

Variables, Types, IO

王慧妍

why@nju.edu.cn

南京大学



计算机科学与技术系



计算机软件研究所



本讲概述

Variables (变量) Constants (常量)

Data Types (数据类型)

Operators (运算符) Expressions (表达式)

Assignment Statements (赋值语句)

I/O (Input/Output; 输入输出)

回顾Hello world程序

```
Hello world!  
12+22=34
```



```
#include <stdio.h>  
  
int main(){  
    printf("Hello world!\n");  
    printf("12+22=%d\n", 12+22);  
    return 0;  
}
```

printf

```
int printf(const char *__restrict__ _Format, ...)
```

- <stdio.h>
- 显示格式串的程序
 - `printf("Hello world!\n");`
 - `printf("11+22=%d", 11+22);`
- 仅仅输出是不够的

计算任务

- `printf("100*22*33=%d", 100*22*33);`
- 如何每次程序源码不动，根据输入完成计算任务？
 - 对于任意输入数字求找零？
 - 对于输入多个数字求乘积？ 体积？

需要：

1. 有地方放输入的数字；
2. 有办法输入数字；
3. 输入的数字能参与计算；
4. 按需输出。

change.c

```
int price = 0;

printf("Please enter the price (yuan): ");
scanf("%d", &price);

int change = 100 - price;

printf("Change: %d yuan.\n", change);

return 0;
```

变量

- 变量是用来存储数据的单元
 - `int price = 0;`
 - 定义了一个变量，名称为price，类型是int，初始值为0

```
int price = 0;
printf("Please enter the price (yuan): ");
scanf("%d", &price);
int change = 100 - price;
printf("Change: %d yuan.\n", change);
return 0;
```

变量定义与初始化

- 变量定义的一般形式:

- <类型名称> <变量名称>;
- `int price;`
- `int amount;`
- `int price, amount;`

```
int price = 0;
```

- 声明与赋值

- `int price;`
- `price = 0;` //赋值运算符=
- `int price = 0;`

变量的命名

```
int price = 0;
```

- 标识符(identifier): 为变量、函数、数据类型等起名
 - 可以是字母 (A-Z, a-z)、数字 (0-9)、下划线 (_), 但第一个字符不能是数字
 - 合法: price, Price, App1_2, _name999, resourceList, app_list
 - 非法: am*, 1name, get-next-char
 - 见名知意!
- 长度要求
 - ANSI: 外部标识符6字符区分, 内部标识符31字符区分
 - C89: 外部标识符6字符区分, 内部标识符31字符区分
 - C99: 外部标识符31字符区分, 内部标识符63字符区分
- 不可以和关键字重复

变量的命名

- 标识符(identifier): 为变量、函数、数据类型等起名
 - 不可以和关键字重复

对C编译器有特殊意义

32+5+7=44

关键字

auto	extern	short	while
break	float	signed	<code>_Alignas</code>
case	for	sizeof	<code>_Alignof</code>
char	goto	static	<code>_Atomic</code>
const	if	struct	<code>_Bool</code>
continue	<code>inline</code>	switch	<code>_Complex</code>
default	int	typedef	<code>_Generic</code>
do	long	union	<code>_Imaginary</code>
double	register	unsigned	<code>_Noreturn</code>
else	<code>restrict</code>	void	<code>_Static_assert</code>
enum	return	volatile	<code>_Thread_local</code>

从C99开始引入 从C11开始引入

变量类型

```
int price = 0;
```

- 类型int

- 整型：存储整数
- 带取值范围，通常是 -2^{31} ~ $2^{31}-1$
- INT_MAX, INT_MIN

```
#include <limits.h>
printf("MAX: %d, MIN: %d", INT_MAX, INT_MIN);
```

- 更多的变量类型

- double, float可以用来表述小数
 - float scale = 10.5f;
 - double scale = 10.5

```
int price;
int change;
```

需要：

1. 有地方放输入的数字；
2. 有办法输入数字；
3. 输入的数字能参与计算；
4. 按需输出。

scanf

- 用于读入标准输入的数据
 - `scanf("%d", &price);`

```
int price = 0;
printf("Please enter the price (yuan): ");
scanf("%d", &price);
int change = 100 - price;
printf("Change: %d yuan.\n", change);
return 0;
```

