

HKOI 簡介:

HKOI 是 **香港電腦奧林匹克競賽** 的簡稱，這是一年一度的全港中學校際比賽。

比賽主要考驗參賽者的**編程技巧**、**解題能力**及**數學能力**。

比賽分為 初賽 及 決賽 兩個階段，初賽以筆試形式舉行，而決賽則採機上形式。

分為初級組及高級組，

初級組參賽條件：1990 年 8 月 2 日或以後出生的學生

高級組參賽條件：1987 年 8 月 2 日或以後出生的學生

初賽中，分為選擇題及填充題兩部份，參賽者必須選用 C++或 Pascal 其中一種語言作答填充部份，期間不可使用任何電子計算工具，一切計算必須透過筆算或心算進行，值得注意的是，主辦單位並不會提供任何算草紙，比賽共一小時三十分。

決賽時，除了大會指定的電腦外，其餘一切電子器材均不得使用，比賽進行時，參賽者使用的電腦將不能連結到網絡上，比賽一共三小時，總共有 5 題，第 0 題總分為 50 分，其餘 4 題總分為 100 分，滿分為 450 分，三小時內所編寫的程式總得分最高的 30 位將可獲得獎狀。

HKOI 的官方網頁：<http://www.hkoi.org>

本訓練班：

本訓練班會教授 **Pascal 編程語言**，這既是一套容易理解的語言亦是 香港中學會考電腦科 - 編程組別 需要學習的編程語言。

除此之外，6 堂內亦會教授 比賽所需 的基礎 數學技巧 及 解題技巧。

然後，暑假時亦會有進階訓練班。

What is a Program (程式)?

A program consists of **instructions** (指令) for computer to run, which usually has a **File extension** (副檔名) **exe**. We call it an **executable file** (執行檔).

Computers know the instructions inside such file, however, human beings can never understand the content through a text editor.

Therefore, we need other tools to create a program.

It is called **Programming Language** (編程語言).

編程語言是一種人類容易理解的文字，方便人類編寫一個程式。

儲存着編程語言的檔案，我們稱呼它做 **Source Code** (源始碼)。

它是有文法、結構等等，就像一篇作文有格式、文體等。

可是，電腦看不懂這種文字，它需要透過 **Compile** (編譯) 這個過程才會明白箇中含意，**編譯**這個過程就和我們使用翻譯機閱讀其他語言寫成的文章一樣。

下圖展示編譯的過程：



負責把源始碼轉換成執行檔的叫做 **Compiler** 編譯器。

編程語言也有很多種，例如 Pascal, C, C++, PHP, ASP, JAVA

當中，以 Pascal 最容易學習。

在訓練班使用的 Pascal 編譯器有兩個：

- 1) FreePascal 2.0
- 2) Turbo Pascal for Windows 1.5

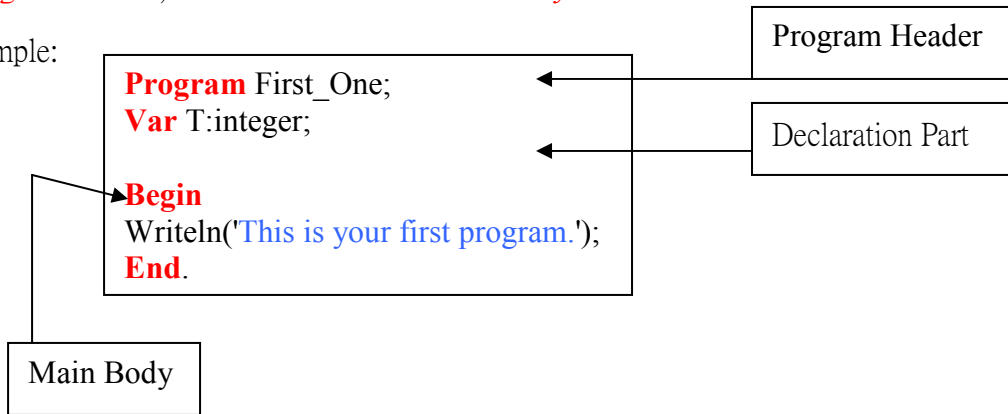
正式比賽時，大會採用的編譯器是 FreePascal 2.0。

Pascal's Code Style:

There are 3 parts in usual Pascal Code:

Program Header, Declaration Part and *Main Body*.

Sample:



```
Program First_One;
```

We call this a statement, every statement must end with a semi-colon.

Notes that some special key words are red in color.

These words are called *reserved word (保留字)*.

They have special meaning to us, which will be discussed later.

Reserved Word	Example	Descriptions
Program	Program First_One;	To give a program a name <i>First_One</i> .
Var	Var T : Integer ;	To declare a integer variable called <i>T</i> .
Being ... End.		Beginning and ending of the main program.

