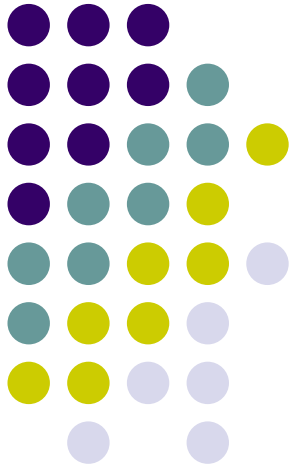
第二章
C 語言基本概述

C 語言的基本語法

關鍵字 vs. 識別字

各種程式錯誤

提高程式的可讀性



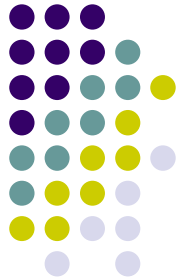
簡單的 C 程式

- 下面的程式碼可印出兩行字串：

```
01  /* prog2_1, 簡單的C語言 */
02  #include <stdio.h>          /* 把 stdio.h 這個檔案含括進來 */
03  #include <stdlib.h>         /* 把 stdlib.h 這個檔案含括進來 */
04  int main(void)              /* 主函數 main() 從這兒開始 */
05  {
06     int num;                 /* 宣告整數變數 num*/
07     num=2;                   /* 把 num 的值設為 2 */
08     printf("I have %d cats.\n",num);          /* 呼叫 printf()函數 */
09     printf("You have %d cats, too.\n",num); /* 呼叫 printf()函數 */
10     system("pause");   /* 呼叫 os 裡的 pause 指令，用來暫停程式的執行 */
11     return 0;
12  }
```

```
/* prog2_1 OUTPUT--

I have 2 cats.
You have 2 cats, too.
--------------------*/
```



含括指令與標頭檔 (1/4)

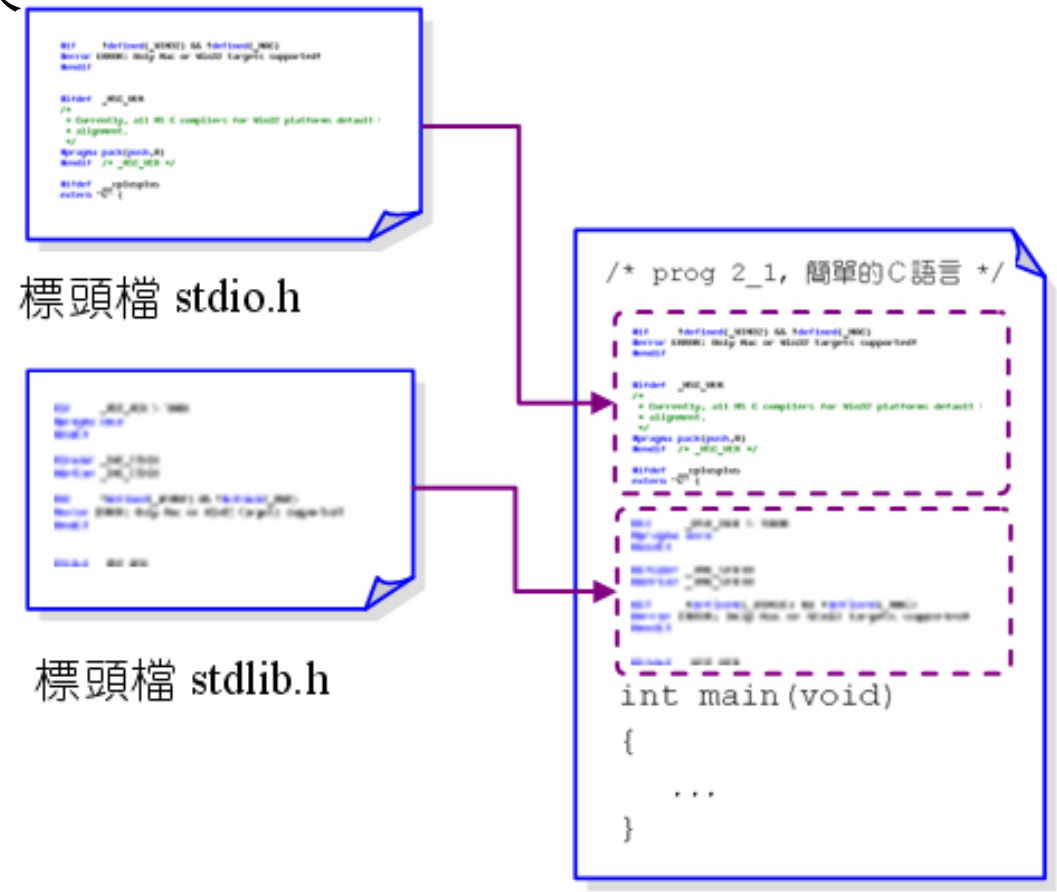
- #include 是前置處理器的指令
  - #include 稱爲含括指令
  - 語法爲 #include <標頭檔>
  - 前置處理器以標頭檔（header file）的內容取代 #include < >
  - 因爲是在編譯前執行，所以稱爲 "前置" 處理器

