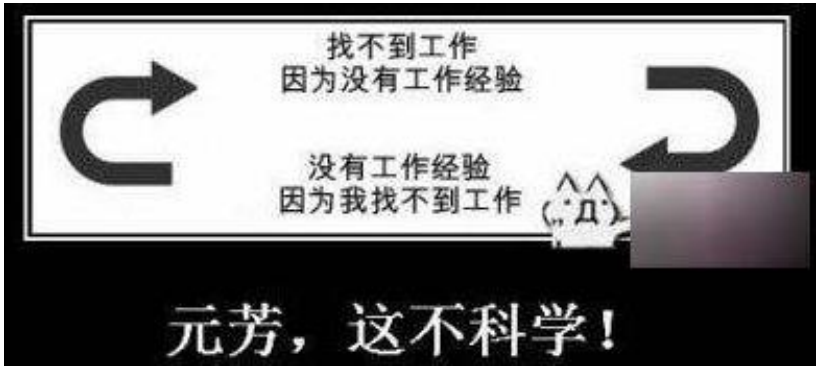
* 4：让"Hello world!"动起来
  * 水平移动—用'\t'
  * 垂直移动—用'\n'
* system("cls"); //清屏操作
* Sleep(500); //延时，以毫秒为单位
  * #include <windows.h>

清屏
在当前位置输出
延时
清屏
在下一个位置输出
延时
……

认识第一个C语言程序

* **5：奔跑吧，"Hello world!"，不要停下来**
* **循环语句**

```
while (1)    //死循环
{
    清屏
    在前一个位置输出
    延时
    清屏
    在下一个位置输出
    延时
    ……
}
```



认识第一个C语言程序

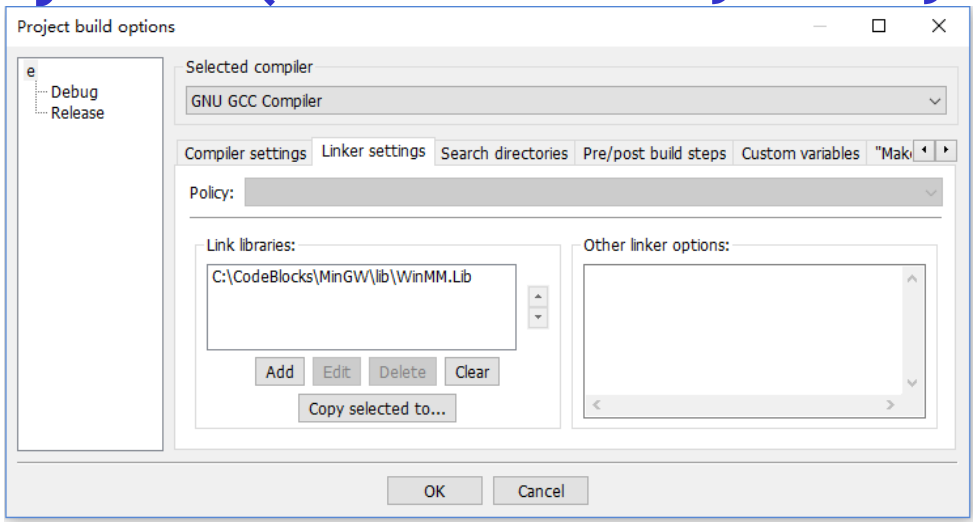
* 6：奔跑吧，"Hello world!"，按任意键戛然而止
* 键盘输入检测函数
  * kbhit()
    * 在用户有键盘输入时返回1，否则返回0
    * #include <conio.h>

```
while (!kbhit()) //没有键盘输入就继续循环
{
    ……
}
```

认识第一个C语言程序

* 7：让"Hello world!"随着音乐奔跑起来吧
* 播放wav格式的声音
  * PlaySound()
    * #include <windows.h>
    * 需要导入多媒体库WinMM.lib

```
PlaySound("huluwa.wav", NULL, SND_ASYNC);//异步
PlaySound("huluwa.wav", NULL, SND_SYNC); //同步
```



单选题

设置

定义一个宏常量PI，其值为3.14159

A　#define　PI　3.14159;

B　#define　PI　3.14159

C　#define　PI = 3.14159;

