计算机导论与程序设计 [CS006001]

段江涛

机电工程学院

西安电子科技大学 XIDIAN UNIVERSITY

2022 年 9 月

Outlines

I 课程介绍, C 语言简介, 数据输入输出, 基本数据类型与表达式

II 选择结构程序设计

III 循环结构程序设计

IV 数组

V 函数

VI 指针

VI 结构体

Part I

课程介绍, C 语言简介, 数据输入输出, 基本数据类型与表达式

Outlines

1 课程介绍

2 导论简介

3 C 语言程序设计简介

4 开发工具

5 数据类型

6 数据的输入输出

7 运算符和表达式

8 数学库函数

9 顺序程序设计举例

课程内容

- 计算机导论：了解计算机的基本知识；掌握计算机操作基本技能。
- 程序设计：掌握结构化程序设计方法，训练程序逻辑思维能力。会读、会编、会调试 C 语言程序。
- 学习方法：线上、线下相结合。认真书写课堂笔记，按时完成上机练习作业，鼓励大量编程练习。
- 教材
  - 大学计算机，龚尚福，贾澎涛，西安电子科技大学出版社
  - C 程序设计第五版，谭浩强，清华大学出版社
- 线上参考课程资源链接：online resource.pdf

线上导论部分学习内容

1. 计算机历史、现状、发展趋势与前沿技术概述
2. 计算机体系结构及其编码方式
3. 计算机组成与软件系统
4. 计算机应用实践

考核

1. 平时成绩: 10% 由上机练习, 课堂讨论等部分组成。
2. 导论部分: 20% 结合线上资源, 自学字处理软件。总结知识点、课堂笔记, 撰写课程学习报告。
3. 期中考试: 30% 根据机试系统给出的题目编写程序, 通过调试得到正确结果并通过**机试系统提交**。
4. 期末考试: 40% 根据机试系统给出的题目编写程序, 通过调试得到正确结果并通过**机试系统提交**。

计算机导论主要内容

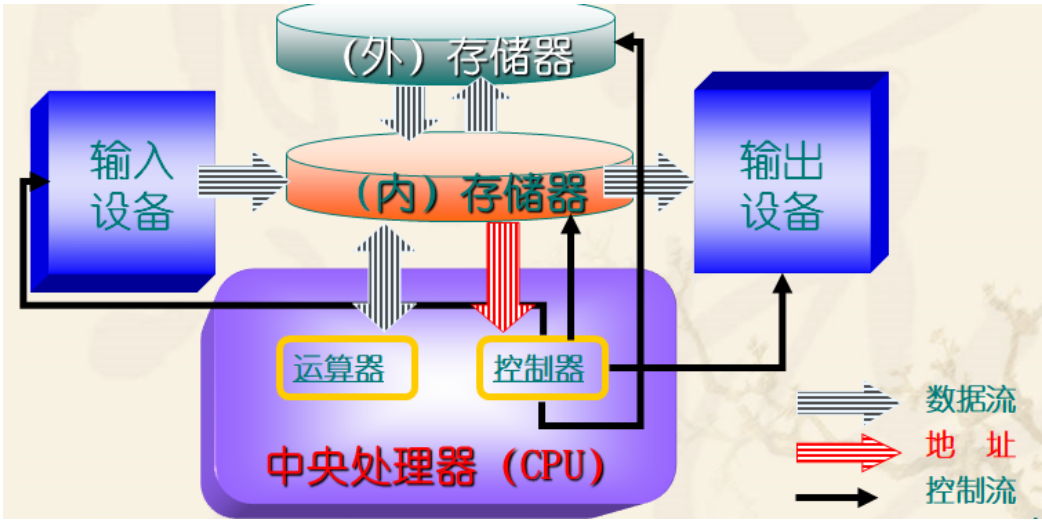
总体要求：了解计算机的基本知识；掌握计算机操作基本技能。

- 计算机系统组成
- 计算机工作原理
- 操作系统
- 字处理：Microsoft Word
- 电子表格：Microsoft Excel
- 演示文稿：Microsoft PowerPoint

计算机工作原理

工作原理: “存储程序” + “程序控制”

1. 以**二进制**形式表示数据和指令
2. 将程序存入存储器中, 由控制器自动读取并执行
3. 外部存储器存储的程序和所需数据 $\Longrightarrow$ 计算机内存 $\Longrightarrow$ 在程序控制下由 CPU 周而复始地取出指令、分析指令、执行指令 $\Longrightarrow$ 操作完成。



计算机程序

指令

可以被计算机理解并执行的基本操作命令。

程序

一组计算机能识别和执行的指令。
一个特定的指令序列用来完成一定的功能。

软件

与计算机系统操作有关的计算机程序、规程、规则，以及可能有的文件、文档及数据。

计算机语言

机器语言

计算机能直接识别和接受的二进制代码称为**机器指令**。机器指令的集合就是该计算机的**机器语言**。

特点：难学，难记，难检查，难修改，难以推广使用。依赖具体机器难以移植。

```
B8  7F 01
BB  21 02
03   D8
B8  1F 04
2B  C3
```

汇编语言

机器语言的符号化。用英文字母和数字表示指令的**符号语言**。

特点：相比机器语言简单好记，但仍然难以普及。

汇编指令需通过**汇编程序**转换为机器指令才能被计算机执行。依赖具体机器难以移植。

```
MOV  AX  383
MOV  BX  545
ADD   BX  AX
MOV  AX  1055
SUB    AX  BX
```

高级语言

高级语言更接近于人们习惯使用的自然语言和数学语言。

特点：功能强大，不依赖于具体机器。用高级语言编写的**源程序**需要通过**编译程序**转换为机器指令的**目标程序**。

```
int  x =1055, y = 383, z = 545
int S;
S = x-(y+z);
S=1055-(383+545)
```