含括指令與標頭檔 (2/4)

- 含括動作前後的比較：

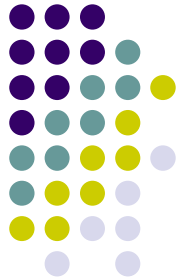
```
/* prog 2_1, 簡單的C語言 */
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   ...
}
```

含括標頭檔前



含括標頭檔後

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



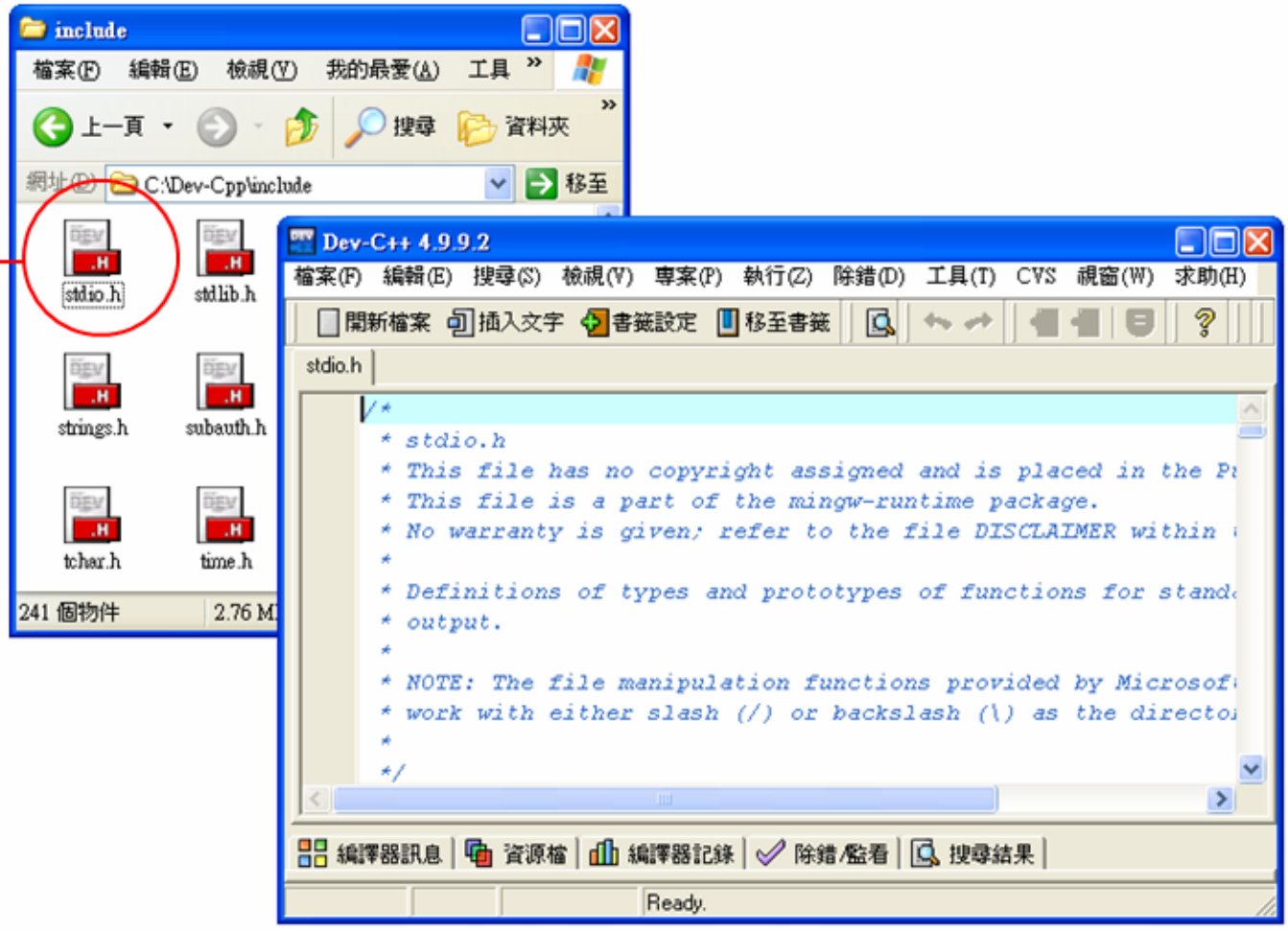
含括指令與標頭檔 (3/4)