D　#define　PI = 3.14159

提交

认识第2个C语言程序

```
#include  <stdio.h>
#define  PI  3.14159
#define  R   5.3
int main()
{
    printf("area = %f\n",  3.14159 * 5.3 * 5.3 );
    printf("circumference = %f\n",  2 * 3.14159 * 5.3 );
    return 0;
}
```

* 优点：减少重复书写常数的工作量，提高程序的可读性和可维护性
* 缺点：无数据类型，编译器在宏替换时不进行类型检查，仅傻瓜式字符串替换，不做语法检查，极易产生意想不到的错误

认识第2个C语言程序

```
#include  <stdio.h>
int main()
{
    const float pi = 3.14159; //存在只读存储区，只能在定义时赋初值
    const float r = 5.3;
    printf("area = %f\n", pi * r * r);
    printf("circumference = %f\n", 2 * pi * r);
    return 0;
}
```

* const常量有数据类型，编译器能对其进行类型检查

认识第2个C语言程序

高级语言为什么要区分数据类型？

- 可读性
- 可靠性
- 时空效率

```
#include  <stdio.h>
int main()
{
    const float pi = 3.14159;
    float r;
    printf("Input r:");
    scanf("%f", &r);    //%f就是个“要债的”，r就是“还债的”
    printf("area = %f\n", pi * r * r);
    printf("circumference = %f\n", 2 * pi * r);
    return 0;
}
```

(1)不同类型数据占用的内存大小不同

```
#include  <stdio.h>
int main()
{
    printf("Data type          Number of bytes\n");
    printf("------------     ---------------------\n");
    printf("char                %d\n", sizeof(char));
    printf("short int           %d\n", sizeof(short));
    printf("int                 %d\n", sizeof(int));
    printf("long int            %d\n", sizeof(long));
    printf("long long int       %d\n", sizeof(long long)); //%I64d
    printf("float               %d\n", sizeof(float));
    printf("double              %d\n", sizeof(double));
    printf("long double         %d\n", sizeof(long double));
    return 0;
}
```

* int,long double,long long与系统和编译器相关，用sizeof运算符计算
* 编译时执行的运算符，不增加额外的运行时间开销，增强可移植性

(1)不同类型数据占用的内存大小不同

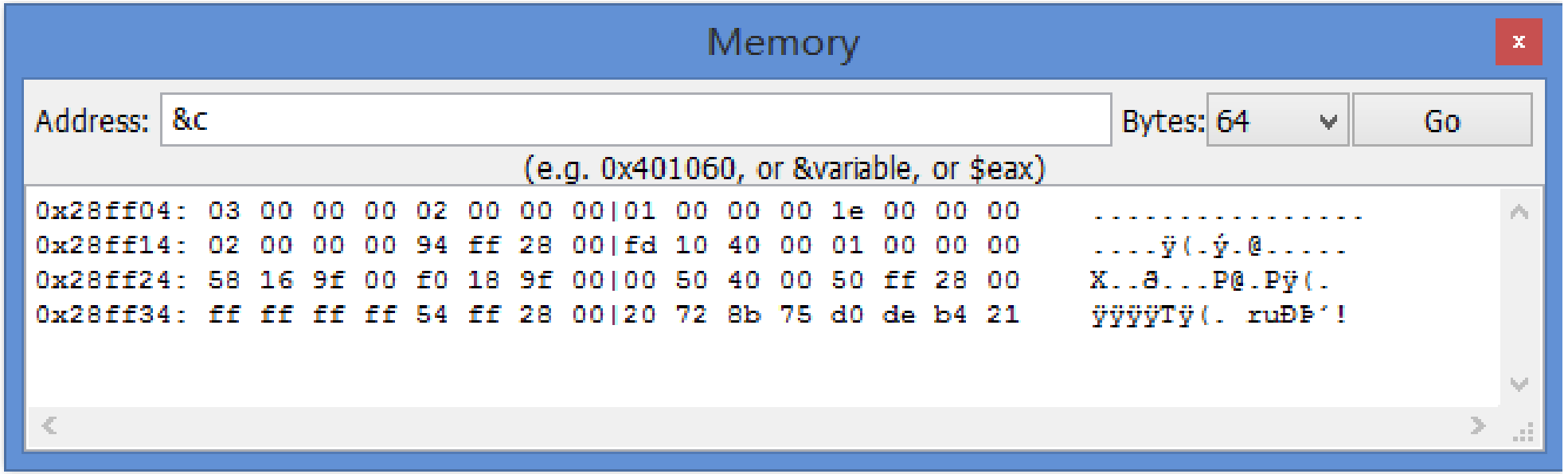
- 如何衡量内存空间的大小？
- 二进制位，也称比特（bit）
  - 计算机存储数据的最基本单元，衡量物理存储器容量的最小单位
- 一个二进制位无法表示太多数据，怎么破？
  - 将许多位合起来使用
- 8bit是1个字节（Byte），可表示0~255间的整数
  - 最小的可寻址的存储器单位
  - 用字节数衡量内存空间大小

