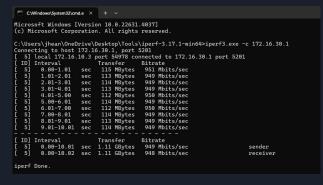
2. 輸入輸出 變數

我們的程式怎麼跟我們互動

程式與使用者互動的方式有很多種,最常見的是

- Command-Line Interface 指令列介面 使用文字輸入輸出 程式最簡單 最不容易壞
- Text-based User Interface 文字使用介面 使用方向鍵、快捷鍵 具有兩者優點
- Graphical User Interface 圖像化使用介面 使用圖形 最直覺 最方便使用者

我們這學期教的競程輸入輸出都是使用CLI



↑ CLI工具 (iperf)

TUI介面 (TrueNAS Setup)↓



怎麼讓程式輸入輸出

```
CLI的輸入/輸出可以稱為 stdin / stdout
```

C++中 最簡單的有下面兩組

C Style <cstdio> or <stdio.h>:

輸入 scanf() 輸出 printf()

C++ Style <iostream>:

輸入 std::cin 輸出 std::cout

因為std::cin std::cout比較直覺 所以我們這學期教這組

cin cout

```
cin cout.cpp
std::cin 會把輸入的值存入變數中 (後面會講)
用 >> 連接 cin和存入的變數 很多個變數可以連續用 >> 串起來
                                       #include <iostream>
std::cout 和cin相反 會將後方的值輸出到螢幕上
                                       using namespace std;
用 << 連接 cout 和資料 很多個資料可以用 << 串起來
'\n' 是換行字元
                                        int a;
std::endl 也是換行字元 都可以用
                                        string b;
                                        float c;
```

cin與輸入

```
cin 怎麼知道在哪裡分段呢
大原則: 空白, 換行字元(就是 Enter 鍵)
之後再考慮資料類型 (後面會講)
例如:
int a, b, c; // a, b, c 都只能是整數
cin >> a >> b >> c;
輸入為
1845611 -17 <- 這裡換行
-1
則 a = 1845611; b = -17; c = -1
```

資料型態 (bool, int)

```
bool - 布林值 (Boolean) 只有是 與 非
```

是: true, 1

非: false, 0

int - 整數 (integer) 為 2³¹ - 1 (2147483647) 到 -2³¹ (-2147483648) 的整數

變數

```
var_operators.cpp
```

```
C++中宣告變數的格式是
```

```
type name; //這樣name會是空的,沒有資料
type name = value; //這樣name的資料是value
```

```
a: 0
b: 0
c: 1
d: 1
e: 0
f: 1
g: -1
h: -2147483648
```

```
宣告一個變數的方式:
bool some_bool = true;
bool a = false;
bool b = 0;
bool c = true;
bool d = 1;
cout << "a: " << a << endl;
cout << "b: " << b << endl:
cout << "c: " << c << endl:
cout << "d: " << d << endl;
int e = 0:
int q = -1;
int h = -2147483648;
cout << "e: " << e << endl;
cout << "f: " << f << endl;
cout << "g: " << g << endl:
cout << "h: " << h << endl;
```

基本運算子

i = j;

```
      + - * / 就是 加減乘除

      a % b 是 a / b 的餘數 (% 念作 mod)

      a++ / a-- 是把a這個變數的值 +1 / -1

      前面有用到的 =

      用法是 var1 = var2; 意思是 把 var1 變成 var2 的值 i = 100;

      j = 101;
```

// i == 101, j == 101

```
cout << "i: " << i << endl;
cout << "i + 2: " << i + 2 << endl;
cout << "i - 2: " << i - 2 << endl:
cout << "i % 2: " << i % 2 << endl;
cout << "i after i++: " << i << endl;
cout << "i % 2 after i++: " << i % 2 << endl;
int k = 11:
cout << "j: " << j << endl;
cout << "k: " << k << endl;
```

基本運算子

```
在 = 前面放 + - * / % 會直接更改他的值 
1 = 100;
1 += 1; //這行跟 1 = 1 + 1 是一樣的效果 
則 1 == 101
```

```
l: 100
m: 100
l after l += 1: 101
m after m = m + 1: 101
```