1

期中考事宜

- 日期： 4/20 (Fri.) 上課時段
- 地點： TR-413-2
- 範圍：考至 (含) 一維陣列
- 題型：選擇、填空、程式
- 考試型式：open book、open note (不可以使用電子裝置如手機、電腦等等)

Lecture 9

課程回顧

- 基本程式架構
- 變數
- 運算
- 分支／選擇
- 迴圈
- 一維陣列

3

1-1.cpp

```
#include <iostream>
using namespace std;

int main() {
    // 輸出 "Hello World" 至螢幕上
    cout << "Hello World";

    /* 函式執行完畢，回傳 0 */
    return 0;
}
```

4

補充 – 另一種主程式**開頭**的寫法

```
#include <iostream>
using namespace std;

void main() {
    // 輸出 "Hello World" 至螢幕上
    cout << "Hello World";

}
```

此寫法在在 Microsoft Visual Studio 上可以接受，但是此寫法不符合 C/C++ 的標準，如果在其它的平台上可能不被接受。

5

C/C++ 程式註解

單行註解

範例：`// output "Hello World" to console`

說明：`//` 之後所有的字都被認為是註解的一部份直到換行為止

區塊註解

範例：`/* return value 0 */`

說明：

- `/*` 表示註解區塊的開始
- `*/` 表示註解區塊的結束
- 可用來註解多行

```
/**********
  這是一個註解
  這裡還是註解
  這行還是註解
  **********/
```

```
/* 這也是一行註解 */
```

```
// 這也是一行註解
```

```
// 這也是一行註解 //
```

6

變數
Variable

電腦具有處理大量資料的能力, 但是資料必需先存在記憶體中, 而變數就是用來幫助我們存取記憶體的工具...

7

變數
Variable

- 一個變數具有：
  - **名稱** (identifier/識別項): 用來區別不同的資料存放位置
  - **型態** (type): 代表該變數所儲存資料的意義
- 所有變數在使用**前**必需先被宣告 (declaration)
  - int a = 3; ← 宣告一個變數, 名稱為 a, 並賦予初始值為 3
  - int b, c; ← 宣告兩個變數, 名稱為 b 和 c。
  - 在 C 語言時代, 所有變數都必需在所有的敍述**之前**宣告完成。在 C++，變數宣告只要在使用前宣告即可，沒有一定要放在程式的開頭。

8

變數名稱、識別項
name, identifier

- 用來區別不同的變數，且大小寫視為不同 (case sensitive)
- 文字、數字、與底線符號的組合
- 第一個字元必需為文字或是底線
  - 正確: myName, _myName2, my_Name5,
  - 錯誤: 4Name, 4_name, my-Name
- 變數名稱不可為保留字 (reserved words), 如C/C++語言中的指令 while, for, if, int, ...

9

變數型態
variable type

運算過程中運算結果超出一型別可儲存的數值範圍的情況被稱為 overflow 或是 underflow.

意義	變數型態	數值範圍	準確度 (Accuracy)	可加的修飾字	
布林	bool	true / false (1 / 0)	精確	const	
字元	char	-128 ~ +127	精確		Signed/ unsigned
整數	short	-32768 ~ 32767	精確		
	int	$-2^{31} \sim 2^{31}-1$	精確		
	long	$-2^{31} \sim 2^{31}-1$	精確		
浮點數	float	$-3.4\times10^{38} \sim +3.4\times10^{38}$	6-7 位有效數字		
	double	$-1.7\times10^{308} \sim +1.7\times10^{308}$	14 – 15 位有效數字		
	long double	$-1.7\times10^{308} \sim +1.7\times10^{308}$	14 – 15 位有效數字		
字串	string	不適用	精確		