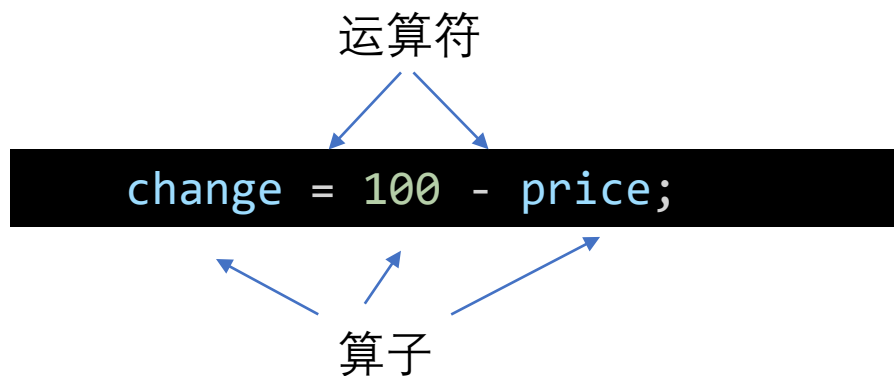
需要：

1. 有地方放输入的数字；
2. 有办法输入数字；
3. 输入的数字能参与计算；
4. 按需输出。

表达式

```
int change = 100 - price;
```

- 表达如何计算值的公式，由运算符和算子组成



表达式

```
int change = 100 - price;
```

- 表达如何计算值的公式，由运算符和算子组成
- 算术运算符
 - 加+，减-，乘*，除/（得到商），取余%（得到余数）
 - /0或%0会产生UB（未定义行为）

```
int hour1, hour2;  
int min1, min2;  
  
scanf("%d %d", &hour1, &min1);  
scanf("%d %d", &hour2, &min2);
```

表达式

```
int change = 100 - price;
```

- 表达如何计算值的公式，由运算符和算子组成
- 算术运算符
 - 加+，减-，乘*，除/（得到商），取余%（得到余数）
 - /0或%0会产生UB（未定义行为）

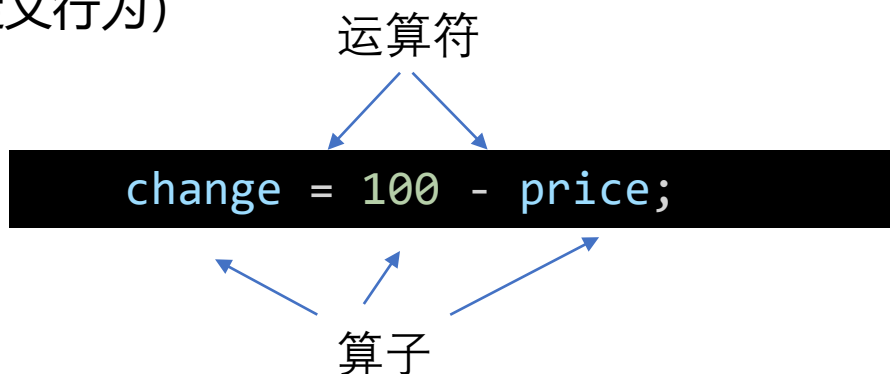
```
int hour1, hour2;  
int min1, min2;  
  
scanf("%d %d", &hour1, &min1);  
scanf("%d %d", &hour2, &min2);  
int t1 = hour1 * 60 + min1;  
int t2 = hour2 * 60 + min2;  
int t = t2 - t1;  
  
printf("interval: %d h %d min", t/60, t%60);
```

表达式

```
int change = 100 - price;
```

- 表达如何计算值的公式，由运算符和算子组成
- 算术运算符
 - 加 $+$ ，减 $-$ ，乘 $*$ ，除 $/$ （得到商），取余 $%$ （得到余数）
 - $/0$ 或 $%0$ 会产生UB（未定义行为）

- 赋值运算符
 - 变量为左值
 - 简单赋值： $=$



- 运算符优先级

优先级	运算符	运算	结合关系	举例
1	$+$	单目不变	自右向左	$a^{*}+b$
1	$-$	单目取负	自右向左	$a^{*}-b$
2	$*$	乘	自左向右	$a^{*}b$
2	$/$	除	自左向右	a/b
2	$%$	取余	自左向右	$a\%b$
3	$+$	加	自左向右	$a+b$
3	$-$	减	自左向右	$a-b$
4	$=$	赋值	自右向左	$a=b$

- 嵌入式赋值，不易理解

```
result = a = b = 3 + c;  
result = 2;  
result = (result = result * 2) * 6 * (result = 3 + result);
```

常见任务

- 交换两个变量
 - 原来: $a = 5, b = 6$
 - 期望: $a = 6, b = 5$
- 三位数逆序
 - 输入: 123
 - 输出: 321

表达式

```
int change = 100 - price;
```

- 复合赋值：+=, -=, *=, /=, %=
 - total += 2
 - total = total + 2
- 自增自减运算符，算子必须是变量
 - 自增++, 自减--
 - count++
 - count = count + 1
 - ++count
 - count = count + 1
 - 不要组合进表达式
 - a = b += c++-d+--e/-f
 - ??????????

a++的值是+1前的值，++a的值是+1后的值，怎么理解？

printf

```
int printf(const char *__restrict__ _Format, ...)
```

- <stdio.h>
- 参数：格式串，可变长的参数列表
- 格式串包括普通字符和以%开始的转换说明

需要：

1. 有地方放输入的数字；
2. 有办法输入数字；
3. 输入的数字能参与计算；
4. 按需输出。

略复杂一点的例子 [circle.c](#)

