

Chapter 1 Basic Expression and Math functions

For every program code written in C, almost all have the similar looking.

<pre>#include <stdio.h> int main(){ // Some instructions return 0; };</pre>	<pre>#include <math.h> int main(){ // Some instructions return 0; };</pre>
--	---

The main() stands for the starting point of the program.

All instructions inside the pair of { } are belongs to main().

Whenever a computer runs a program, it finds the main() and performs the instruction from left to right and from up to down until every thing in the main() has been done.

Note: C language is case-sensitive, which means Main() and main() are different.

Arithmetic Expressions:

To ask the computer evaluate some mathematical expression, we have to use the basic operations like addition, subtraction, multiplication and division to help us.

The operators are very similar to mathematics except for multiplication and division. The precedence is also the same as mathematics.

Mathematics	C Language	Description
$7 + 2$	$7 + 2$	Addition
$7 - 2$	$7 - 2$	Subtraction
7×2	$7 * 2$	Multiplication
$\left\lfloor \frac{7}{2} \right\rfloor$	$7 / 2$	Integer division (find quotient, i.e. 3)
$7 \div 2$	$7 / 2.0$	Real division (the answer is 3.5)
$7 \div 2$	$7.0 / 2$	Real division
$7 \div 2$	$7.0 / 2.0$	Real division
$7 \bmod 3$	$7 \% 3$	Remainder of 7 divided by 3 (i.e. 1)
$(1 + 2) \times 3$	$(1+2)*3$	
7^4		
$7^4 \div 5^2$		
	$(1+2*3)/(4.0+2*9)$	
mc^2	$m*c*c$	
$x^4y^7 + xz^3$		

Try to fill in the empty blanks.

Math function in C:

We can use many times of multiplication to find the power of some number.
 But how can we find the square root of any non-negative number?
 Moreover, sin, cos, tan, log, ...

First, you must type `#include <math.h>` at the beginning of your code.
 That **statement** will tell the computer to use some mathematical functions.

Mathematics	C Language	Description
$\cos x$	<code>cos(x)</code>	Get the value of $\cos x$, the angle is measured in radian.
$\sin x$	<code>sin(x)</code>	Get the value of $\sin x$, the angle is measured in radian.
$\tan x$	<code>tan(x)</code>	Get the value of $\tan x$, the angle is measured in radian.
$\cos^{-1} x$	<code>acos(x)</code>	Get the value of $\cos^{-1} x$
$\sin^{-1} x$	<code>asin(x)</code>	Get the value of $\sin^{-1} x$
$\tan^{-1} x$	<code>atan(x)</code>	Get the value of $\tan^{-1} x$
$\log x$	<code>log10(x)</code>	Get the value of $\log x$, the common logarithm.
\sqrt{x}	<code>sqrt(x)</code>	
x^y	<code>pow(x,y)</code>	
	<code>round(x)</code>	Round the number x to the nearest integer.
$\lceil x \rceil$	<code>ceil(x)</code>	Get the smallest integer greater than or equal to x.
$\lfloor x \rfloor$	<code>floor(x)</code>	Get the greatest integer smaller than or equal to x.
$ x $	<code>abs(x)</code>	Get the absolute value of x.

Precedence:

As if in mathematics, we always perform multiplication and then addition.
 There is a similar rule in C language. (Smaller the number, higher the priority)

1	<code>()</code>	Perform from left to right
2	<code>*, /, %</code>	Perform from left to right
3	<code>+, -</code>	Perform from left to right
4	functions	Perform from inner to outer

Left to right means, when some expression like $1-2+3$, we perform $(1-2)$ first.
 Inner to outer means, $\text{sqrt}(4*\text{sqrt}(16))$, we do $\text{sqrt}(16)$ first and $4*\text{sqrt}(16)$.

Try the following:

`floor(sqrt(ceil(9.5+floor(6.6)+1.3-round(2.8))))`

Variables:`int answer=1;``int M;``int x,y,z;`

In the introduction, we have some instructions like the above one.

The left one tells the computer to store 1 to a location called answer.

The middle one tells the computer to prepare a location called M for storage.

The right one tells the computer to prepare 3 locations namely x , y , z for storage.

This kind of things is called **variable**, which is used to store value of some expression so that we can use it later on. Like the “M+” of your calculator which save a number.

But, how does the computer know what kind of value it is going to store??

It can store English words, integers, real number, even true and false.

In C, we will state the **data type** before the name of the variable.

In the above one, the **data type** of **answer** is **int**.