表 1-1　衡量内存空间大小的表示单位

英文称谓	容量大小（单位字节）	换算方法
B	2 的 0 次方	1 B == 8 b
KB	2 的 10 次方	1 KB == 1,024 B
MB	2 的 20 次方	1 MB == 1,024 KB
GB	2 的 30 次方	1 GB == 1,024 MB
TB	2 的 40 次方	1 TB == 1,024 GB
PB	2 的 50 次方	1 PB == 1024 TB
EB	2 的 60 次方	1 EB == 1024 PB
ZB	2 的 70 次方	1 ZB == 1024 EB
YB	2 的 80 次方	1 YB == 1024 ZB

(1)不同类型数据占用的内存大小不同

- 内存是如何编址的？
  - 内存按字节（Byte）编址（线性地址表）
    - 1个字节，一个地址（Address）——十六进制无符号整数



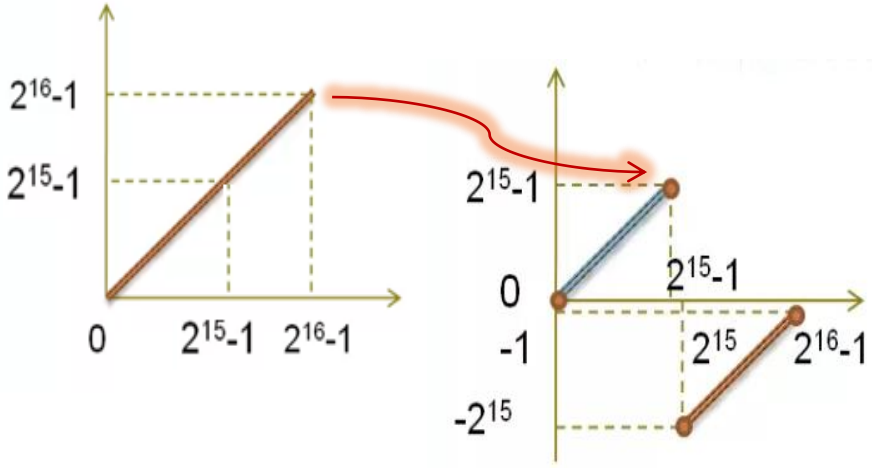
(2)不同类型数据的表数范围不同

```
#include <stdio.h>
#include <limits.h>
int main()
{
  printf("Maximum of char:  %d\n", CHAR_MAX);
  printf("Minimum of char:  %d\n", CHAR_MIN);
  printf("Maximum of short: %d\n", SHRT_MAX);
  printf("Minimum of short: %d\n", SHRT_MIN);
  printf("Maximum of int:   %d\n", INT_MAX);
  printf("Minimum of int:   %d\n", INT_MIN);
  printf("Maximum of unsigned int:  %u\n",  UINT_MAX);
  printf("Maximum of long:          %ld\n", LONG_MAX);
  printf("Minimum of long:          %ld\n", LONG_MIN);
  printf("Maximum of unsigned long: %lu\n", ULONG_MAX);
  return 0;
}
```

* 占内存字节数大，则表数范围就大

(2)不同类型数据的表数范围不同

- 以2字节短整型为例
  - 有符号数与无符号数的区别在于如何解释其最高位
  - 有符号整数的数据位比无符号整数的数据位少1位



无符号短整型（最高位是数据位）		有符号短整型（最高位是符号位）	
二进制补码	十进制	二进制补码	十进制
00000000 00000000	0	00000000 00000000	0
00000000 00000001	1	00000000 00000001	1
00000000 00000010	2	00000000 00000010	2
00000000 00000011	3	00000000 00000011	3
…		…	
01111111 11111111	32767	01111111 11111111	32767
10000000 00000000	32768	10000000 00000000	-32768
10000000 00000001	32769	10000000 00000001	-32767
…		…	
11111111 11111110	65534	11111111 11111110	-2
11111111 11111111	65535	11111111 11111111	-1

