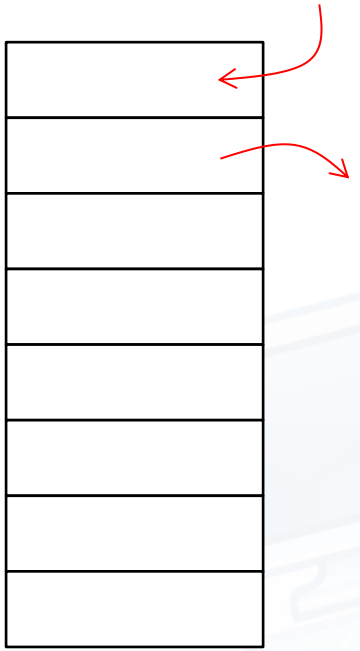
變數與輸入、輸出

記憶體

- 電腦裡面用來儲存資料的地方叫做「記憶體(memory)」
- 你可以把它想像成是一堆連續排列在一起的盒子
- 我們可以把東西放進去，也可以把東西拿出來



變數

- 我們可以把「變數」想像成是電腦裡一塊有名字的記憶體。
- 隨時可以對變數寫入資料或讀取其內的資料。
- 每個變數都有
  - 名字
    - 以底線'_'或英文字母開頭，後面為數字、底線或英文數字的字串
  - 型別
    - 如：整數(integer)、浮點數(floating-point number)、字串(string)……

宣告變數

- 每個變數在使用前都要先宣告(declare)
- 在宣告時，我們會指定該變數的型別和名字。
- 如：
  - int age; // 宣告一個整數(integer)型別，名為 age 的變數
  - string name; // 宣告一個字串(string)型別，名為 name 的變數

變數名稱

- 合法的變數名稱
  - [_ 或 英文字母]+[_ 或 英文字母 或 數字]
  - Age age length _name id1246
  - 注意：變數名稱有分大小寫，Age 和 age 是兩個不同的變數

- 不合法的變數名稱
  - 369city
  - my#name
  - 年齡

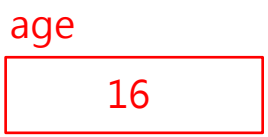
範例

```
#include <iostream>

using namespace std;

int main()
{
    int age;

    age = 16;

    cout << "你的年齡是 " << age << " 歲" << endl;

    return 0;
}
```

宣告整數型別的變數 age

將 16 這個值指派(賦值)給變數 age

讀取變數 age 的值




```
你的年齡是 16 歲
```

標準輸入 cin

- cout 可以將資料輸出到標準輸出(通常為螢幕)
- cin 則可以自標準輸入(通常為鍵盤)讀取資料

範例

```
#include <iostream>

using namespace std;

int main()
{
    int age;

    cout << "請輸入你的年齡:";
    cin >> age;
    cout << "你的年齡是 " << age << " 歲" << endl;

    return 0;
}
```

自標準輸入讀取值，並指派給 age

age

18

```
請輸入你的年齡: 18
你的年齡是 18 歲
```

為何要指定型別?

- 比較一下這兩個程式(執行看看結果有何不同)

```
#include <iostream>

using namespace std;

int main()
{
    int name;

    cout << "請輸入你的名字: ";
    cin >> name;
    cout << "你的名字是 " << name << endl;

    return 0;
}
```

```
#include <iostream>

using namespace std;

int main()
{
    string name;

    cout << "請輸入你的名字: ";
    cin >> name;
    cout << "你的名字是 " << name << endl;

    return 0;
}
```

```
#include <iostream>

using namespace std;

int main()
{
    int name;

    cout << "請輸入你的名字: ";
    cin >> name;
    cout << "你的名字是 " << name << endl;

    return 0;
}
```

```
請輸入你的名字: John
你的名字是 0
```

Why ?

```
#include <iostream>

using namespace std;

int main()
{
    string name;

    cout << "請輸入你的名字: ";
    cin >> name;
    cout << "你的名字是 " << name << endl;

    return 0;
}
```

```
請輸入你的名字: John
你的名字是 John
```

為何要指定型別?

- 比較一下這兩個程式(執行看看結果有何不同)

```
#include <iostream>

using namespace std;

int main()
{
    int D;

    cout << "請輸入直徑: ";
    cin >> D;
    cout << "半徑為 " << D/2 << endl;

    return 0;
}
```

```
#include <iostream>

using namespace std;

int main()
{
    float D;

    cout << "請輸入直徑: ";
    cin >> D;
    cout << "半徑為 " << D/2 << endl;

    return 0;
}
```

常用的基本資料型別與其範圍

名稱	關鍵字	大小	範圍	備註
字元	char	1 Byte	0x00 ~ 0xFF	1. ASCII 2. 如：'a', '@'
整數	int	4 Byte	-2^{31} ~ $2^{31}-1$	如：12, -65
無號整數	unsigned int	4 Byte	0 ~ $2^{32}-1$	如：23, 65372
單精確浮點數	float	4 Byte		1. IEEE 754 2. 如：3.14, 5.0
雙精確浮點數	double	8 Byte		
字串	string			如："Peter", "hello"

誤用型別的傷害

Jserv與他愉快的小夥伴
2016年7月4日 ·

1996 年 6 月 4 日，歐洲太空總署發射了一艘名為 "Ariane 5" 的火箭，造價不菲，火箭本身加上載運的人造衛星及科學儀器值 USD $7.5 billion (台幣約 2250 億)。