- 不含括 stdio.h 或 stdlib.h 標頭檔也可以編譯？
  - 標頭檔內是工具函式的宣告, 例如 printf() 在 stdio.h 中, system() 在 stdlib.h 中, 程式裡有使用到的工具函式才需要含括對應的標頭檔案
  - 某些編譯器會將常用的標頭檔自動含括
  - 有些編譯器會出現警告訊息，並自動含括一些標頭檔
- 早期的 C 編譯器沒有適當宣告時有一些預設的法則; ANSI C 的編譯器如果沒有含括所使用函數的標頭檔，所用到的工具函數沒有適當的宣告時即無法編譯

含括指令與標頭檔 (4/4)

- 標頭檔的內容：

stdio.h 存放在 include 資料夾內



函數 (function) main()

- main() 函數是你的程式執行的起點
- 每個 C 程式必須有一個 main() 函數，而且只能有一個

函式

傳回型態為整數　　main()函數不需傳入引數

```
int main( void )
{
    程式敘述;
    return 0;
}
```

return 0; → main()函數執行完畢，傳回整數 0

程式區塊及本體

- 程式區塊與本體的範圍：

```
/* prog2_2 OUTPUT--
變數 i 的值小於 5
--------------------*/
```

```
01  /* prog2_2, main()函數的本體與程式區塊 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06    int i=2;  /* 宣告整數 i，並設值為 2 */
07    if(i<5)   /* if 區塊由此開始 */
08    {
09       printf("變數 i 的值小於 5");
10       printf("\n");    /* 換行 */
11    }                   /* if 區塊結束 */
12    system("pause");
13    return 0;
14  }
```

main() 函數的本體

if 敘述的程式區塊

變數的使用

- 宣告方式：　變數是CPU執行演算法過程中存放資料的地方
  - `int num;` `/* 宣告名稱爲 num 的整數變數 */`
  - `int a,b,c;` `/* 宣告 a,b 與 c 爲整數變數 */` 同一敘述宣告三個變數
  - `float sum=0.0;` `/* 宣告浮點數變數 sum，並設值爲 0.0 */`

    變數的初始化
- 變數裡存放的資料型態：
  - char　字元，如 'A'、'2'與 '&'等
  - int　整數
  - long　長整數　} 如12、-27 等
  - short　短整數
  - float　單精度浮點數
  - double　倍精度浮點數　} 如12.762、-37.483 等

變數的使用

Integer Type	Range in Typical Microprocessor Implementation
short	-32768 ~ 32767
unsigned short	0 ~ 65535
[1]int	-2147483648 ~ 2147483647
[1]unsigned int	0 ~ 65535
long	-2147483648 ~ 2147483647
unsigned long	0 ~ 4294967295

Floating-Point Type	[2]Approximate Range	Significant Digits
float	[3]10^{-37} ~10^{38}	6
double	10^{-307} ~10^{308}	15
[4]long double	10^{-4931} ~10^{4932}	19

1: machine, operating system, and compiler dependent

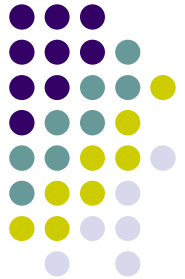
2: not all numbers in the range can be represented precisely

3: The mass of one electron is approximately 10^{-27} grams.
Diameter of the Milky Way galaxy in kilometers is approximately 10^{23} kms.