高级语言的发展

结构化语言

规定：

程序必须由具有良好特性的基本结构(**顺序结构、选择结构、循环结构**)构成，程序中的流程不允许随意跳转，程序总是由上而下顺序执行各个基本结构。

特点：

程序结构清晰，易于编写、阅读和维护。

C 语言的特点

1. 语言简洁、紧凑,使用方便、灵活
2. 运算符丰富
3. 数据类型丰富
4. C 语言是完全模块化和结构化的语言

   具有结构化的控制语句 (顺序、选择、循环结构)

   用函数作为程序的模块单位,便于实现程序的模块化
5. 兼具高级语言和低级语言的功能

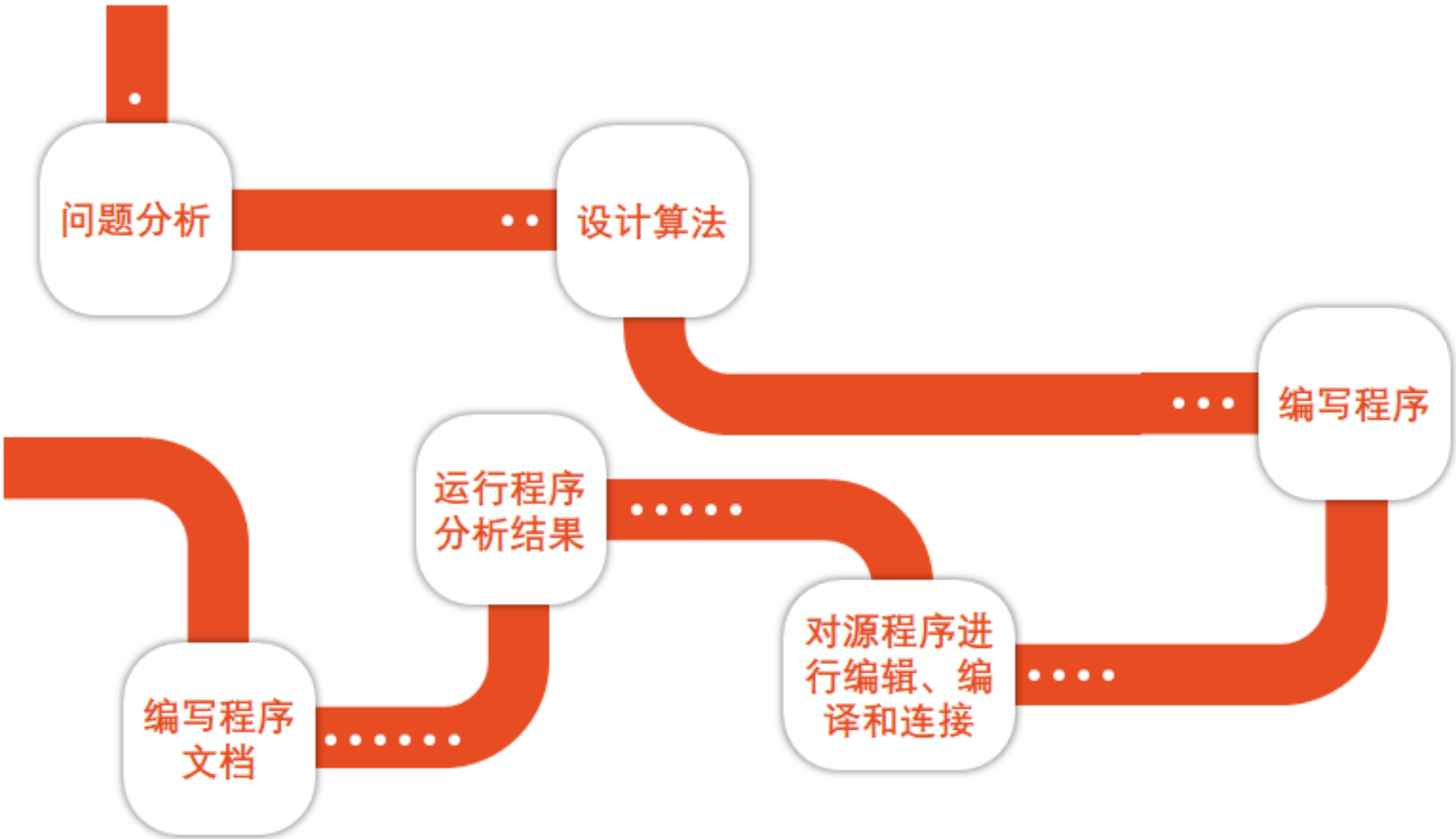
   允许直接访问物理地址

   能进行位 (bit) 操作

   能实现汇编语言的大部分功能

   可以直接对硬件进行操作

程序设计的任务



第一个 C 语言程序

```c
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
   printf("Hello World!"); // printf函数，输出一行信息
   return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

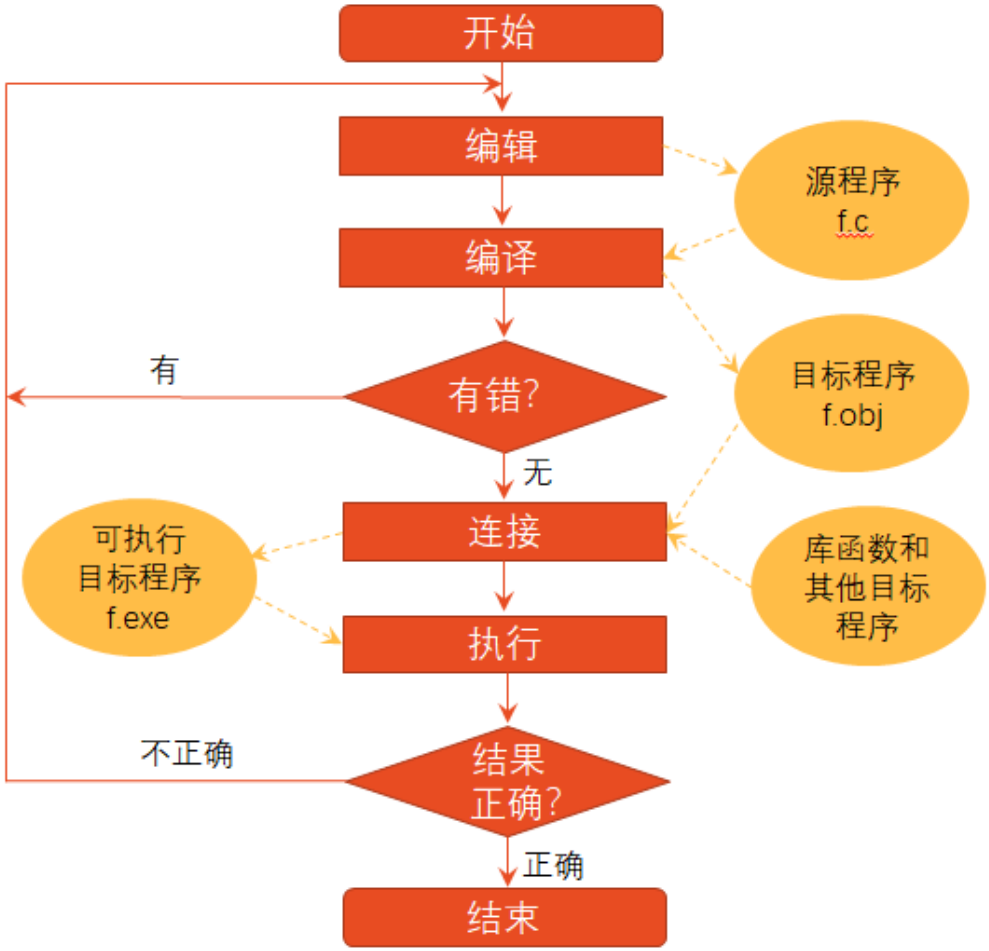
求两个整数之和

```
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
   int a,b,sum; // 定义a,b,sum为整型变量
   a=123; // 对变量a赋值
   b=456;     // 对变量b赋值
   sum=a+b; // 计算a+b, 并把结果存放在变量sum中
   printf("sum is %d\n",sum);// printf函数, 输出结果
   return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

