台南一中第一次資訊段考複習-特殊版本

基礎注意事項

1. 變數型態

C++中有許多變數型態，像是整數、浮點數(小數)、字串(第一次不會考)等等。

下表整理了常見的變數型態

類別	變數型態	範圍
整數	int	±2147483647
整數	long int	±2147483647
整數	long long int ◎	±9223372036854775807
正整數	unsigned long long int ◎	18446744073709551615
小數	float	有效位數 7 位
小數	double ◎	有效位數 15 位

unsigned 是無號的型態，因此若使用他，就必須確保數字不是負數。

標示 ◎ 為推薦使用的變數型態，有時候題目的數字出很大，不妨使用這些變數型態，說不定就會解決一些奇怪的錯誤了!

必須注意: main 一定要用 int

2. 標頭檔

這個只對於非 403 教室的同學有效，因為 Mac 沒有這個標頭檔，但你上傳上去還是可以用。

平常我們在寫程式的時候老師都會教我們引用 iostream

```
#include<iostream>
```

但如果你要使用一些特定功能，如控制輸出位數就要引入 math.h

```
#include<math.h>
```

十分麻煩，因此教你一招: 引入 bits/stdc++.h

```
#include<bits/stdc++.h>
```

這樣所有標頭檔都幫你引用好了，不須多花心思在這上面

3. TLE 超過時限?(雖然發生的機率微乎其微)

當你辛辛苦苦寫出的程式丟到 SkyOJ 卻 TLE，不妨試試下面兩招吧!

- 在 main 裡寫下【`ios::sync_with_stdio(0);cin.tie(0);cout.tie(0);`】

但使用這一招會有一缺點，就是你不能 printf 和 cout 一起混用。

- 在#include 後面寫下#define endl '\n'或是將程式碼內所有 endl 改成'\n'會快一點點喔!

4. 小數整數互換問題

題目應該會出到這種問題，這種問題的解決辦法是變數轉型要轉好，在原本是 int 的變數前面加上(double)即可強制轉型。

※遇到加減乘除運算即使是對一個整數，也必須把那個數字後面加上.0 不然會被轉為 int

例子:

```
double a = 10.0 , b = 20.5
double result = (a+b) / 2.0
```

5. 小數輸出控制

我認為這是必考題，小數輸出位數控制有兩種方法

假設有一個小數 a，控制輸出至小數點後第 2 位。

- `cout << fixed << setprecision(2) << a << endl;`
- `print("%.2f\n",a)`

6. 智障語法錯誤

請記得在每一行後面加上分號

另外，請在本機先跑過再上傳到 Sky0J。

7. 數學公式

記得背一下數學公式

- 海龍 :

$$S = \frac{(a+b+c)}{2}$$

$$三角形面積 = \sqrt{S*(S-a)*(S-b)*(S-c)}$$

- 二元一次方程式

$$x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$$

- 二項式定理

$$(x+a)^n = \sum_{k=0}^{n} \binom{n}{k} x^k a^{n-k}$$

- 銳角三角形、鈍角三角形、直角三角形

$$\text{直角三角形: } a^2 + b^2 = c^2$$
$$\text{銳角三角形: } a^2 + b^2 > c^2$$
$$\text{鈍角三角形: } a^2 + b^2 < c^2$$

- 矩陣相乘(?

8. 奇偶數、因數判斷

這是基本功，拜託一定不要錯。

偶數判斷: `a%2 == 0`

奇數判斷: `a%2 == 1`

因數判斷: `a%b == 0`（b 為 a 的因數）

9. `if else`

`if else` 很基本、沒什麼技術難度，但在考題上可能會有很多變化，謹記先後關係，一層一層判斷。

10. 迴圈

for 迴圈必考，謹記語法。

while 迴圈也是相同概念。

for(初始值;條件;改變量)

while(條件)

11. 交換兩個變數

有兩個變數 a,b，今天要交換 ab 使 a 為 b，b 為 a

簡單版:

```
swap(a,b)
```

三行版

```
int _ = a;a = b;b = _;
```

12. 善用資源

老師在考試時會鎖外網，因此無法使用 Google，但這個時候，有些東西就很時用啦，像是

- Moodle（https://moodle.tnfsh.tn.edu.tw）你寫過的作業
- TOJ（https://toj.tfcis.org）你練過的題目

這些在內網的都還是連得上的，突然忘了可以查一下。

13. 強迫取分

當你不會寫題目的時候，請記得仍有部分分數，仔細看看題序，想一想有甚麼可能的答案，直接 cout 出去，這樣說不定還有幾十分。

迴圈使用範例

市政質詢抽抽樂: 判斷是否有 87 這個數字

```
int trash;cin >> trash;
int x;
while (cin >> x) {
  if(x==87){
    cout << "YES" << endl;
    return 0;
  }
}
cout << "NO" << endl;
return 0;
```

持續讀入數字，當數字為 87 時，輸出 YES 並立即停止程式(return 0;)

有些人會問為甚麼 return 0 會結束程式呢?