4: *long double* is the same as *double* for VC6 and VC2005

ASCII 字元內碼表

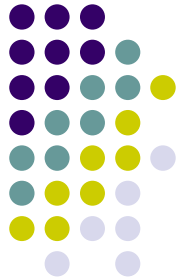
	0	1	2	3	4	5	6	7	8	9
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT
1	LF	VT	FF	CR	SO	SI	DLE	DCL	DC2	DC3
2	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS
3	RS	US	SP	!	"	#	$	%	&	'
4	(	)	*	+	,	-	.	/	0	1
5	2	3	4	5	6	7	8	9	:	;
6	<	=	>	?	@	A	B	C	D	E
7	F	G	H	I	J	K	L	M	N	O
8	P	Q	R	S	T	U	V	W	X	Y
9	Z	[	\	]	^	_	`	a	b	c
10	d	e	f	g	h	i	j	k	l	m
11	n	o	p	q	r	s	t	u	v	w
12	x	y	z	{	|	}	~	DEL		



變數的命名規則

- 變數名稱可以是英文字母、數字或底線
  - 名稱中不能有空白字元
  - 第一個字元不能是數字
  - 不能使用到關鍵字

```
intel_4x      /* 正確 */
_AMD          /* 正確，變數的第一個字母可以是底線 */
2dos          /* 錯誤，變數的第一個字母不能是數字 */
my dogs       /* 錯誤，變數不能有空格 */
goto          /* 錯誤，變數不能是C語言的關鍵字 */
```



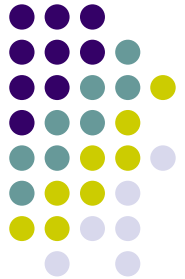
變數的設值方式

- 宣告的時候設值 (初始化)

```
int num = 2;    /* 宣告變數，並直接設值 */
```

- 宣告後再設值

```
int num1,num2;      /* 宣告變數 */
char ch;
num1 = 2;      /* 將整數變數num1的值設爲 2 */
num2 = 30;     /* 將整數變數num2的值設爲 30 */
ch   = 'm';    /* 將字元變數ch的值設爲 'm' */
```



Strong type Language

- Weak-type Language: 變數使用前不需要宣告, 同一個變數裡可以放不同型態的資料
- Strong-type Language: 變數在使用之前一定要宣告, 並且只能存放指定型態的資料, 限制嚴格的好處如下：
  - 避免變數名稱打錯 （如數字0與英文字母O）
  - 增加程式的可讀性
  - 便於程式碼的維護
  - 除錯容易

格式化的輸出函數 printf()

- 利用 printf() 函數在螢幕上印出字串：

```
01  /* prog2_3, printf()函數的練習 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int num=2;     /* 定義變數 num，並設值為 2 */
07     printf("I have %d cats.\n",num);   /* 呼叫 printf()函數 */
08     system("pause");
09     return 0;
10  }
```

```
/* prog2_3 OUTPUT---

I have 2 cats.
----------------------*/
```

識別字 (identifier)

- 識別字是用來命名變數或函數的文字

```
01  /* prog2_3, printf()函數的練習 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int num=2;     /* 定義變數 num，並設值為 2 */
07     printf("I have %d cats.\n",num);   /* 呼叫 printf()函數 *
08     system("pause");
09     return 0;
10  }
```

識別字

關鍵字（keyword）(1/2)

- 關鍵字是 C 語法的基本元素

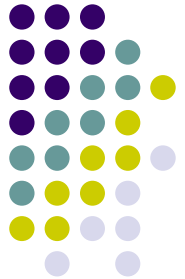
```
01  /* prog2_3, printf()函數的練習 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06    int num=2;      /* 定義變數 num，並設值為 2 */
07    printf("I have %d cats.\n",num);   /* 呼叫 printf()函數 */
08    system("pause");
09    return 0;
10  }
```