* 單位為 bytes。記憶體量與所使用之作業系統、硬體、以及編譯器等皆有關係，本表則依據win32作業系統之 Visual C++ 9.0 。

10

1-2.cpp

```
#include<iostream>
using namespace std;

int main() {
  int a = 3;
  int b, c;

  b = 4;
  c = a + b;
  cout << "a+b=" << c;

  return 0;
}
```

宣告一個變數 a, 並給予初始值 3
宣告兩個變數 b 與 c

令 b 變數的值為 4
令 c 變數的值為 a+b 的計算結果
輸出 字串 "a+b=" 以及變數 c 的值

a+b=7

11

運算

12

運算子

- 指定運算子: =
- 算術運算子: +、- 、 * 、 / 、 %、++、--
- 關係運算子: ==、!= 、> 、 < 、 >= 、 <=
- 布林運算子: && 、 || 、!
- 複合指定運算子: +=、-= 、 *= 、 /= 、 %=

13

運算子優先順序 (level) 與運算方向 (grouping)

Level	Operator	Description	Grouping
2	() ++ --	postfix	Left-to-right
3	++ -- !	unary (prefix)	Right-to-left
4	(type)	type casting	Right-to-left
6	* / %	multiplicative	Left-to-right
7	+ -	additive	Left-to-right
8	<< >>	shift	Left-to-right
9	< > <= >=	relational	Left-to-right
10	== !=	equality	Left-to-right
14	&&	logical AND	Left-to-right
15	||	logical OR	Left-to-right
17	= *= /= %= += -=	assignment	Right-to-left

http://www.cplusplus.com/doc/tutorial/operators/

14

前置 (prefix) 與後置 (postfix)

- 前置
  - ++a、--b、++c、...
  - 在這行敍述執行之**前**完成該運算

C=3;
D=++C;

C=3;
C=C+1;
D=C;

- 後置
  - a--、b++、...
  - 在這行敍述執行之**後**再進行該運算

C=3;
D=C++;

C=3;
D=C;
C=C+1;

15

分支 / 選擇

16

如果我們今天希望使用者輸入的是
200 – 300 的數字，要怎麼改?

4-4.cpp

如果我們今天希望使用者輸入的是
100 – 200 或是 1000 – 2000 的數字，要怎麼改?

記得其中 else 的部份 (分支) 是可以省略的

```
#include <iostream>
using namespace std;
int main() {
    int data;

    cout << "請輸入一介於 1 到 100 的數字: ";
    cin >> data;
    if(data>=1 && data <= 100) {
        cout << "謝謝您的合作 \n";
    }
    else {
        cout << "你在找我麻煩哦 \n";
    }

  return 0;
}
```

17

4-5.cpp

多重分支、又稱為多途選擇

注意到如果我們把所有的 else 都去掉，程式還是正確的，但結果不是我們想要的。 - logical error

```
#include <iostream>
using namespace std;

int main() {
  int score;

  cout << "請輸入成績: ";
  cin >> score;

  if(score >= 90) {
    cout << "A";
  }
  else if(score >= 80) {
    cout << "B";
  }
  else if(score >= 70) {
    cout << "C";
  }
  else if(score >= 60) {
    cout << "D";
  }
  else {
    cout << "F";
  }

  return 0;
}
```

18

4-6.cpp – 巢狀式分支

```
if(number >= 1 && number <= 10) {
  if(number%2 == 0) {
    cout << "你輸入的數字" << number << "是偶數";
  }
  else {
    cout << "你輸入的數字" << number << "是奇數";
  }
}
else {
  cout << "你輸入的數字" << number << "超出範圍";
}

  return 0;
}
```

19

巢狀式的判斷，有助於避免條件的重複判斷、並減低判斷條件的難度。

4-6.cpp 改

```
if(number>=1 && number<=10 && number%2 == 0) {
    cout << "你輸入的數字" << number << "是偶數";
}
else if(number>=1 && number<=10 && number%2==1) {
    cout << "你輸入的數字" << number << "是奇數";
}
else {
    cout << "你輸入的數字" << number << "超出範圍";
}

 return 0;
}
```