這是因為當一個函式 return 時，函式就會終止，而 main 這個進入點也是一個函式，因此 return 後就會結束執行也就是結束整個程式。

Flag 的使用時機

當題目要求你做兩件事以上，但你無法在同一時間做完，那麼可以使用 Flag，這個 Flag 就像是一個按鈕，記錄你「是否做過某事」，那麼廢話不多說，直接用程式演練看看吧!

因數和: 給定一個數字，求其正因數和及是否為質數。

若一個數字為質數，那麼他就沒有除了 1 和他本身以外的因數，我們使用一個 for 迴圈列舉出該數之所有因數，並算出其總和，同時也得確認是否為質數，這麼時候我們就可以用 Flag 了，先在前面將 Flag 設為 False(還沒執行過、沒有因數)，而在 for 迴圈中，如果有找到因數的話，那就將 Flag 設為 True，最後我們可以將所有的因數加總，算出因數和，再利用 Flag 進行判斷，倘若 Flag 為 True 那麼代表此數為合數，反之，則為質數。

```
int n;cin >> n;
int result=0;bool notPrime = false;
for(int i=1;i<n;i++){
  if(n%i==0){
    notPrime = true;
    result += i;
  }
}
if(notPrime)cout << "XD" << endl;
else cout << result << endl;
```

費氏數列的解法

- 陣列建表法

```
int Fi[100000+5] = {0};
int main() {
    Fi[0] = 0;Fi[1] = 1;
    for(int i=2;i<100000;i++){
      Fi[i] = Fi[i-1] + Fi[i-2];
    }
    int ask;cin >> ask;
    //第N項
    cout << Fi[ask] << " ";
    //前N項
    for(int i=0;i<ask;i++)cout << Fi[i] << " ";
     return 0;
}
```

- 遞迴法

```
int dp[100000+5] = {0};
int Fi(int n) {
  if(n == 1) {
      return 1;
  } else if(n <= 0) {
      return 0;
  } else {
      if(!dp[n]){
          int x = Fi(n-1) + Fi(n-2) ;
          dp[n] = x;
          return x;
      }else{
```

```
            return dp[n];
        }
    }
  }
  int main() {
    for(int i=0; i<10; i++)cout << Fi(i) << " ";
    return 0;
  }
```

陣列 (by 鄭弘煒)

陣列是一個鏈狀的資料結構。但究竟和一般的變數差別在那裡呢？

1. 陣列可以用相同的變數名稱建立一個大型的空間存放資料。

2. 陣列跟數學的集合似乎差異不大。

建立陣列：

資料結構型態 變數名稱[資料大小];

EX: `int Array[100];` 命名一個叫 Array 的陣列且共有 100 個儲存空間（索引值從 0 到 99）

使用陣列：

1. 於初始化時可順便賦值

EX：`int Array[8] = {1,2,5,4,8,6,2,74};`

2. 在設定變數大小後隨意更改項目

EX：`int Array[8];`

`Array[0] = 1;Array[1] = 2;…`

Array[0] 就好比是一個變數可以隨意更改

實際使用：

```
int Array[100];
int N;cin >> N;
for(int i = 0;i < N;++i)cin >> Array[i];
```

N 為一個初始值，在這個程式中 N 不可大於 100。

如此一來，利用陣列我們就可以很輕鬆地把資料儲存起來，不再需要大量的變數了。

DP (by 玹、參考 SA、Algorithm Note)

DP 的核心精神就是將大問題切分為小問題，透過小問題去解決大問題。

以階層為例，假設今天要算兩個階層，5! 和 8!，那麼你會發現，5!=5*4*3*2*1，

8!=8*7*6*5*4*3*2*1，你可能會想，那就用一個 for 迴圈慢慢乘下去就好了，但這樣子會導致執行的時間太久，有時候會 TLE，我們可以考慮開一個陣列 dp，dp[n]就存 n 階層，那麼就只需要用 for 迴圈跑過一次，每一次將 dp[n-1] * n，再存到 dp[n]裡面，這樣就可以用比較節省時間的方式計算(算過的不重算)。

DP 還有很多經典的題目，但由於這是段復講義，而且這次段考應該考不多吧!可以下次再仔細研究，這個章節只是為了給你一種「想法」。

Ref : https://bit.ly/TNFSHinfo

Sort (by 玹)

這裡的 sort 只教程式碼，不教理論的部分，為了速成(?

需要引入#include<algorithm>這個標頭檔

假設今天有一個名字叫做 Array、長度為 5 的陣列[3,9,4,2,1]

若我今天要把它變成[1,2,3,4,9]，你可以使用

```
sort(Array,Array+n);
```

若我今天要把它變成[9,4,3,2,1]，你可以使用

```
sort(Array,Array+n,greater<int>);
```

就是這短短的一行程式碼，可以用來排列陣列，而甚麼時候會用到，那就仔細看題目的詢問吧!

最後提醒，不論是哪一題，題序都有可能引導妳走多餘的路，試著將題目脈絡解析清楚，想想看有沒有甚麼更好的解法。

大概就這樣了，祝考試順利，段考每科都 100。

不會地記得要問，不論是問 Eva 還是問我都好，不要把問題帶到考場，資訊段考真的沒有很難，放輕鬆，妳一定會考得很好的。

第一次資訊段複謝詠晴版 / 編輯者: 張睿玹、鄭弘煒 20200515