關鍵字 (keyword) 或稱爲
保留字 (reserved word)

關鍵字（keyword） (2/2)

- 下表爲 C 語言的關鍵字

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



程式錯誤的分類

- 語法錯誤（syntax error）
  - 程式含有不合語法的敘述，它無法被編譯程式翻譯
- 語意錯誤（semantic error）
  - 語意錯誤(又稱邏輯錯誤)，就是程式的執行結果與寫程式者的預期不同

語法錯誤

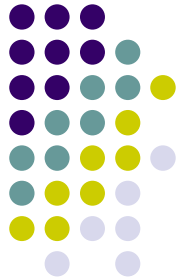
- 下面是有語法錯誤的程式：

```
01  /* prog2_4, 有錯誤的程式  */
02  #include <stdio.h>
03  #include <stdlib.h>
04
05  int main(void)
06  {
07     int num;        /* 宣告整數 num  /*
08     num=2;          /* 將 num 設值為 2 /*
09     printf("I have %d dogs. \n",num);
10     printf("You have %d dogs, too. \n,num);
11     system("pause")
12     return 0;
13  )
```

```
/* prog2_4 OUTPUT 除錯後的結果 --

I have 2 dogs.
You have 2 dogs, too.

--------------------------------*/
```



語意錯誤

- 下面是語意錯誤的程式：

語意錯誤通常是你以為電腦會做的, 和電腦實際做的之間有落差 ….誤會一場

```
/* prog2_5a, 語意錯誤的程式 */
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    int num = '2';    /* 宣告整數變數 num, 並設值為 '2' */
    printf("I have %d dogs.\n", num);
    system("pause");
    return 0;
}
                        /* prog2_5a OUTPUT ---
                        I have 50 dogs.
                        ---------------------------------*/
```

提高程式的可讀性 (1/4)

- 列印程式碼時請用固定字距

```
/* 使用固定字距的程式碼，字型為 Courier New  */
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   printf("We all love C. \n");
   system("pause");
   return 0;
}
```

提高程式的可讀性 (2/4)

- 列印程式碼時使用非固定字距，且斜體字的程式碼，較難閱讀

```
/* 使用非固定字距，且斜體字的程式碼，字型為 Times New Roman*/
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
    printf("We all love C. \n");;
    system("pause");
    return 0;
}
```

提高程式的可讀性 (3/4)

- 程式碼縮排與對齊，分隔不同的細節層次

```
/* prog2_6, prog2_7 OUTPUT------
cats are running, dogs are chasing.
cats are running, dogs are chasing.
-------------------------------*/
```

```
01  /* prog2 6,有縮排的程式碼  */
02  #include <stdio.h>
03  #include <stdlib.h>
04
05  int main(void)
06  {
07     int i;
08     for(i=1;i<=2;i++)
09     {
10        printf("Cats are running, ");
11        printf("dogs are chasing.\n");
12     }
13     system("pause");
14     return 0;
15  }
```

```
01  /* prog2 7, 沒有縮排的程式碼  */
02  #include <stdio.h>
03  #include <stdlib.h>
05  int main(void)
06  {
07  int i;
08  for(i=1;i<=2;i++)
09  {
10  printf("Cats are running, ");
11  printf("dogs are chasing.\n");
12  }
13  system("pause");
14  return 0;
15  }
```

提高程式的可讀性 (4/4)

- 註解有助於程式的閱讀與偵錯

```
/* prog2_7, examples */
/* created by Wien Hong */
```

以註解符號對每一行做註解

```
/*
   This paragraph demonstrates the
   capability of comments used by C
   November 06 2003
*/
```

於「/*」和「*/」符號之間的文字均是註解

C 是 Free Format 的語言

- #include<stdio.h>
  #include<stdlib.h>
  int main(void){
  printf("HelloWorld!\n");
  system("pause");
  return 0;
  }
- #include<stdio.h>
  #include<stdlib.h>
  int main(void){printf("HelloWorld!\n");system("pause");return 0;}
- int i;main(){for(;i["]<i;++i){--i;}"];read('-'-'-',i+++“Hell\
  oWorld!\n",'/'/'/'));}read(j,i,p){write(j/p+p,i---j,i/i);}
- *The International Obfuscated C Code Contest*
  http://www.ioccc.org/years.html