求 5! 的 C 语言程序

```c
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
    int i,p; // p表示被乘数, i表示乘数
    p=1; // 对变量p赋值
    i=2; // 对循环变量i赋值
    while(i<=5) // 循环结构
    {
        p=p*i; // =右端p记录本语句执行前的值, 左端p记录执行后p的值。
        i++; // i = i + 1
    }
    printf("%d\n",p); // printf函数, 输出p的计算结果
    return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

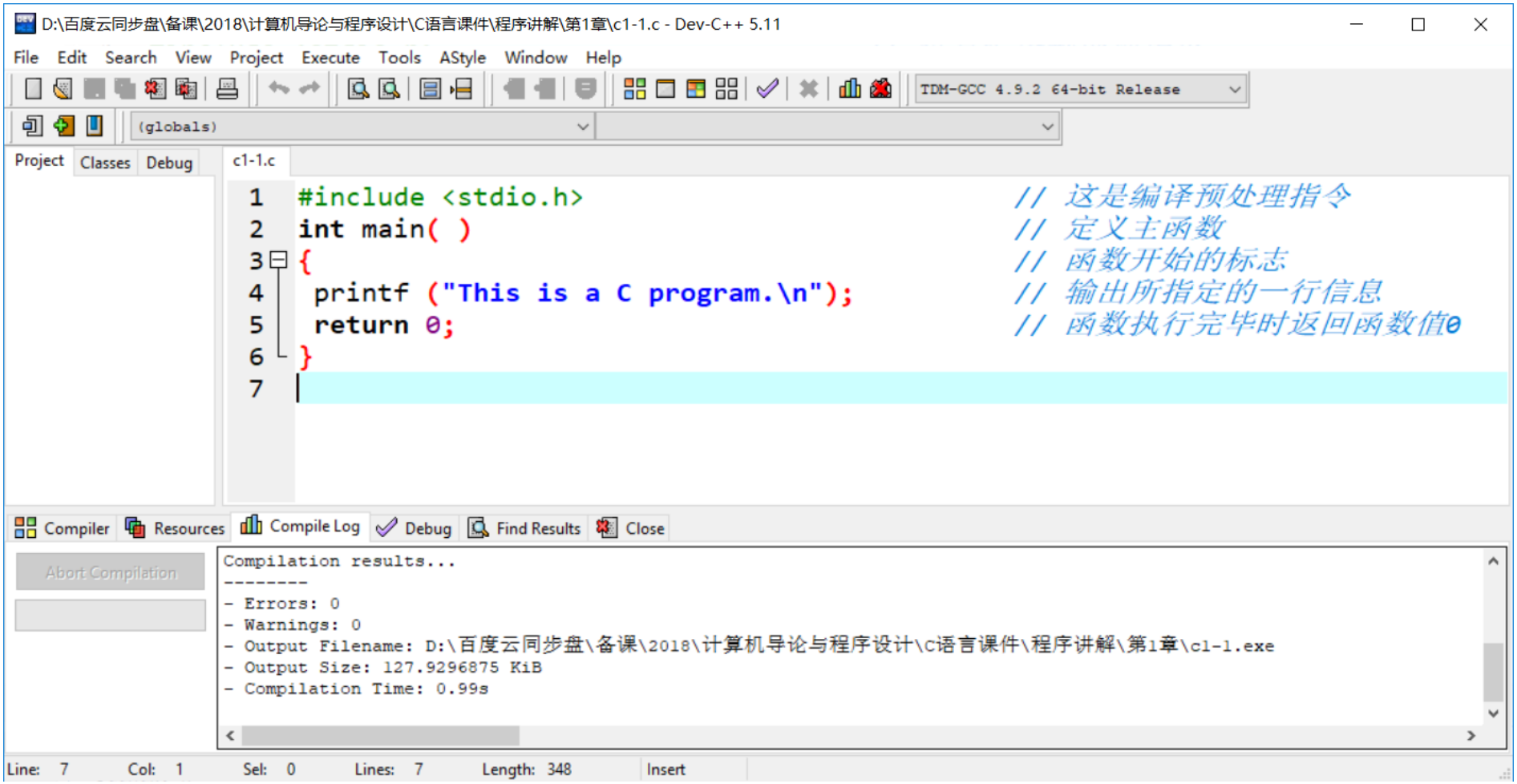