20

有時多重分支順序重新調整，也有助於避免重複的判斷、並讓程式易讀。

4-6.cpp 再改

```
if(number<1 || number>10) {
    cout << "你輸入的數字" << number << "超出範圍";
}
else if(number%2==1) {
    cout << "你輸入的數字" << number << "是奇數";
}
else {
    cout << "你輸入的數字" << number << "是偶數";
}

 return 0;
}
```

21

其中， default: 的部份可以不寫

Switch 的依據必需是廣義的**整數**：int, char, long, short, bool, ...

4-8.cpp

輸入年齡可以辨認適合看什麼級別的電影

```
#include <iostream>
using namespace std;

int main() {
  int age;

  cout << "請輸入年齡: ";
  cin >> age;

  cout << "您可觀賞";
```

```
  switch(age/6) {
    case 0:
      cout << "普通級";
      break;
    case 1:
      cout << "保護級";
      break;
    case 2:
      cout << "輔導級";
      break;
    default:
      cout << "限制級";
  }

  return 0;
}
```

22

4-9a.cpp

```
switch(grade) {
   case 'A': case 'a':
     cout << "90 - 100";
     break;
   case 'B': case 'b':
     cout << "80 - 89";
     break;
   case 'C': case 'c':
     cout << "70 - 79";
     break;
   case 'D': case 'd':
     cout << "60 - 69";
     break;
   default:
     cout << "0 - 59";
 }
```

在switch-case 中，cases (e.g. case 'A') 的作用為標籤，供 switch 參考而跳躍。

在 case 裡執行時會一直執行到 break; 或是到 switch (...) { ... } 的最後為止

此現象稱為 fall through

左邊即為 fall through 的範例

23

迴圈

24

for 迴圈

```
for (i=0; i<3; i++) {
        cout << "a";
}

cout << "b";
```

1 2 3 4 5

1 迴圈被執行到時初始化的動作

2 迴圈要執行或要被繼續執行的條件

3 迴圈完成一次後要做的動作

4 迴圈內欲重複執行的敘述

5 迴圈結束後要執行的敘述

1 令 i = 0

2 0 < 3 → *true*

4 cout << "a";

3 i++;

2 1 < 3 → *true*

4 cout << "a";

3 i++;

2 2 < 3 → *true*

4 cout << "a";

3 i++;

2 3 < 3 → ***false***

5 cout << "b";

25

5-7.cpp & 5-8.cpp

```
#include <iostream>
using namespace std;

int main() {
  int i;

  for(i=0;i<15;i++) {
    if(i>=5 && i<10) break;
    cout << "i=" << i << endl;
  }
  cout << "i-迴圈結束" << endl;

  return 0;
}
```

```
#include <iostream>
using namespace std;

int main() {
  int i;

  for(i=0;i<15;i++) {
    if(i>=5 && i<10) continue;
    cout << "i=" << i << endl;
  }
  cout << "i-迴圈結束" << endl;

  return 0;
}
```

26

5-7 & 5-8 執行結果

```
i=0
i=1
i=2
i=3
i=4
i-迴圈結束
```

```
for(i=0;i<15;i++) {
  if(i>=5 && i<10) break;
  cout << "i=" << i << endl;
}
cout << "i-迴圈結束" << endl;
```

```
i=0
i=1
i=2
i=3
i=4
i=10
i=11
i=12
i=13
i=14
i-迴圈結束
```

27

while 迴圈

```
int a = 0;
1
while( a < 3) {
2       cout << “a”;
        a++;
}
3 cout << “b”;
```

1 判斷迴圈內容是否執行或繼續執行
2 迴圈內欲重複執行的敘述
3 迴圈結束後要執行的敘述

1	0 < 3 → *true*
2	cout << “a”;
2	a++;
1	1 < 3 → *true*
2	cout << “a”;
2	a++;
1	2 < 3 → *true*
2	cout << “a”;
2	a++;
1	3 < 3 → ***false***
3	cout << “b”;