Winner of the international C obfuscation contest in 2001

```
#include <unistd.h> #include <curses.h> #include <sys/socket.h> #include <netinet/in.h>
#include <netdb.h> #include <sys/time.h> #define o0(M,W) mvprintw(W,M?M-1:M,"%s%s
",M?" ":"",_) #define O0(M,W) M##M=(M+=W##M)-W##M #define l1(M,W)
M.tv_##W##sec #define L1(m,M,l,L,o,O) for(L=l;L--;)((char*)(m))[o]=((char*)(M))[O]
#define I1 lL,(struct sockaddr*)&il #define i1 COLS #define j LINES #define L_ ((j%2)?j:j-1)
fd_set I;struct socka\ ddr_in il;struct host\ ent*LI; struct timeval IL,l;char L[9],_[1<<9] ;void
___(int __ ){_[__--]=+0;if( ++__)___(--__);_ [__]='=';}double o,oo=+0,Oo=+0.2; long
O,OO=0,oO=1 ,ii,iI,Ii,Ll,lL, II=sizeof(il),Il ,ll,LL=0,i=0,li, lI;int main(int\ iL,char *Li[]){\
initscr();cbreak ();noecho();nonl ();___(lI=i1/4); _[0]='[';_[lI-1] =']';L1(&il,&_,\
II,O,+O,+lI);il. sin_port=htons(( unsigned long)(\ PORT&0xffff));lL =l_;if(iL=!--iL) {il.
sin_addr .\ s_addr=0;bind(I1 ,II);listen(lL,5 );lL=accept(I1,& II);}else{oO-=2;
LI=gethostbyname (Li[1]);L1(&(il. sin_addr),(*LI). h_addr_list[0],\ LI->h_length,iI,
iI,iI);(*(&il)). sin_family=(&(*\ LI))->h_addrtype ;connect(I1,II); }ii=Ii=(o=i1*0.5 )-
lI/2;iI=L_-1;O =li=L_*0.5;while (_){mvaddch(+OO, oo,' ');o0(ii,iI );o0(Ii,Il-=Il);
mvprintw(li-1,Il ,"%d\n\n%d",i,LL );mvhline(li,+0, '-
',i1);mvaddch( O,o,'*');move(li ,Il);refresh();\ timeout(+SPEED);
gettimeofday(&IL ,+0);Ll=getch(); timeout(0);while (getch()!=ERR);\ if(Ll=='q'&&iL)\
write(lL,_+1,1); if(ii>(ll=0)&&Ll ==','){write(lL, _,-(--Il));}else if(Ll=='.'&&ii+\
lI<i1){write(lL, _+lI,++Il);}else if(iL||!Il)write (lL,_+lI-1,4-3); gettimeofday(&l,
0);II=((II=l1(IL ,)+(l1(l,u)-=l1( IL,u))-l1(l,)+(\ l1(l,)-=l1(IL,)) )<0)?1+II-l1(l,) +1e6+(--l1(l,)):
II;usleep((II+=\ l1(l,)*1e6-SPEED *1e3)<0?-II:+0); if(Ll=='q'&&!iL)
break;FD_ZERO(&I );FD_SET(lL,&I); memset(&*&IL,ll, sizeof(l));if((\ Ll=select(lL+1,&
I,0,0,&IL)));{if (read(lL,&L,ll+1 )){if(!*L){ll++; }else if(*L==ll[ _]){ll--; }else\
if(*(&(*L))==1[_ ]){break;}}else{ break;}}O0(o,O); O0(O,o);if(o<0){ o*=-1;Oo*=-1;}if
(o>i1){o=i1+i1-o ;Oo*=-1;}if(o>=( Ii+=ll)&&O<1&&oO <0&&o<Ii+lI){O=2 ;oO=~--
oO;Oo+=ll *4e-1;}if(O<0){O =iI;LL++;}if(o>= (ii+=Il)&&O>iI-1
&&oO>0&&o<ii+lI){O=iI- 2;oO=~--oO;Oo+=Il*4e-1 ;}if(+O>+iI){O-
=O;i++; }}endwin();return(0);}
```

Network-based Pong game