运行 C 程序的步骤与方法



集成开发环境—编译系统

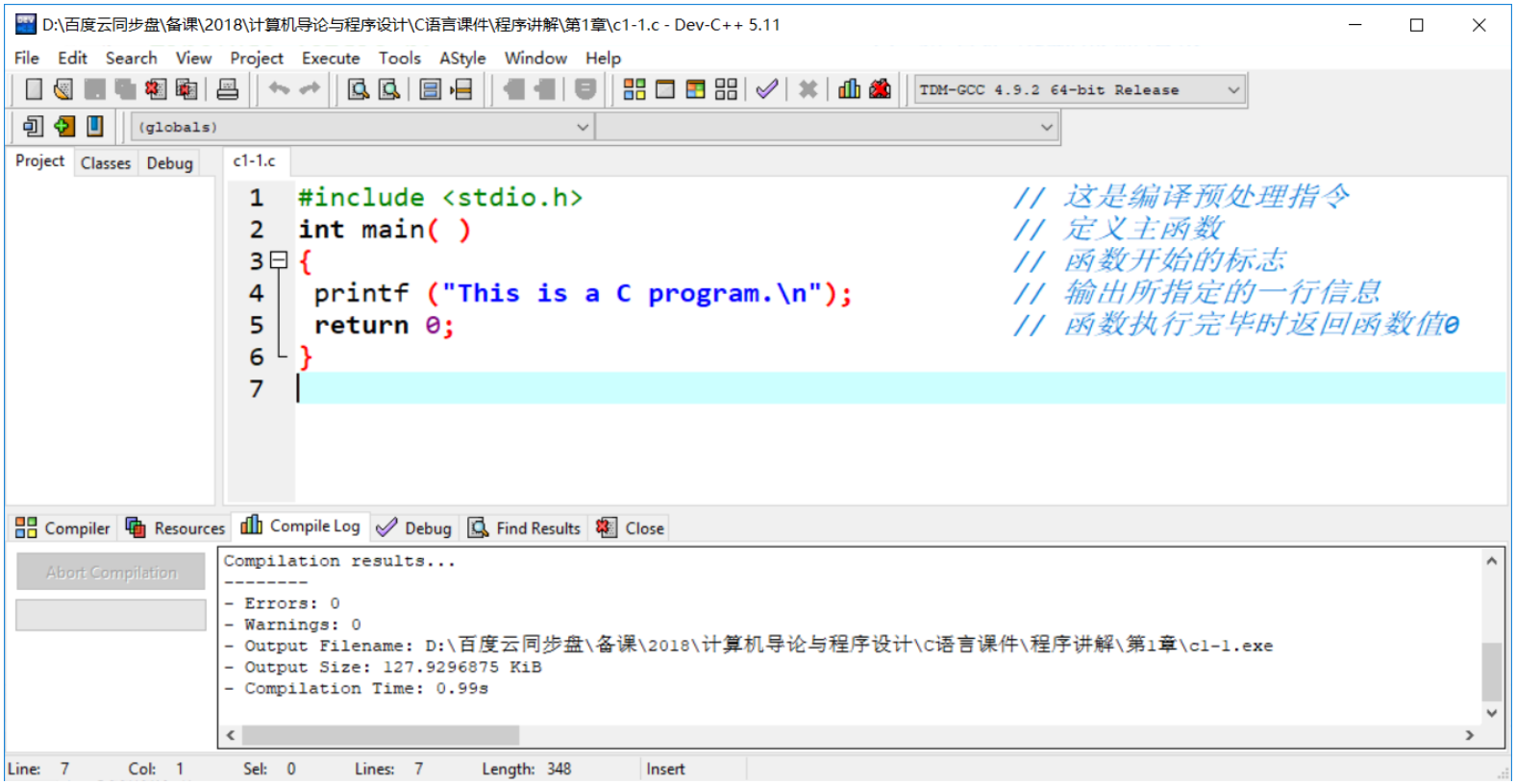
- Bloodshed Dev-C++
- Turbo C
- Visual C++6.0
- Visual Studio(VS2015, VS Community 2019 等)

Bloodshed Dev-C++ 集成开发环境



Bloodshed Dev-C++ 集成开发环境

- 选择“文件”菜单，选择“源文件”，编辑程序。
- 保存时，保存为.cpp 或.c 文件。
- 选择“编译和运行”菜单，生成.exe 文件，运行程序。



基本数据类型

- 整数`int`
- 单精度浮点数`float`
- 双精度浮点数`double`
- 字符`char`

机内用二进制表示, 不同数据类型占用存储空间大小不同。

变量在使用之前首先要定义它的数据类型