在當時，Ariane 系列火箭承包了全球商業衛星的發射約 50% 的業務量，而人們萬萬沒想到，Ariane 5 發射後 37 秒，升空到 4 公里處，火箭偏離預定飛行軌跡，解體並爆炸。

失事調查報告指出：水平加速偵測儀傳了一個 64-bit 浮點數給電腦，電腦用 16-bit 有號整數來接受這個浮點數，很不巧傳來的數字大於 32767，發生溢位 (Integer Overflow) 例外，這樣的軟體設計缺失很常見，但對 Ariane 5 來說，下場竟是火箭的推進向量噴嘴忽然朝某個方向轉到底，在高速偏航後，最終火箭解體。

開發 Ariane 5 火箭的軟體工程師在前一代 Ariane 4 中，小心翼翼地確認數值分佈， 確保水平速率的數值絕不會超過 16-bit 表達範圍以外。這些工程師在 Ariane 5 系統沿用這段程式碼，卻沒有檢查前述假設是否符合新的設計。

C 語言測試程式: https://github.com/hylin-embedded-2015/ariane5

OUTPUT:
d = 32767.000000, n = 32767
d = 32768.000000, n = -32768

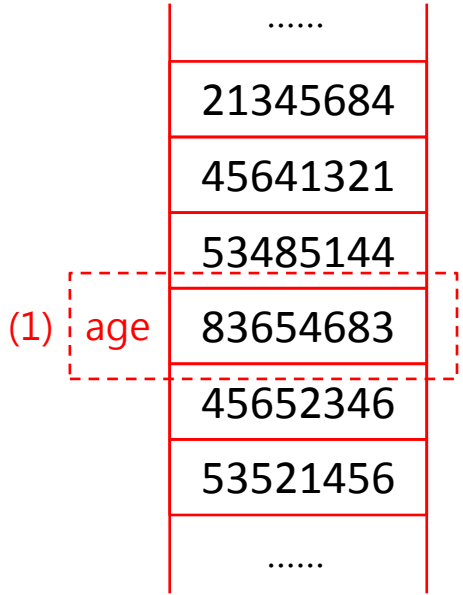
Ariane 5 rocket, Flight 501 - EPIC FAIL

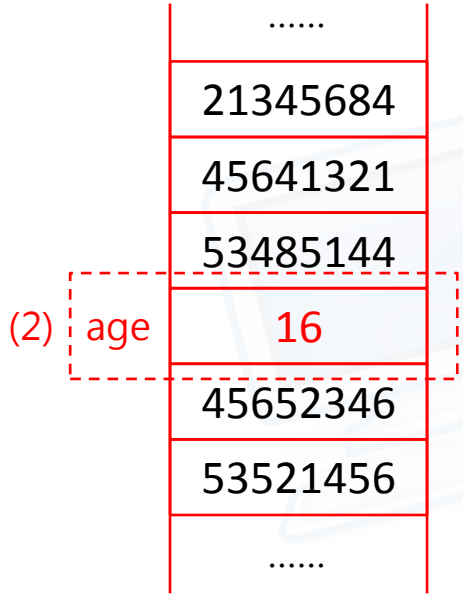
I hate the phrase "Epic Fail" but heck this is one case it truly fits... The guys face when it goes wrong is priceless, he waits a moment then jots...

YOUTUBE.COM

養成給變數指定初始值的習慣

- 宣告只是「幫我留一個空間用來放某某型別的變數」
- (1) int age;
  - 這時候，age的值是當下那塊記憶體的值
  - 我們無法確定它是多少
- (2) age= 16;
  - 這時候，我們才能確定age 現在的值是16





養成給變數指定初始值的習慣

```
#include <iostream>

using namespace std;

int main()
{
    int price;
    int quantity;

    cout << "請輸入你要購買的數量: ";
    cin >> quantity;
    cout << "總價是 " << price*quantity << endl;

    return 0;
}
```

未指定初值就讀取

```
請輸入你要購買的數量: 25
總價是 -2120130902
```

```
#include <iostream>

using namespace std;

int main()
{
    int price = 0;
    int quantity = 0;

    cout << "請輸入你要購買的數量: ";
    cin >> quantity;
    cout << "總價是 " << price*quantity << endl;

    return 0;
}
```

宣告後就指定初值

```
請輸入你要購買的數量: 25
總價是 0
```

= 運算子，它是指派(assign)不是等於(equal)

- 下面這個程式的執行結果為何?

```
#include <iostream>

using namespace std;

int main()
{
    int a = 3;

    a = a + 1;

    cout << a << endl;

    return 0;
}
```

(1) 3

(2) 4 ✓

(3) 無限大

(4) 無法確定

= 運算子，它是指派(assign)不是等於(equal)

- 下面這個程式的執行結果為何?

```
#include <iostream>

using namespace std;

int main()
{
(1)  int a = 3;

(2)  a = a + 1;

     cout << a << endl;

     return 0;
}
```

(1) a 3

(2) 先把 = 右邊的運算結果求出

a = a + 1;

4

a = 4; 再把運算結果指派給 = 左邊的變數

a 4

= 運算子，它是指派(assign)不是等於(equal)

- 下面這個程式的執行結果為何?

```
#include <iostream>

using namespace std;

int main()
{
    int a = 3;
    int b = 5;

    a = a + b;
    b = a - b;
    a = a - b;

    cout << a << ":" << b << endl;

    return 0;
}
```

(1) 3:5

(2) 8:3

(3) 5:3 ✓

(4) -2:-2