Pascal is a case-insensitive language, there is no different between Capital letters and Small letters. So `var t:integer;` and `Var T:Integer;` have the same meanings.

```
WRITELN ( expr1 , expr2 , ... , exprN ) ;
```

This Statement will output the value of `expr1`, `expr2` ... `exprN` to the screen and go to a new line, where **expr** stands for any expression.

```
READLN ( var1 , var2 , ... , varN ) ;
```

This Statement will ask user to enter at least `N` values, which are then stored into `var1`, `var2`, ... `varN` separately, where **var** stands for **Variable**.

**If there are more than `N` values, ignore those extra values and go to the next line for input.*

Variable (變數) / (變量):

It stores data in a computer's RAM (Random-access memory).

In fact, we need them to store changing information.

In Pascal, we use the following statements to declare/open variables.

```
Var name1,name2 : type1 ;
    name3,name4 : type2 ;
```

Example: BI2.pas

```
Program SecondProgram;
Var a:string;
Begin
Writeln('Please enter your name :');
Readln (a);
Writeln('Hi ',a,');
End.
```

Constant (常數) :

It represents unchanged data, for example, mathematical constants.

Pi is a default value for π in Pascal .

```
Const e = 2.71828;
      Root2 = 1.41421;
```

*Name must begin with **alphabetical letters** (a-z) or **underscore** _ and consists of a-z , 0-9, _ , it must also not the same as any **reserved word**.*

Identifier (識別符) is a technical term for **Name**.

For example, **a** is valid identifier, while **2ndBase** and **Money\$** are invalid.

Basic Operator:**Assignment (:=)**

It stores value to variable.

Example : BI3.pas

```

Program Third;
Var k:integer;
Begin
  K := 1;
  Writeln(K);
End.

```

Arithmetic Operator:

Operator	Descriptions
+	Addition or Concat
-	Subtraction
*	Multiplication
/	Real Division
div	Integer Division
mod	Modulus (Remainder)

Comparison Operator:

Operator	Descriptions
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Smaller than or equal to
<>	Not equal to

Example: BI4.pas

```

Program Forth;
Var a:integer;
Begin
  A:=2;
  Writeln('A equal to 2 is ', a=2);
End.

```

Logical Operators:

A and B represents **Boolean value**.

The **xor** pronounced “exclusive or” is equal to operator “ $\lt \gt$ ” for logical comparison.

Not

A	Not A
True	False
False	True

And (is A and B both true?)

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

Or (is at least one of A and B true?)

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

Xor (is only either A or B true?)

A	B	A xor B
True	True	False
True	False	True
False	True	True
False	False	False

Precedence Table (順序表):

()	0
Not, - (Unary)	1
And, *, /, div, mod	2
Or, Xor, +, -	3
>, <, <=, >=, <>, =	4

Always calculate an expression from left to right, when operators have equal precedence.

For example, $10 \text{ div } 2 + 1 - 2 * 3$.

- 1) Do $10 \text{ div } 2$ then $2 * 3$, it become $5 + 1 - 6$
- 2) Calculate $5 + 1$ first, and become $6 - 6$, finally it become 0.

Data Type:

Type	Example	Descriptions
Integer **	55	Integer in the range: $-2^{31} \sim 2^{31}-1$
Real ***	7.5838	Real Ranged: $2.9 \times 10^{-39} \sim 1.7 \times 10^{38}$
Boolean	TRUE	Only TRUE or FALSE
Char	'A'	Any one key on the keyboard
String **	'This is a string'	Combination of char, max length 255
Type	Total Sizes (Bytes)	Descriptions
Byte	1	Unsigned 8-bit integer
Word	2	Unsigned 16-bit integer
LongWord *	3	Unsigned 32-bit integer
Qword *	4	Unsigned 64-bit integer
Shortint	1	Signed 8-bit integer
Smallint *	2	Signed 16-bit integer
Longint	3	Signed 32-bit integer
Int64 *	4	Signed 64-bit integer
Cardinal ****	2, 4, 8	Either word, longword or Qword
ShortString *	256	String, max length 255
AnsiString *	4 + # of chars + 1	String, infinite long
WideString *	4 + # of chars x 2 + 2	Unicode String, infinite long
Single	4	Real Ranged: $1.5 \times 10^{-45} \sim 3.4 \times 10^{38}$
Double	8	Real Ranged: $5.0 \times 10^{-324} \sim 1.7 \times 10^{308}$
Extended	10	Real Ranged: $1.9 \times 10^{-4951} \sim 1.1 \times 10^{4932}$
Pointer	4	Store memory address of a variable
Pchar	4 + # of chars + 1	C-Style null-terminated string
PwideChar	4 + # of chars x 2 + 2	C-Style null-terminated Unicode String

* : This data type is only in FreePascal.

** : The description of this data type vary from FreePascal to TPW.

*** : This data type is platform-dependent.

**** : This data type has neither fixed size nor range, and only exists in FreePascal.

Unless specified explicitly, all strings are assumed to be ASCII-string.