28

6-3.cpp

```
#include <iostream>
using namespace std;
int main() {
  int ans;
  bool isOdd = true;

  while(isOdd) {
      cout << "請輸入一個偶數: ";
      cin >> ans;

      isOdd = (ans % 2 == 1);
  }
  cout << "你輸入了一個偶數: " << ans << endl;

  return 0;
}
```

```
請輸入一個偶數: 1
請輸入一個偶數: 3
請輸入一個偶數: 5
請輸入一個偶數: 7
請輸入一個偶數: 6
你輸入了一個偶數: 6
```

29

do/while 迴圈

```
int a = 0;

do {
1       cout << “a”;
        a++;
2 } while(a<3);
3 cout << “b”;
```

1 迴圈內欲重複執行的敘述
2 迴圈要執行或要被繼續執行的條件
3 迴圈結束後要執行的敘述

1	cout << “a”;
1	a++;
2	1 < 3 → *true*
1	cout << “a”;
1	a++;
2	2 < 3 → *true*
1	cout << “a”;
1	a++;
2	3 < 3 → ***false***
3	cout << “b”;

30

6-6.cpp

```
#include <iostream>
using namespace std;
int main() {
  int ans;

  do {
      cout << "請輸入一個偶數: ";
      cin >> ans;
  } while(ans%2 != 0);

  cout << "你輸入了一個偶數" << endl;

  return 0;
}
```

```
請輸入一個偶數: 1
請輸入一個偶數: 3
請輸入一個偶數: 5
請輸入一個偶數: 7
請輸入一個偶數: 6
你輸入了一個偶數
```

31

6-8.cpp

```
#include <iostream>
using namespace std;

int main() {
  int j;

  for(int i=10; i<=20; i=i+5) {
    cout << "* " << i << endl;
    for(j=2; j<=4; j++) {
      cout << "** " << i << ", " << j << endl;
    }
    cout << "+ " << i << ", " << j << endl;
  }
  return 0;
}
```

```
* 10
** 10, 2
** 10, 3
** 10, 4
+ 10, 5
* 15
** 15, 2
** 15, 3
** 15, 4
+ 15, 5
* 20
** 20, 2
** 20, 3
** 20, 4
+ 20, 5
```

32

6-8.cpp

```
int j;
    1a   1b   1c
  for(int i=10; i<=20; i=i+5) {
2   cout << "* " << i << endl;
      3a   3b   3c
    for(j=2; j<=4; j++) {
4     cout<<"** "<<i<<", "<<j<<endl;
    }
5   cout<<"+ "<<i<<", "<<j<<endl;
  }
6
```

1a	1b		2
	3a	3b	4
	3c	3b	4
	3c	3b	4
	3c	3b	5
1c	1b		2
	3a	3b	4
	3c	3b	4
	3c	3b	4
	3c	3b	5
1c	1b		2
	3a	3b	4
	3c	3b	4
	3c	3b	4
	3c	3b	5
1c	1b		6

```
* 10
** 10, 2
** 10, 3
** 10, 4
+ 10, 5
* 15
** 15, 2
** 15, 3
** 15, 4
+ 15, 5
* 20
** 20, 2
** 20, 3
** 20, 4
+ 20, 5
```

33

6-9.cpp

```
#include <iostream>
using namespace std;

int main() {
  int i, j;

  for(i=0; i<5; i++) {
    for(j=0; j<4; j++) {
      cout << "*";
    }
    cout << endl;
  }

  return 0;
}
```

```
****
****
****
****
****
```

34

6-10.cpp

```
#include <iostream>
using namespace std;

int main() {
  int i, j;

  for(i=0; i<10; i++) {
    for(j=0; j<i; j++) {
      cout << "*";
    }
    cout << endl;
  }

  return 0;
}
```

```
*
**
***
****
*****
******
*******
********
*********
```