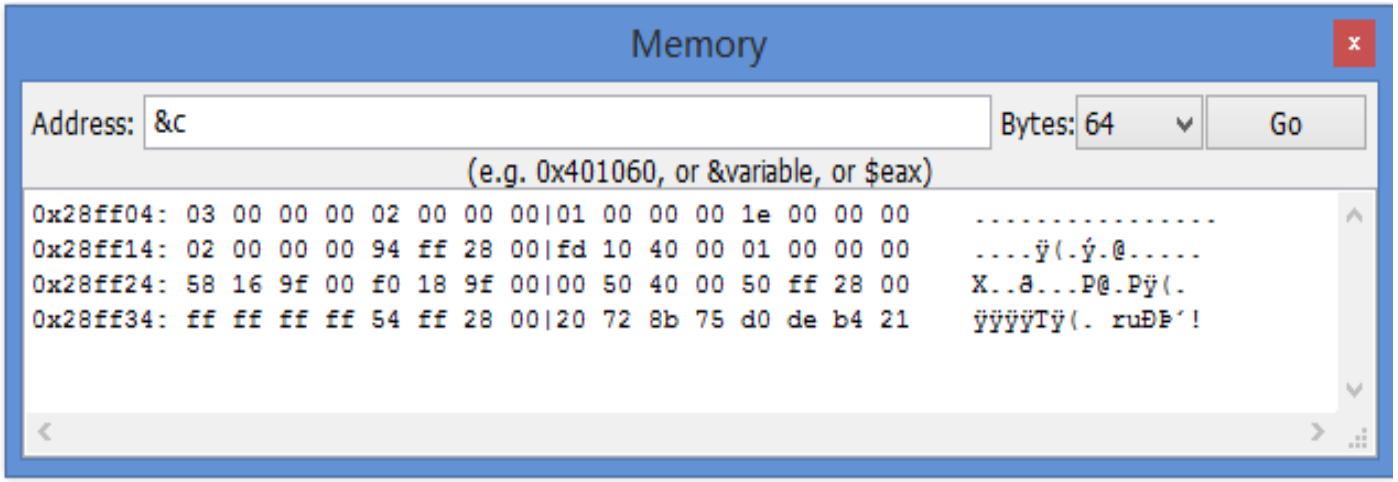
(3)不同类型数据的存储形式不同

- 1.如何存储一个多字节整数？

高位字节	低位字节

小端次序（Little-endian）

- 小端次序：便于计算机从低位字节向高位字节运算
- 大端次序：与人的书写顺序相同，便于处理字符串



大端次序（Big-endian）

(3)不同类型数据的存储形式不同

- 2.如何存储一个**实数**？
  - 关键：确定小数点的位置
  - **定点数（Fixed Point）**
    - 小数点的位置固定
    - 定点整数（只有整数部分，小数点位于数值位的最低位）
    - 定点小数（纯小数，小数点位于符号位和最高数位之间）

整数部分	小数部分

  - **浮点数（Floating-Point）**
    - 指数表示形式——科学计数法
    - 小数点位置可浮动（改变指数）：更适合表示绝对值很大或很小的数
    - 拆分成**阶码（Exponent）**和**尾数（Mantissa）**分别存储

阶码（Exponent）

指数部分	小数部分

尾数（Mantissa）

(3)不同类型数据的存储形式不同

- 为什么浮点数有单精度和双精度之分？
- 计算机为什么采用浮点数而非定点数来表示实数？
- 浮点数是实数的精确表示吗？

既然浮点数能表示更大的数，可否用浮点数取代整数？

阶码E（指数部分）	尾数M（小数部分）

决定实数的表数范围

决定实数的表数精度

$N = r^E \times M$

r 为基数

通常取 2

定点数					$-2^{31} \sim 2^{31}-1$
单精度浮点数					$-3.402823466\times10^{38}$ ~ 3.402823466×10^{38}

(3)不同类型数据的存储形式不同

- 3.如何存储一个字符？
  - 以二进制编码方式存储
- 字符编码方式
  - 取决于计算机系统所使用的字符集
  - ASCII（美国标准信息交换码）字符集
  - 每个字符有一个编码值（查ASCII码表）
  - 字符常数就是一个普通整数
  - 256个字符——1字节