```
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
    int a,b; // 定义变量a, b为整型数值, 同类型变量可以在一条语句中定义。
    float f; // 定义变量f为单精度浮点数
    double d; // 定义变量d为双精度浮点数
    char c; // 定义变量c为单个英文字母
    a=10; // 变量赋值
    b=20;
    f=10.2;
    d=20.3;
    c='A'; // 字符用单引号括起来
    return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

标识符

标识符就是一个对象的名字。用于标识变量、符号常量、函数、数组、类型等。

以字母或下划线开始; 区分大小写; 不能使用关键字; 最好有含义。

```
#include<stdio.h>
int main()
{
    int r = 123; // 合法整型变量名
    int 3a; // 不合法的变量名
    int break; // 不合法的变量名, 因为break是关键字, 被系统使用。
    int Radius; // 变量名最好有含义
    int radius; // 与Radius是不同的变量, C语言是到小写敏感的语言
    return 0;
}
```

C 语言关键字

auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

输出语句 printf(“原样输出, % 格式符”, 对应变量值);

```
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
    int a=10,b; // 定义变量a, b为整型数值, 定义变量时, 可以指定变量的初值
    float f=10.2; // 定义变量f为单精度浮点数
    double d; // 定义变量d为双精度浮点数
    char c; // 定义变量c为单个英文字母
    f=10.2;
    d=20.356;
    c='A';
    printf("a=%d,b=%d,c=%c,f=%f,d=%.2lf\n",a,b,c,f,d); // %.2f, %.2lf保留
        两位小数, \n为换行符
    return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

变量 b 没有被赋值, 将是一个随机值。

常用格式描述符与数据类型的对应关系

格式符	对应的数据类型	备注
%d	int	
%f	float	
%c	char	
%lf	double	
%.2f	float	保留两位小数, 四舍五入。不适用于 scanf()。
%.2lf	double	保留两位小数, 四舍五入。不适用于 scanf()。
%x	int	十六进制显示
%ld	long int	

详见 p73, 表 3.6

十进制与二进制

十进制: 以 10 为底的幂展开式:

$(123)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0;$

自低到高各位数 (除 10 取余至商为 0): $3 = 123\%10$, $2 = 123/10\%10 = 12\%10$,

$1 = 123/10/10\%10 = 1\%10$

二进制: 以 2 为底的幂展开式:

$$(77)_{10} = (0100 \quad 1101)_2 = 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4$$
$$+ 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

自低到高各位数 (除 2 取余至商为 0): $1 = 77\%2$, $0 = 77/2\%2 = 38\%2$,

$1 = 77/2/2\%2 = 19\%2$, $1 = 77/2/2/2\%2 = 9\%2$,

$0 = 77/2/2/2/2/\%2 = 4\%2$, $0 = 77/2/2/2/2/2/\%2 = 2\%2, \ldots$

10 进制、2 进制、16 进制的幂展开式

$$(D)_{10} = D_{n-1} \times 10^{n-1} + D_{n-2} \times 10^{n-2} + \cdots + D_1 \times 10^1 + D_0 \times 10^0 + D_{-1} \times 10^{-1} + D_{-2} \times 10^{-2} + \cdots + D_{-m+1} \times 10^{-m+1} + D_{-m} \times 10^{-m}$$

$$(B)_2 = B_{n-1} \times 2^{n-1} + B_{n-2} \times 2^{n-2} + \cdots + B_1 \times 2^1 + B_0 \times 2^0 + B_{-1} \times 2^{-1} + B_{-2} \times 2^{-2} + \cdots + B_{-m+1} \times 2^{-m+1} + B_{-m} \times 2^{-m}$$

$$(H)_{16} = H_{n-1} \times 16^{n-1} + H_{n-2} \times 16^{n-2} + \cdots + H_1 \times 16^1 + H_0 \times 16^0 + H_{-1} \times 16^{-1} + H_{-2} \times 16^{-2} + \cdots + 16_{-m+1} \times 16^{-m+1} + H_{-m} \times 16^{-m}$$

进制对照表 $2^3 2^2 2^1 2^0 = 8 + 4 + 2 + 1$

十进制	二进制	十六进制	十进制	二进制	十六进制
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

十进制、二进制与十六进制举例

$$(123)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0;$$

自低到高各位数: $3 = 123\%10,\ 2 = 123/10\%10,\ 1 = 123/10/10\%10$

$$(77)_{10} = (0100 \quad 1101)_2 = 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4$$
$$+ 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$(77)_{10} = (4D)_{16} = 4 \times 16^1 + 13 \times 16^0$$

数值在计算机中的表示 (以 8bit 编码为例)

- 原码:正数的符号为 0,负数的符号为 1,其它位按一般的方法表示数的绝对值。

$$x = (+103)_{10} \qquad [x]_{原} = (01100111)_2$$

$$x = (-103)_{10} \qquad [x]_{原} = (11100111)_2$$

- 反码: 正数的反码与原码相同;负数的反码是符号位不变,其他位按位取反
- 补码: 正数的补码与其原码相同;负数的补码为其反码最末位加 1. 即,

负数补码 = 反码 +1 = 2^n − 该数的绝对值, n 是编码二进制位数.

$$(77)_{10} = (0100\quad 1101)_2, \qquad (-77)_{10} = (1100\quad 1101)_2$$

$$(-77)_{补} = 2^8 - 77 = 1111\quad 1111 + 0000\quad 0001 - 0100\quad 1101$$

$$= \underbrace{1111\quad 1111 - 0100\quad 1101}_{(-77)_{反}} + 0000\quad 0001$$

$$= \underbrace{1011\quad 0010}_{(-77)_{反}} + 0000\quad 0001 = 1011\quad 0011$$

数值表示示例


+77
原码
0 1 0 0 1 1 0 1
反码
0 1 0 0 1 1 0 1
补码
0 1 0 0 1 1 0 1
-77
原码
1 1 0 0 1 1 0 1
反码
1 0 1 1 0 0 1 0
补码
1 0 1 1 0 0 1 1

机内以补码形式存储有符号数

1. 对于正数，原码 = 反码 = 补码
2. 对于负数，补码 = 反码 + 1

   反码 = 符号位不变，其他位按位取反
3. 补码是可逆的，即再对补码求补得到原码。
4. 引入补码后，使减法统一为加法。

   $(+77)_{补} + (-77)_{补} = 0100\ 1101 + 1011\ 0011 = 0000\ 0000$

补码运算实例 (以 8bit 编码为例)

补码可逆:

$$[-25]_{原} = (1001\ 1001)_2 \quad [-25]_{反} = (1110\ 0110)_2$$

$$[-25]_{补} = [-25]_{反} + 1 = (1110\ 0110 + 1)_2 = (1110\ 0111)_2$$

$$[-25]_{原} = ([-25]_{补})_{补} = (1001\ 1000 + 1)_2 = (1001\ 1001)_2$$

减法统一为加法: $[a-b]_{补} = a_{补} + [-b]_{补}$

$$[102 - 25]_{补} = [77]_{补} = (0100\ 1101)_2 = 77$$

$$[102]_{补} + [-25]_{补} = (0110\ 0110)_2 + (1110\ 0111)_2 = (0100\ 1101)_2 = 77$$

所以, $[102 - 25]_{补} = [102]_{补} + [-25]_{补}$

同样有, $[25 - 102]_{补} = [25]_{补} + [-102]_{补}$

计算机数据存储单位

位 (bit) 是最小的存储单位，每一位存储 1 位二进制码，一个字节 (Byte) 由 8 位组成。

- 1B = 8b
- 1KB = 1024B
- 1MB = 1024KB
- 1GB = 1024MB
- 1TB = 1024GB

整型数据输出 printf("%d,%x", -25, -25);