35

陣列

36

陣列 (Array) 的使用

- int A[10];
  - 宣告一整數陣列, 名稱為 A，可存放 10 個整數型別的資料

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]

- 使用
  - A[0] = 3;→ 將 3 存入 A 陣列中的第 0 號元素
  - A[3]++; → 將 A 陣列中的 3 號元素增加 1
  - A[2]=A[4]+a;
  - 中括弧 [] 中的數字 (0, 3, 2, 4, ...) 用來選擇使用陣列中的那一個元素，被稱為索引 (index) 或下標 (subscript)
  - 常和 **for** 迴圈配合
  - 假設一陣列有 n 個元素，則可以**正確使用**的索引值為 0, 1, 2, ... n-1

37

7-2.cpp

```
int a[10];

for(int i=0;i<10;i++) {
      cout << "請輸入 " << i << " 號元素: ";
      cin >> a[i];
}

for(int i=9;i>=0;i--) {
      cout << a[i] << ", ";
}
```

```
請輸入 0 號元素: 1
請輸入 1 號元素: 2
請輸入 2 號元素: 3
請輸入 3 號元素: 4
請輸入 4 號元素: 5
請輸入 5 號元素: 6
請輸入 6 號元素: 7
請輸入 7 號元素: 8
請輸入 8 號元素: 9
請輸入 9 號元素: 10
10, 9, 8, 7, 6, 5, 4, 3, 2, 1,
```

38

陣列的初始化 (initialization)

- 變數在宣告/定義時，可給初始值
  - `int a=5, b=3;`
  - `double c=20;`
- 陣列亦可在宣告時，給定陣列中元素的**初始值**。
  - `int a[3] = {1, 2, 3};`
  - `double b[5] = {1, 2, 3};`
  - `float c[] = {1, 2, 3, 4};`
  - `double d[10] = {0};`
  - `char f[5] = {'H', 'E', 'L', 'L', 'O'}`

39

7-4.cpp

```
#include <iostream>
using namespace std;
int main() {
  double a[20] = {1, 1};

  for(int i=2;i<20;i++) {
      a[i] = a[i-1] + a[i-2];
  }

  for(int i=0;i<20;i++) {
      cout << a[i] << " ";
  }
  return 0;
}
```

```
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765
```

40

7-8.cpp

```
#include <iostream>
using namespace std;
int main() {
  char a[]= "Hello World";
  int i=0;
  while(a[i]!=0) {
      cout << i << ":" << a[i] << endl;
      i++;
  }
  a[4]=0; // a[4]='\0';
  cout << a << endl;
  return 0;
}
```

```
0:H
1:e
2:l
3:l
4:o
5:
6:W
7:o
8:r
9:l
10:d
Hell
```

41

作業

42

1-1. 標準體重

請撰寫一程式，首先利用分別不同的變數儲存自己的姓名、學號、性別、身高、體重。之後將前述所有資料印出，並印出根據以下公式計算出來之標準體重以及 BMI 值。

男生的標準體重：(身高-80) x 0.7
女生的標準體重：(身高-70) x 0.6
BMI 值的計算方法：$\text{BMI} = \frac{\text{體重(公斤)}}{\text{身高}^2\text{(公尺}^2\text{)}}$

1-2. 推土機

```
car.forward();          // 前進
car.backward();         // 後退
car.turnLeft();         // 左轉
car.turnRight();        // 右轉

car.slabDown();         // 放下置土板
car.moveMoundUP();      // 搬起土堆
car.putMoundDown();     // 放下土堆
```