‘H’	0	1	0	0	1	0	0	0	72

十进制 ASCII 码	字　符	十进制 ASCII 码	字　符	十进制 ASCII 码	字　符
0	NUL	43	+	86	V
1	SOH(^A)	44	,	87	W
2	STX(^B)	45	-	88	X
3	ETX(^C)	46	.	89	Y
4	EOT(^D)	47	/	90	Z
5	EDQ(^E)	48	0	91	[
6	ACK(^F)	49	1	92	\
7	BEL(bell)	50	2	93	]
8	BS(^H)	51	3	94	^
9	HT(^I)	52	4	95	-
10	LF(^J)	53	5	96	`
11	VT(^K)	54	6	97	a
12	FF(^L)	55	7	98	b
13	CR(^M)	56	8	99	c
14	SO(^N)	57	9	100	d
15	SI(^O)	58	:	101	e
16	DLE(^P)	59	;	102	f
17	DC1(^Q)	60	<	103	g
18	DC2(^R)	61	=	104	h
19	DC3(^S)	62	>	105	i
20	DC4(^T)	63	?	106	j
21	NAK(^U)	64	@	107	k
22	SYN(^V)	65	A	108	l
23	ETB(^W)	66	B	109	m
24	CAN(^X)	67	C	110	n
25	EM(^Y)	68	D	111	o
26	SUB(^Z)	69	E	112	p
27	ESC	70	F	113	q
28	FS	71	G	114	r
29	GS	72	H	115	s
30	RS	73	I	116	t
31	US	74	J	117	u
32	Space(空格)	75	K	118	v
33	!	76	L	119	w
34	"	77	M	120	x
35	#	78	N	121	y
36	$	79	O	122	z
37	%	80	P	123	{
38	&	81	Q	124	|
39	'	82	R	125	}
40	(	83	S	126	~
41	)	84	T	127	del
42	*	85	U		

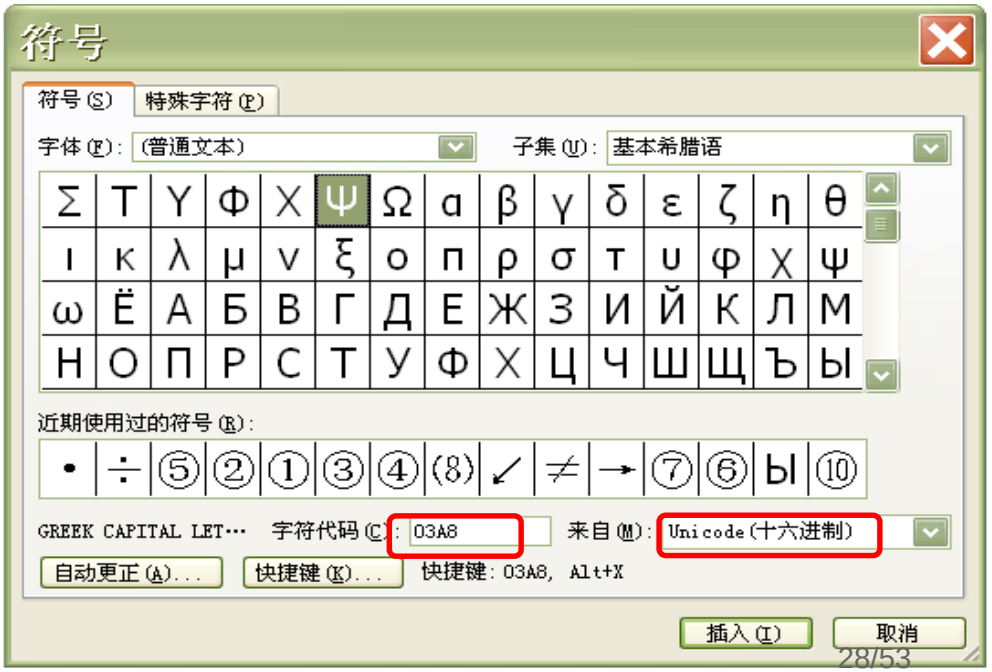
(3)不同类型数据的存储形式不同

- 如何表示一个字符?
- 字符常量：用单引号括起来的一个字符
  - '3'表示一个数字字符，而3则表示一个整数数值
  - 'b'表示一个字符，而b则表示一个标识符
- 转义字符（Escape Character）
  - 一些特殊的控制字符