- 输入半径 r ，自动计算并输出
 - 以 r 为半径的圆的周长 C 和面积 A
 - $Circu = 2\pi r$, $Area = \pi r^2$
 - 以 r 为半径的球的表面积 S 和体积 V
 - $Surface = 4\pi r^2$, $Volume = \frac{4}{3}\pi r^3$
- 假设，半径 r 为float可以表示的小数， π 取3.14
 - `#define PI 3.14`
 - `const double PI = 3.14 //still a variable`

printf

```
int printf(const char *__restrict__ _Format, ...)
```

- <stdio.h>
- 参数：格式串，可变长的参数列表
- 格式串包括普通字符和以%开始的转换说明

转换说明	传入数据类型	输出
%d (%i)	int	decimal ([-]dddd)
%f	double	decimal ([-]ddd.ddd)
%e (%E)	double	decimal ([-]d.ddde[+-]dd)
%g (%G)	double	%f or %e
%c	int	character
%s	pointer to a char array	string
%%		%

printf

`%[flags][minimalwidth][.precision]specifier`

- 转换说明可以控制以特定输出格式输出
 - `%m.pX`, `%-m.pX` (强制左对齐)
- `m`: 最小栏宽, 指定最少需要显示的字符数量
 - 以空格补全
- `p`: 根据转换说明`specifier`类型控制精度
 - `%d`: 最少显示的数字个数, 必要时补0
 - `%e`: 科学计数法显示, 指明小数点后应该出现的数字个数
 - `%f`: 指明小数点后应该出现的数字个数
 - `%g`: 所有可以显示的有效字符的最大数量

```

#include<stdio.h>
int main(){
    int i = 40;
    float x = 839.21f;
    printf("|%d|%5d|%-5d|%5.3d|\n", i, i, i, i);
    printf("|%10.3f|%10.3e|%-10g|\n", x, x, x);
    return 0;
}

```

```

|40|    40|40    |   040|
|   839.210|8.392e+002|839.21  |

```


略复杂一点的例子 [circle.c](#)

- 输入半径 r ，自动计算并输出
 - 以 r 为半径的圆的周长 C 和面积 A
 - $Circu = 2\pi r$, $Area = \pi r^2$
 - 以 r 为半径的球的表面积 S 和体积 V
 - $Surface = 4\pi r^2$, $Volume = \frac{4}{3}\pi r^3$
- 格式化输出
 - 小数点后保留四位
 - 每个结果占15个字符，左对齐
 - _____: circumference
 - _____: area
 - _____: surface
 - _____: volume

mol

- 6g氧气的物质的量是多少?
 - $Q = 6/32 \times 6.02 \times 10^{23}$
 - 格式化输出, 科学计数法
 - 第一遍结果 (小数点后保留3位)
 - 第二遍结果 (保留5位有效数字)
- [mol.c](#)

Data Types

- `int` ($\approx \mathbb{Z}$)
- `double/float` ($\approx \mathbb{R}$)
- `char` (字符)
- `C string` (字符数组)

A naïve administration system

- 录入信息

- Name (EN) Gender (F/M)
- Birthday (mm-dd-yyyy) Weekday (Xyz.)
- C_oj C_midexam C_finalexam
- Score (.d) Ranking (%)

- 录入要求

- 每组信息占一行
- 各项信息使用 "TAB" 间隔
- 各项信息要遵循特定格式要求

Xiaoming WANG	M
01-01-2000	Sat.
100 85	90
93.5 10%	

- [admin.c](#) [admin1.c](#)

char

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_ 0	NUL 0000	SOH 0001	STX 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007	BS 0008	HT 0009	LF 000A	VT 000B	FF 000C	CR 000D	SO 000E	SI 000F
1_ 16	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017	CAN 0018	EM 0019	SUB 001A	ESC 001B	FS 001C	GS 001D	RS 001E	US 001F
2_ 32	SP 0020	! 0021	" 0022	# 0023	\$ 0024	% 0025	& 0026	' 0027	(0028) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3_ 48	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4_ 64	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5_ 80	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C] 005D	^ 005E	_ 005F
6_ 96	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7_ 112	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	 007C	} 007D	~ 007E	DEL 007F

cctype.h

- isdigit isalpha isalnum
- islower isupper tolower toupper
- isspace (, \n, \t)

C String: admin.c

- A C string is an array of characters.
- `char first_name[] = "Xiaoming";`
 - 'X', 'i', 'a', 'o', 'm', 'i', 'n', 'g', '\0'
 - '\0': terminating null character
- `char first_name[10] = "Xiaoming";`
- `char first_name[20] = "Xiaoming";`
- `char first_name[5] = "Xiaoming";`

scanf

```
int scanf(const char *__restrict__ _Format, ...)
```

- <stdio.h>
- 参数：格式字符串，可变长的参数列表
- 以%开始的转换说明

空白字符：包括空格符，水平和垂直制表符/换页符和换行符

转换说明	传入参数	输出
%d	跳过空白字符，匹配int	pointer to int
%le, %lf, %lg	跳过空白字符，匹配double	pointer to double
%e, %f, %g	跳过空白字符，匹配float	pointer to float
%c	a character	pointer to a char
%s	a sequence of non-white-spaces	pointer to a char array
%[abc]	scanlist	pointer to a char array
%[^abc]	not in scanlist	pointer to a char array

How to use scanf...

- [admin1.c](#)
- [scanf_exp.c](#)
- [scanf_exp2.c](#)

Do not use scanf, use ...

End.