Conditional Statements I:

We introduced a new statement, which allow a computer to do something conditionally.

e.g.

I ask the computer to output “Tuesday” if today is Tuesday,
otherwise if today is Monday output “Tomorrow is Tuesday”,
otherwise output “Today is neither Tuesday nor Monday”.

Syntax (語法):

```
if expr1 then statement1;
```

```
if expr1 then statement1 else statement2;
```

```
if expr1 then statement1 else if expr2 then statement2;
```

* *if ... then ... else* is considered as a whole statement.

So never put a *semi-colon (;)* before the word *else*.

This statement can be thought as

If *expr1* is true Do the then part, Otherwise Do the else part .

Example: B15.pas

```
Program Fifth;
var today:integer;
begin
write('Please enter Today's day of week :');
readln(today);
if today=2 then writeln('Tuesday.')
else if today=1 then writeln('Tomorrow is Tuesday.')
else writeln('Today is neither Monday nor Tuesday.');
```

You may notice that when you type today’s day of week, the cursor is placed at the right hand side of “:”.

It is because I use **write** instead of **writeln**.

The only difference between them is just a new line.

This is similar for **read** and **readln**.

Conditional Statements II:

Block of statements:

Sometimes there is need for enclosing several statements by **begin ... end;** to form a block of code.

For example,

I ask a computer to do the following things if today is Friday.

- 1) Output "Friday"
- 2) Output "Today is a Black Friday"

See Example BI6.pas

```
Program sixth;
Var today:integer;
Begin
write('Please enter Today's day of week :');
readln(today);
if today=5 then writeln('Friday. ');writeln('Today is Black Friday');
End.
```

Try Enter 5, this code seems perform what I want to, but when you enter 1,2 or any numbers other than 5, it still give the output 'Today is Black Friday'.

It's because only `writeln('Friday.')` is considered as part of the `if ... then`, `writeln('Today is Black Friday');` is always executed.

To do exactly what I want, use **begin ... end;** to enclose the two statements.

```
if today=5 then begin writeln('Friday. ');writeln('Today is Black Friday');end;
```

General Syntax:

```
If expr1 then
Begin
Statement1;
Statement2;
...
End
Else
Begin
OtherStatement1;
OtherStatement2;
...
End;
```

Conditional Statements III:

Nested-If:

```
If expr1 then statement1 else if expr2 then statement2 else .....
```

This is called **nested-if** statement, which means if then else if then else ... repeatedly.

Example BI7.pas

```
Program seventh;
Var I:integer;
Begin
  Readln(i);
  If i=1 then writeln('It's number one.')
  else if (i>=10) and (i<20) then writeln('It's tenth digit is 1.')
  else if (i>=100) and (i<=199) then writeln('It's hundredth digit is 1.')
  else if (i>=1000) and (i<=1999) then writeln('It's thousandth digit is 1.')
  else writeln('It's greater than 1999 or it isn't start with 1.');
```

But, you may find it annoying to type so many if then else.
Here is an alternative to it.

Case ... of

Example BI8.pas

```
Program seventh;
Var I:integer;
Begin
  Readln(i);
  Case I of
    1: writeln('It's number one.')
    10..19 : writeln('It's tenth digit is 1.')
    100..199 : writeln('It's hundredth digit is 1.')
    1000..1999: writeln('It's thousandth digit is 1.')
  else writeln('It's greater than 1999 or it isn't start with 1.');
```

Syntax:

```
Case expr1 of
  Value1: statement1;
  Range1:statement2;
  Value2,value3,value4:statement3;
  ...
End;
```

```
Case expr1 of
  Value1: statement1;
  Range1:statement2;
  Value2,value3,value4:statement3;
  ...
Else
  Statement4;
  Statement5;
  ...
End;
```

Theoretically there is nested case...of statement, but never try to do so in your program, this causes your program difficult to understand.

Exercise I:

Write programs to do the following tasks:

- 由用戶輸入三個整數，輸出最大的一個。
- 用戶輸入圓周，求出圓形的面積。
- 總共有 m 粒糖果，每個盒子可以盛載 n 粒糖果，由用戶輸入 m 及 n ，輸出最少數目的盒子以盛載所有糖果。

Exercise II:

Without using computers, complete the following questions:

English	Chinese	Question No.
1999hje.doc	1999hjc.doc	6-8,11-13,15,19,21,26,31,37-39
2000hje.doc	2000hjc.doc	1,17,18
2003hje.doc	2003hjc.doc	A26

Exercise III * :

Without using computers, complete the following questions:

English	Chinese	Question No.
2003hje.doc	2003hjc.doc	A17, A24, B3
2004hje.doc	2004hjc.doc	A1, A14, B3

* : Optional, sometimes very difficult and tricky, you may leave it blank.