字　符	含　义	字　符	含　义
'\n'	换行（Newline）	'\a'	响铃报警提示音（Alert or Bell）
'\r'	回车（不换行）（Carriage Return）	'\" '	一个双引号（Double Quotation Mark）
'\0'	空字符，通常用作字符串结束标志（Null）	'\' '	单引号（Single Quotation Mark）
'\t'	水平制表（Horizontal Tabulation ）	'\\'	一个反斜线（Backslash）
'\v'	垂直制表（Vertical Tabulation）	'\?'	问号（Question Mark）
'\b'	退格（Backspace）	'\ddd'	1 到 3 位八进制 ASCII 码值所代表的字符
'\f'	走纸换页（Form Feed）	'\xhh'	1 到 2 位十六进制 ASCII 码值所代表的字符

(3)不同类型数据的存储形式不同

- **4.如何存储一个汉字？**
  - 汉字编码，兼容ASCII码，连续的2个字节，仅在其第7位均为1时认为是汉字
  - GB2312，6763字；GB13000.1，20902字；GB18030，27533字
- **其他国家的语言文字呢？**
  - 不同国家和地区制定了不同的编码标准，互不兼容，**不能跨语言文本转换**
- **更强大的编码**
  - ISO制定了Unicode字符集，为各种语言中的字符设定统一且唯一的数字编号
  - 所有字符统一用2个字节保存，宽字节字符——65536个字符



(4)不同数据类型可参与的运算不同

- 整型
  - 加、减、乘、除、求余
- 实型
  - 加、减、乘、除
- 字符型
  - 加、减（整数）
  - 对ASCII码值的运算
- 指针类型
  - 加、减（整数）和比较运算