```
#include<stdio.h> // standard input/output编译预处理指令
int main()
{
    int a=-25,b=102; // 定义变量a, b为整型数值, 定义变量时, 可以指定变量的初值
    printf("a=%d,b=%d,a=%x,b=%x\n",a,b,a,b);
    printf("%d,%d,%x,%x\n",-25,102,-25,102);
    printf("%d,%d,%x,%x,%X\n", 77,-77,77,-77,-77);
    return 0;
}
```

```
// 编译运行, 解释输出结果。
a=-25,b=102,a=ffffffe7,b=66
-25,102,ffffffe7,66
77,-77,4d,ffffffb3,FFFFFFB3
```

$[-25]_{补} = (1110\ 0111)_2 = \text{E7H (0XE7)}$

$[-77]_{补} = (1011\ 0011)_2 = \text{B3H (0XB3)}$

$66\text{H} = 6 \times 16^1 + 6 \times 16^0 = 102$

$4\text{DH} = 4 \times 16^1 + 13 \times 16^0 = 77$

ASCII 编码表 $B_6B_5B_4B_3B_2B_1B_0$

$B_6B_5B_4$ / $B_3B_2B_1B_0$	000	001	010	011	100	101	110	111
0000	NUL	DLE	空格	0	@	P	、	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	”	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	•	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

- ASCII 码连续排列

  ‘0’～‘9’, ‘A’～‘Z’, ‘a’～‘z’

- 数字 = 编码值 - ‘0’

  9=‘9’-‘0’

- 大小字符间隔:

  ‘a’ - ‘A’ = 32

  ‘a’=0110 0001=61H=0X61=97

  ’A’=0100 0001=41H=0X41=65

字符类型

```c
#include<stdio.h>
int main()
{
    char c1 = 'A', c2 = 'a', c3 = '\n'; // 字符型变量
    printf("%c,%c,%d\n",c1,c2,c3); // A,a,10
    // 整型变量的整数值就是ASCII编码值
    printf("%c,0X%x,%d\n",c1,c1,c1); // A,0X41,65
    c1 = c1 + 1; // 在表达式中, char类型看作int处理
    printf("%c,0X%x,%d\n",c1,c1,c1); // B,0X42,66
    c1 = c1 + 32; // 转换为小写字母
    printf("%c,0X%x,%d\n",c1,c1,c1); // b,0X62,98
    printf("%d\n",'9'-'0'); // 数字 = 编码值- '0'
    printf("%c,%d,%c,%d\n",'A','A','a','a'); // 输出字符和相应的ASIII编码
    return 0;
}
```

常量

```
#include<stdio.h>
#define PI 3.14 // 符号常量, 注意没有分号
int main()
{
    int a = 123; // 整型常量
    float f = 12.2, f1=123E-1; // 实型常量
    char c1 = 'A', c2='\n'; // 字符常量
    char s[50] = "boy"; // 字符串常量
    printf("半径为%d的圆周长是%f\n",a,2*PI*a);
    printf("回车换行\n");
    printf("单引号\',双引号\"转义字符前缀\,\n");
    return 0;
}
```

转义字符, 见 p40, 表 3.1

常量与常变量

```c
#include<stdio.h>
#define PI 3.14 // 符号常量，注意没有分号
int main()
{
    int r = 123; // 整型变量
    const int a = 425; // 常变量
    r = 100; // 合法，因为r是变量，可以随时更改它的值
    a = 100; // 不合法，因为a是常变量，不能更改
    printf("半径为%d的圆周长是%f\n",r,2*PI*r);
    return 0;
}
```

长整型、无符号整型, 浮点型数据类型