2-1

1. 請根據以下敘述，撰寫一程式輸入「必要的資料」，根據以下規則計算後輸出所應收取的計程車費用。
   - 計程運價：起程1.25公里70元，續程每250公尺5元(即按計費表夜間加成收費)。
   - 延滯計時：車速每小時5公里以下，累計每1分鐘40秒5元。
   - 夜間加成：自夜間11時起至翌晨6時止(遇跨夜間加成時段之情形，統一以「上車時間」為準)，除按計費表夜間加成收費外，每旅次並加收20元。
   - 春節運價：自除夕前二日起至年假結束之期間，非夜間加成時段按計費表夜間加成收費外，每旅次再加收20元；夜間加成時段按計費表夜間加成收費外，每旅次再加收40元。

http://www.pto.taipei.gov.tw/ct.asp?xItem=1065628&ctNode=12593&mp=117041

2-2 & 2-3

2. 已知西元 1960 年為鼠年，撰寫一程式讓使用者輸入西元年份，並輸出該年份之生肖。
   - Hint: 假設輸入 1980, 1980 – 1960 = 20。20% 12 == 8, 餘 0,1, 2, ... 11 分別為屬鼠、牛、虎、 ... 豬。
   - Hint: 注意到 1959 – 1960 = -1, -1 % 12 == -1!!

3. 請撰寫一程式，分別輸入兩個圓的圓心與半徑，並由程式判斷此兩圓之間的關係為: a)沒關係、b) 相切、c) 同心圓、d) 部份重疊、e) 完全重疊。

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 \quad vs. \quad (r_1 + r_2)^2$$

3-1. 計算階乘表

- 請使用者輸入一正整數 x，接著由你的程式列出由 1 至 x 中間所

```
請輸入正整數 x 以計算它的階乘表：10
1!=1
2!=2
3!=6
4!=24
5!=120
6!=720
7!=5040
8!=40320
9!=362880
10!=3628800
```

並請回答電腦能準確地算到多少的階乘？(用你的程式試試看)

3-2. 列印數列

- 令 x 與 y 為你的學號的後面數起非零、非 1、且相異的兩個數字
  - e.g. 你的學號為 B9505532 → x=3, y=2
  - e.g. 你的學號為 B9705510 → x=5, y=7
- 請寫一程式，將 1 – 50 之間 x 和 y 的倍數都挑掉，列印出剩下的數字。

學號為 B9505047:
1, 2, 3, 5, 6, 9, 10, 11, 13, 15, 17, 18, 19, 22, 23, 25, 26, 27, 29, 30, 31, 33, 34, 37, 38, 39, 41, 43, 45, 46, 47, 50,

4-1. 統計

- 請寫一程式由使用者輸入「任意」數量的數值，直到輸入 0 為止，並計算使用者輸入資料的 1) 總和、2) 平均值 (average)、3) 所有資料裡的最大值、以及 4)所有資料裡的最小值。

- 提示：
  - 在資料輸入過程中可以順便累加總和
  - 在資料輸入過程中可以判斷目前輸入資料與之前輸入之最大或最小值的關係，並更新目前輸入之最大或最小值。
  - 使用無窮迴圈 for(;;) 與 break;

4-2. 單層迴圈

- 請讓使用者輸入 n ，並計算亦於 1 至 n 之間所有奇數的和。

4-3. 雙層迴圈

- 請寫一程式由使用者輸入 n，透過你的程式輸出介於 1 與 n 之間的所有質數，且每一行不可印出超過10個質數、並最後輸出總共找到幾個質數。

```
請輸入正整數 n：100
2, 3, 5, 7, 11, 13, 17, 19, 23, 29,
31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97,

總共有25個質數
```

4-4. 雙層迴圈

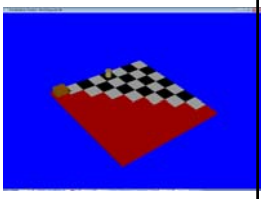
- 請寫一程式讓使用者輸入 n, 並由程式根據 n 輸出以下圖形。

```
請輸入 n: 5

*
**
***
****
*****
*****
*****
*****
*****
****
***
**
*
```