数据的基本运算

教材2.5，3.1,3.4,3.5，5.3,5.9节

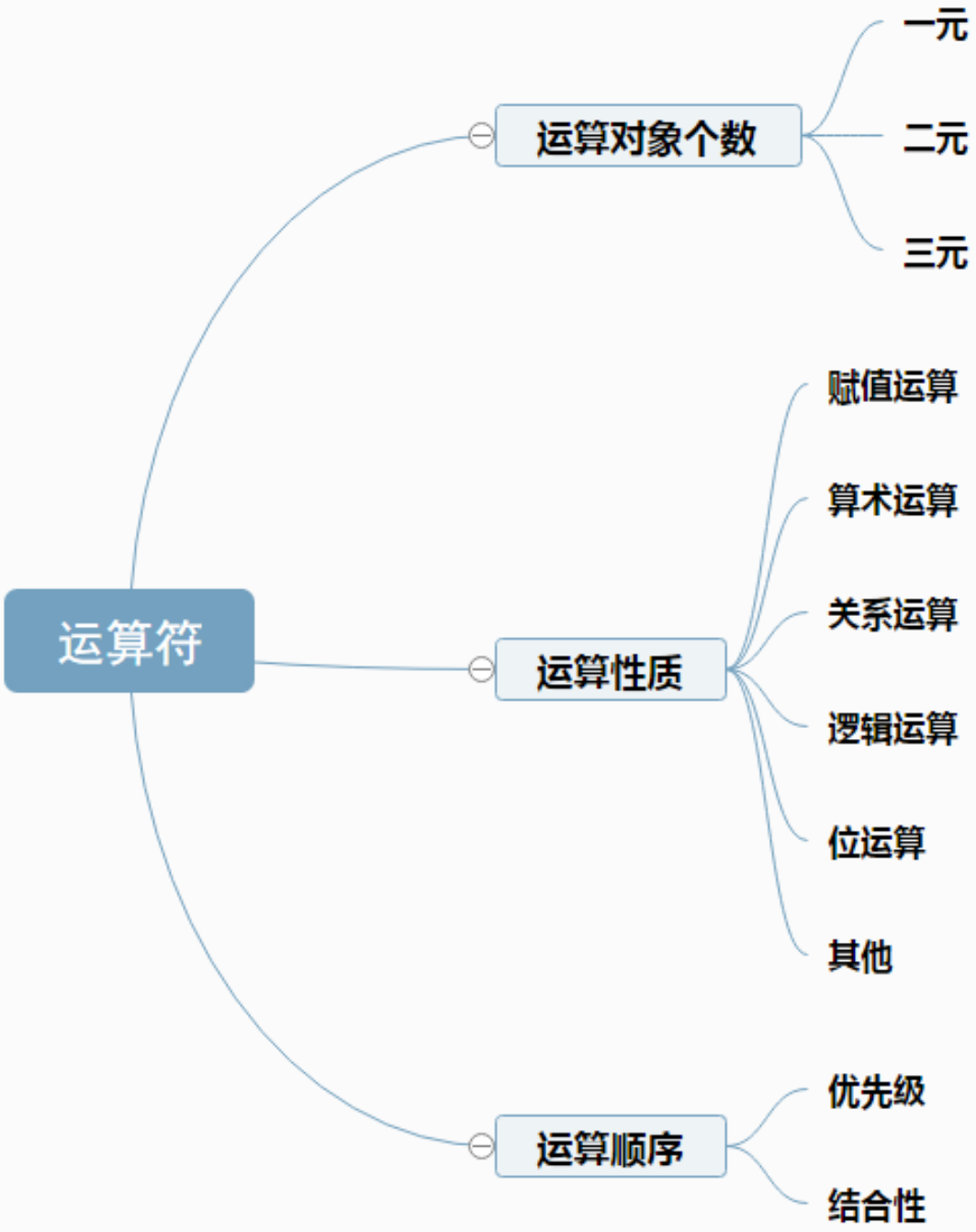
MOOC第2周，第4周4.1

哈尔滨工业大学

苏小红

sxh@hit.edu.cn

C运算符（Operator）的分类



单选题

下列语句中错误的是（）。

A
```
int a, b, c;
a=b=c=0;
```

B
```
int a, b, c;
a=0;
b=0;
c=0;
```

C
```
int a=0;
int b=0;
int c=0;
```

D
```
int a=b=c=0;
```

提交

赋值运算符和数学中的等号有何区别？

* 如何理解 a = a + 1?
  * 有方向性，要考虑优先级和结合性

a = b = 1　　赋值表达式的值就是它的左值

a += 1

a++

++a

单选题

已知n=3;则执行printf(“%d\n”,-n++);语句后屏幕输出的结果是（）

A -3

B -2

C -4

D 以上都不对

提交

前缀与后缀对变量和表达式有何影响？

```
m = -n++;
```

```
m = -(n++);
```

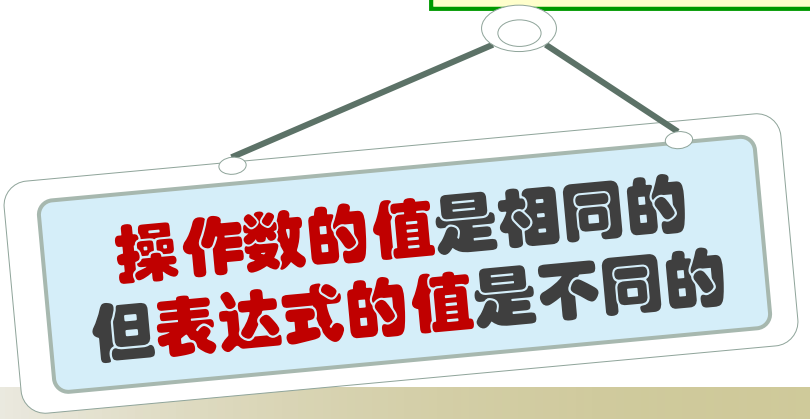

```
m = -++n;
```

```
m = -(++n);
```

```
m = -n;
n = n + 1;
```

建议一个变量只出现一次自增或自减运算

```
n = n + 1;
m = -n;
```



除法运算有何玄机？

11 / 5 = 2

整数除法

(Integer Division)

11.0 / 5 = 2.2

浮点数除法

(Floating Division)

自动类型转换

- C编译器将所有操作数都转换成取值范围较大的操作数的类型——类型提升（Type Promotion）

除法运算有何玄机？

- 若两变量都是int，还想得到浮点运算结果，怎么办？

```
aver = total / n;
aver = (float)total / n;
```

- 强制类型转换运算符–类型强转（Casting）

(float)(total/n) 结果如何？

* 类型强转改变变量原有的类型吗？

求余运算有何玄机？

* 操作数必须是**整数**，返回a与b相除之后的**余数**

```
11 % 5 = 1
```

```
    2
5 ) 11
    10
    ---
    1
```

```
11 % (-5) = 1
```

```
    -2
-5 ) 11
     10
     ---
     1
```

```
(-11) % 5 = -1
```

```
    -2
5 ) -11
    -10
    ----
    -1
```