```c
#include<stdio.h>
int main()
{
    int a = 123; // 整型变量
    long int b = 1E+8; // 长整型变量
    unsigned int u = 0XFF; // 无符号整型, 最高为不作为符号位处理
    float f = 10.2; // 单精度浮点数
    double d = 1E-8; // 双精度浮点数
    printf("%x,%d\n",u,u); // ff, 255
    printf("%d,%ld,%x,%f,%lf\n",a,b,u,f,d);
    // sizeof函数返回类型分配的字节数, 整型数据存储空间和值的范围见p45, 表3.2
    printf("%d,%d,%d,%d\n",sizeof(int),sizeof(float),sizeof(double),
        sizeof(long int),sizeof(long long int));
    return 0;
}
```

常用格式描述符与数据类型的对应关系

格式符	对应的数据类型	备注
%d	int	
%f	float	
%c	char	
%lf	double	
%.2f	float	保留两位小数, 四舍五入。不适用于 scanf()。
%.2lf	double	保留两位小数, 四舍五入。不适用于 scanf()。
%x	int	十六进制显示
%ld	long int	

详见 p73, 表 3.6

输出语句 printf(“原样输出, % 格式符”, 对应变量值);

```
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
  int a=10,b; // 定义变量a, b为整型数值, 定义变量时, 可以指定变量的初值
  float f=10.2; // 定义变量f为单精度浮点数
  double d; // 定义变量d为双精度浮点数
  char c; // 定义变量c为单个英文字母
  f=10.2;
  d=20.356;
  c='A';
  printf("a=%d,b=%d,c=%c,f=%f,d=%.2lf\n",a,b,c,f,d); // %.2f, %.2lf保留
      两位小数
  return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

变量 b 没有被赋值, 将是一个随机值。

输入语句 scanf(“% 变量格式符”, & 变量名);

```
#include<stdio.h> // standard input/output编译预处理指令
int main() // 主函数
{ // 函数开始标志
  int a=10,b; // 定义变量a, b为整型数值, 定义变量时, 可以指定变量的初值
  float f=10.2; // 定义变量f为单精度浮点数
  double d; // 定义变量d为双精度浮点数
  char c='A'; // 定义变量c为单个英文字母, 字符输入以后讲
  printf("请输入整数和浮点数, 空格隔开:\n"); // 提示语句[可选]
  scanf("%d%f",&a,&f); // 尽量简单, 不要有其它字符和'\n'
  printf("请输入两个浮点数, 空格隔开:\n"); // 提示语句[可选]
  scanf("%f%lf",&f,&d);
  printf("a=%d,b=%d,c=%c,f=%f,d=%lf\n",a,b,c,f,d); // \n为换行符
  return 0; // 函数执行完毕返回函数值0
} // 函数结束标志
```

字符输出函数 putchar

```
#include<stdio.h>
int main()
{
   char a = 'B',b = 'O',c = 'Y'; //定义3个字符变量并初始化
   putchar(a); //向显示器输出字符B
   putchar(b); //向显示器输出字符O
   putchar(c); //向显示器输出字符Y
   putchar ('\n'); //向显示器输出一个换行符
   return 0;
}
```

字符输入函数 getchar, 遇到回车, 开始从缓冲区中接收字符。

```
#include<stdio.h>
int main()
{
  char a,b,c; //定义字符变量a,b,c
  a = getchar(); //从键盘输入一个字符，送给字符变量a
  b = getchar(); //从键盘输入一个字符，送给字符变量b
  c = getchar(); //从键盘输入一个字符，送给字符变量c
  putchar(a); //将变量a的值输出
  putchar(b); //将变量b的值输出
  putchar(c); //将变量c的值输出
  printf("\na=%d,b=%d,c=%d,a=%c,b=%c,c=%c\n",a,b,c,a,b,c);
  return 0;
}
```

字符输入函数 getchar, 遇到回车, 开始从缓冲区中接收字符。

```
char a,b,c; //定义字符变量a,b,c
a = getchar(); //从键盘输入一个字符，送给字符变量a
b = getchar(); //从键盘输入一个字符，送给字符变量b
c = getchar(); //从键盘输入一个字符，送给字符变量c
putchar(a); //将变量a的值输出
putchar(b); //将变量b的值输出
putchar(c); //将变量c的值输出
printf("\na=%d,b=%d,c=%d,a=%c,b=%c,c=%c\n",a,b,c,a,b,c);
```

从键盘输入 abc 回车, 观察结果, 应该是正确的结果。遇到回车, 开始从缓冲区中接收字符。

字符输入函数 getchar, 遇到回车, 开始从缓冲区中接收字符。

```
a = getchar(); //从键盘输入一个字符，送给字符变量a='a'
b = getchar(); //从键盘输入一个字符，送给字符变量b='\n'
c = getchar(); //从键盘输入一个字符，送给字符变量c='b'
putchar(a); //将变量a的值输出a
putchar(b); //将变量b的值输出\n
putchar(c); //将变量c的值输出b
printf("\na=%d,b=%d,c=%d,a=%c,b=%c,c=%c\n",a,b,c,a,b,c);
```

再运行一次程序, 输入 a 回车, 输入 b 回车, 输入 c 回车, 观察结果。

```
a
b
a
b
a=97, b=10, c=98, a=a, b=
, c=b
```

开发平台上演示讲解

在开发平台，以具体的示例，详细讲解以下内容:

- int, float, double, char 数据类型, sizeof( ) 函数
- %d, %f, %c, %lf, %x 格式符的使用 (见 ppt 中的表格)
- if( ){ }, while( ){ } 简单语句
- char c; scanf("%c", &c); 接收输入的字符
- char c; c=getchar() 接收输入的字符, putchar() 输出一个字符
- 编程理解数字 ASCII 码与整数的对应关系以及大小写字符之间的关系。
- 避免数字，字符在一条语句中输入的情况，如:

  scanf("%d%c%d",...);
- 重点理解字符缓冲区的概念，以及消费无用字符的技巧。

算术运算符 +, −, *, /, %, ++, −−

整数 = 整数/整数, 结果不会四舍五入。

```
#include<stdio.h>
int main()
{
   int a=5, b=2; float c=5,d=2,f;
   f = a/b; printf("%f\n",f); // 2.000000
   f = c/d; printf("%f\n",f); // 2.500000
   f = (float)a/b; printf("%f\n",f); // 2.500000
   printf("%f\n",5.0/2); // 2.500000
   printf("%d\n",2a); // 错误
   printf("%d\n",2*a); // 正确
   return 0;
}
```

余数 r=a%b, a,b 必须是整数。

```
#include<stdio.h>
int main()
{
    int a = 123;
    while(a != 0) // 当a不等于0时, 执行循环体。
    {
      printf("%d ",a%10); // 3 2 1
      a = a/10; // 或 a /= 10;
    }
    printf("\n");
    return 0;
}
```

算术运算符 ++, −−

++i, - -i: 先加 (减)1, 再使用。i++, i- -: 先使用, 再加 (减)1

```
#include<stdio.h>
int main()
{
    int a,b=10;
    a = ++b;
    printf("a=%d,b=%d\n",a,b); // a=11,b=11
    a = b++;
    printf("a=%d,b=%d\n",a,b); // a=11,b=12
    a--;
    b--;
    printf("a=%d,b=%d\n",a,b); // a=10,b=11
    return 0;
}
```

数学库函数, 详见附录 E p365.

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

函数原型	功能	调用举例
`int abs(int x);`	求整数 x 的绝对值	`int x=-10,y; y=abs(x);`
`double fabs(double x);`	求浮点数 x 的绝对值	`double x=-0.5,y; y=fabs(x);`
`double sqrt(double x);`	计算 $\sqrt{x}$	`double x=0.5,y; y=sqrt(x);`
`double pow(double x, double y);`	计算 x^y	`double x=0.5,y=2,z; z=pow(x,y);`
`int rand(void);`	产生-90~32767 的随机整数	`int y; y=rand();`
`double log(double x);`	求 $\log_e x$, 即 $\ln x$	`double x=2.0; y=log(x);`
`double log10(double x);`	求 $\log_{10} x$	`double x=2; y=log10(x);`

因为函数内是用二进制计算浮点数的指数值, 注意, 调用 pow 函数使用整型参数有精度问题, 可能得不到正确的值。以后学习函数章节后, 最好编写自己的函数计算整数的指数值。

Example (例 3.5 p64)

求 $ax^2 + bx + c = 0$ 方程的根。a, b, c 由键盘输入，设 $b^2 - 4ac > 0$。

```
#include<stdio.h>
#include<math.h> // 数学库函数
int main()
{
   double a,b,c,x1,x2,delta;
   scanf("%lf%lf%lf",&a,&b,&c);
   if(b*b-4*a*c <= 0) { printf("输入错误!"); return 0; } // 程序结束
   delta = sqrt(b*b-4*a*c);
   x1 = -b + delta/(2*a);
   x2 = -b - delta/(2*a);
   printf("x1=%lf,x2=%lf\n",x1,x2);
   return 0;
}
```

Example (例 3.1 p37)

有人用温度计测量出用华氏法表示的温度 (如 64°F), 今要求把它转换为以摄氏法表示的温度 (如 17.8°C)。

$$c = \frac{5}{9}(f - 32)$$

其中, f 代表华氏温度, c 代表摄氏温度.

特别注意: 整数/整数 = 整数, 不会四舍五入。

Example (例 3.2 p38)

计算存款利息。有 1000 元, 想存一年。有 3 种方法可选:

(1) 活期, 年利率为 r1;

(2) 一年期定期, 年利率为 r2;

(3) 存两次半年定期, 年利率为 r3。

请分别计算出一年后按 3 种方法所得到的本息和。

$$p1 = p_0(1 + r_1), p2 = p_0(1 + r_2), p3 = p_0(1 + \frac{r_3}{2})(1 + \frac{r_3}{2})$$

Example

1. 自定义各变量类型和值, 求 $y = |x^3 + \log_{10} x|$
2. 自定义各变量类型和值, 求 $y = \frac{3ae}{cd}$
3. 自定义各变量类型和值, 求 $y = \frac{ax + \frac{a+x}{4a}}{2}$
4. m 是一个已知 3 位整数, 从左到右用 a, b, c 表示各位数字。
   (1) 求数 bac 的值;
   (2) 计算 m 的最后一个字节;
   (3) 思考, 如果 m 是多位数, 如何计算获取每位数字?

开发平台上演示讲解

- 回顾基本数据类型`int`,`float`,`double`,

  输出输入语句`printf( );scanf();putchar(),getchar()`,

  格式转换符`%d,%f,%lf,%c,%x`, ASCII 编码与整数之间的对应关系。
- 无符号整型

```
unsigned int a=0x85; int b=-5; printf("%X,%X,%d",a,b,b); //补码存储
```

- 定义常数`#define PI 3.14`
- 标识符, 以字母或下划线开始。区分大小写, 不能使用关键字。
- 算术运算符`+,-,*,/,%,++,--`。特别注意: 整数 = 整数/整数, 不会四射五入。
- `++i, --i`: 先加 (减)1, 再使用。`i++, i--`: 先使用, 再加 (减)1
- 数学库函数`int abs(int x); double fabs(double x); double sqrt(double x); doubel pow(double x,double y); double log10(double x)`
- 作业: 练习编程: 上页 Example, 下节课检查。