4-5. 推土機畫圖

- 請下載 http://140.118.105.174/Courses/CVB/2012/HW03/___.zip
- 請利用提供給你的專案，利用雙層 for 迴圈完成如右之圖形。
  - 欲將地板填為紅色請使用 car.paint().
  - 檢查前方是否有貨物請使用 car. IsFrontMoveAble(false)，回傳 true表示前方一格處有土堆會擋路。



5-1. 一維陣列

- 請寫一程式讓使用者輸入 x 與 y,，爾後宣告兩陣列 a[20]、b[20]，並於 a 陣列中填入以 x 為首項、y為公差的等差級數；於 b 陣列中填入以 x 為首、y為公比的等比級數。爾後分別將之印出來
  - E.g. x=3, y=2
  - a: 3, 5, 7, 9, 11, 13, 15, …
  - b: 3, 6, 12, 24, 48, 96, 192, …
- 接著，
  - 計算 a 陣列的總和，看看是否與等差級數和的公式相等
  - 計算 b 陣列的總和，看看是否與等比級數和的公式相等

$$S_n = \frac{n\left[2a_1 + (n-1)d\right]}{2} \qquad S_n = \frac{a_1\left(r^n - 1\right)}{r-1}$$

5-2. 字串操作與一維陣列

- 請寫一程式讓使用者輸入任意字串 (<500)，然後
  1) 找出使用者輸入字串的長度
  2) 每次將字串中所有字元向右搬一格、並將最尾端的字元搬到字串最開頭後印出，直到字串復原為止。

  提示:
  a) 要找字串的長度, 即計算該字串中從頭開始到遇到字元代碼 0 之前共有幾個字元。
  b) 請不要使用 strlen 函式，因為我們還沒教到 …

執行結果

```
請輸入一個字串: ABCDEFG
字串長度: 7
GABCDEF
FGABCDE
EFGABCD
DEFGABC
CDEFGAB
BCDEFGA
ABCDEFG
```

```
請輸入一個字串: 1234567890ABC
字串長度: 13
C1234567890AB
BC1234567890A
ABC1234567890
0ABC123456789
90ABC12345678
890ABC1234567
7890ABC123456
67890ABC12345
567890ABC1234
4567890ABC123
34567890ABC12
234567890ABC1
1234567890ABC
```

5-3. 陣列與迴圈的使用

- 請完成以下的程式，並自行初始化陣列 c 的內容，並讓你完成的程式根據 c 的內容來印出不同數量的米字號 (見下頁範例)

```
int c[10] = { …. };

// 接下來根據 c 的內容來印出如
.
.
.
```

執行結果

```
int c[10]=
 {1,4,2,3,7,4,6,0,0,0};
```

```
*
****
**
***
*******
****
******
```

```
int c[10]=
 {1,2,3,4,5,6,7,8,9,10};
```

```
*
**
***
****
*****
******
*******
********
*********
**********
```


5-4. 推土機找土堆

- 請下載 http://140.118.105.174/Courses/CVB/2012/HW04/____.zip (底線部份以自己學號取代)
- 請利用提供給你的專案，利用 while 或是 do...while 完成此題
- 目的是讓推土機自己找到土堆，並停留在它的前方。而土堆的位置確定是在棋盤的四週，但位置不固定，且請不要限定棋盤大小 (不要去數它)，而是利用while / do while 去完成。而推土機的初始位置約略在棋盤正中央。
  - 除了前後移動、轉彎外，會使用到 while 或是 do/while 或是 if 配合以下兩個功能來完成本題
    - car.IsFrontMoveAble(true) 用來檢查前方 (不限定幾步內) 是否有土堆
    - car.IsFrontMoveAble(false) 用來檢查前方一格是否有土堆
  - p.s. 如果你試著把土堆搬起來的話，你會發現土堆還會在。它是取之不盡、用之不竭的土堆來源！