单选题

（-11）%（-5）的运算结果是什么？

A -2

B 2

C -1

D 1

提交

课后思考题

* 如何计算正余数？

```
11 % 5 = 1
```

```
    2
5 | 11
    10
    --
    1
```

```
11 % (-5) = 1
```

```
    -2
-5 | 11
    10
    --
    1
```

```
(-11) % 5 = -1
```

```
    -2
5 | -11
    -10
    ---
    -1
```

求余运算有何用？

- 分离个位、十位、百位数字

  1234%10 = 4

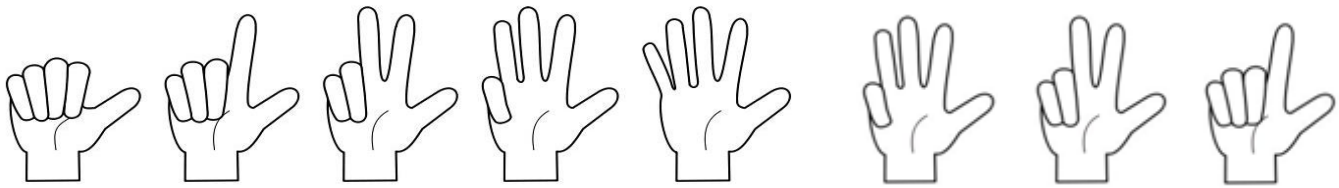
  1234%100 = 34

- 判断n能被m整除

* 判断n是偶数

讨论题

- 一个小女孩正在用左手手指数数，从1到1000。她从拇指算作1开始数起，然后食指为2，中指为3，无名指为4，小指为5。接下来调转方向，无名指算作6，中指为7，食指为8，大拇指为9，接下来食指算作10，如此反复。问如果继续这种方式数下去，最后结束时是停在哪根手指上？



手指	大拇指	食指	中指	无名指	小指	无名指	中指	食指
计数	1	2	3	4	5	6	7	8
计数	9	10	11	12	13	14	15	16
计数	17	18	19	20	21	22	23	24
计数	25	26	27	28	29	30	31	32

求余运算还有什么用？

- 将一个大集合映射到一个只有p个元素的小集合上
  - 对12求余：H(k) = k % 12
- 生活中的例子
  - 一天是24小时，一个星期为7天，一年是12个月
- 生成一个指定范围（如1~100）内的随机数
  - magic = rand() % 100 ;　　//0~99
  - magic = rand() % 100 + 1; //1~100
  - 随机函数rand()
    - 生成一个在0~32767之间的随机数

讨论题

- 若要将n个数装到10个箱子里，随便给一个数，如何能快速知道它装在哪个箱子中？

如何进行更复杂的数学运算？

- 常用的标准数学函数
- #include <math.h>
  - pow函数
    - 当x=0 而y<0或x<0而y不为整数时，结果错误
    - 要求参数x和y及函数返回值为double类型，否则有可能出现数值溢出
    - 千万不要对其进行强转，否则会发生精度损失

函数名	功能
exp(x)	e^x
pow(x,y)	x^y
sqrt(x)	x的平方根，(x>=0)
fabs(x)	|x|
log(x)	lnx，(x>0)
log10(x)	lgx，(x>0)
sin(x)	sinx，x为弧度值
cos(x)	cosx，x为弧度值

课上练习

- 数位拆分与计算V1.0
  - 从键盘输入一个4位数的正整数n，计算并输出拆分后的两个数a和b的加、减、乘、除和求余的结果。
- 数位拆分与计算V2.0
  - 忽略负号
  - 浮点数除法，结果保留小数点后2位小数
- 数位拆分与计算V3.0
  - 对浮点数除法结果进行四舍五入取整

课后练习

- 数位拆分与计算V4.0
  - 要求求余计算的是正余数
- 数位拆分与计算V5.0
  - 考虑除数为0
- 数位拆分与计算V6.0
  - 输出一个菜单，用户选择输入1~5，选择不同的运算