Integer is number 0,1,-1,2,-2,... but not $2.5, \sqrt{2}, \pi$ etc.

Real number is all number including **integer, decimal number, rational, irrationals**.

In computer there is size limitation, we cannot store any number as great as we want.

Also, there is limitation of accuracy and precision, as like your calculator only 9-digit can be displayed for a very long number.

Data type	Meaning	Range
int	Integer	In HKOI (-32768 to 32767)
long int	Integer with bigger size	In HKOI (-2^{31} to $2^{31}-1$)
float	Real number	With less accuracy
double	Real number	More accuracy

From now on, we will switch our word from “instruction” to “statement”.

Statement is a technical term which refers to a single code with semi-colon.

In fact, a statement usually causes a computer to perform multiple instructions.

Names:

We have discussed the **data type**, but is there any rule for naming a variable?

Name must begin with a character or underscore, and may be followed by any combination of characters, underscores, or the digits 0 - 9.

So, the following names are valid.

M1	x	_12	ans	X	_
----	---	-----	-----	---	---

Output:

```
printf("%d", 35);
```

```
printf("%s", "Hi");
```

In the beginning, we use the *printf* statement to display some number and words to the screen.

Before you use this, remember to add the statement.

```
#include <stdio.h>
```

Unlike the previous maths function, the *printf* function can accept multiples **parameters**.

Note: Everything enclosed by "" will be treated as English words in computer. These "English words" are called **string**.

The first one is the **format string**, which tells the computer the format of the output.

Specifier and special characters	Description
%d	integer
%f	real number
%s	string
\n	New line

Those %d, %s stands for a integer value, string value etc.

But what is the actual value of those specifier?

We have to tell the computer as well.

So, the second one will be the first value etc...

Examples:

Statement	Result
printf("%d + %d = ? ", 1 , 2);	1 + 2 = ?
printf("%d + %d %s %d", 3 , 7 , "is" , 10);	3 + 7 is 10

Use of variables:

We can always use the value stored in a variable by using its name.

We can change the content of a variable as well.

```
printf("%d", answer);
```

The assignment operator: =

```
M = 3;
```

This equal sign is not the same as the mathematical equal sign.

It is called the assignment operator and will store the value on right hand side to the variable on the left.

The previous content will disappear immediately, and we have NO way to recall it.

Note: So **variables** in computer may change its value from time to time.

The **variables** in mathematics will never change its value.

First program:

```
#include <stdio.h>
int main(){
    int x;
    int answer;

    x = 10;
    answer = x + 12;
    x = answer + x;
    printf("The value of x is %d \n" , x);
    printf("The value of answer is %d \n" , answer);
    printf("End of first program. \n");
    scanf("%d", &x);                // You can ignore this line
    return 0;
}
```

Input:

Our program can now display values and sentences to the screen.

Can we type in some numbers to the computer and let our program receive them?

Yes, but we need to use **variables** to store them.

And we need to use **scanf** to receive the inputs.

```
scanf("%d", &x);
```

As like **printf**, we have to give the format of inputs.

They are more or less the same as **printf**, but the second one and so on are variables instead of value, because we have to give them a place to live in our program.

Also, there is also one difference that before the variable name, an extra & is added.

This & is very important, but the reason cannot be taught until you know pointer.

For more about I/O, visit the following links:

http://www.cs.ntu.edu.au/sit/resources/cprogram/c_002.htm

Programming Examples:

[Ch1_01.cpp](#), [Ch1_02.cpp](#)

Exercises:

(Without using calculator)

#1 Evaluate the following expressions:

- | | |
|-----------------------------------|-----------------------------------|
| (a) $\text{floor}(2.5*3+0.2)$ | (b) $\log(10*10*10)$ |
| (c) $(2+\text{abs}(3-7))/2$ | (d) $\text{sqrt}(-4*-4)$ |
| (e) $\text{ceil}(\text{sqrt}(7))$ | (f) $\text{floor}(\log(12345))+1$ |
| (h) $10 \% 3 \% 2$ | (i) $100 \% 31$ |
| (j) $3 * 7 \% 2+3 \% 3$ | (k) $(1+2 * 4-5)\%4$ |

#2

Complete the following past paper problems.

Questions can be found from <http://www.hkoi.org/hkoi/ref.php>

HKOI2007hje Section A #17

HKOI2006hje Section A #8, #10, #16, #17, Section B #1 , #3

HKOI2005hje Section A #10, Section B #6

hje stands for Heat Event Junior English

End of Chapter