

slido:~\$



函式 & 競賽簡介

函式 : ~ 函式 / 變數作用域 \$

變數作用域是 APCS 的常見考題之一
只要把握一個原則就可以解決這種問題

找向上最近區塊的變數

函式：~ 函式 / 變數作用域 / 範例一 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a=0;
5
6  √ int main(){
7
8      int a=0;
9      a++;
10     cout << a << endl;
11
12     return 0;
13 }
```

函式：~ 函式 / 變數作用域 / 範例一 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a=0;
5
6  ✓ int main(){
7
8      int a=0;
9      a++;
10     cout << a << endl;
11
12     return 0;
13 }
```

小區域

大區域

函式：~ 函式 / 變數作用域 / 範例二 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int main(){
5
6      int x, y, z;
7      cin >> x >> y >> z;
8
9      int i=0, j=0, k=0;
10     for ( ; i<x ; i++){
11         for ( ; j<y ; j++){
12             for ( ; k<z ; k++){
13                 // code
14             }
15         }
16     }
17
18     return 0;
19 }
```

函式 : ~ 函式 / 變數作用域 / 範例二 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int main(){
5
6      int x, y, z;
7      cin >> x >> y >> z;
8
9      int i=0, j=0, k=0;
10     for ( ; i<x ; i++){
11         for ( ; j<y ; j++){
12             for ( ; k<z ; k++){
13                 // code
14             }
15         }
16     }
17
18     return 0;
19 }
```

函式：~ 函式 / 變數作用域 / 範例三 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a=0;
5
6  void add(){
7      a++;
8  }
9
10 int main(){
11     int a=0;
12
13     add();
14     cout << a << endl;
15
16     return 0;
17 }
```


函式：~ 函式 / 變數作用域 / 範例三 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a=0;
5
6  void add(){
7      a++;
8  }
9
10 int main(){
11     int a=0;
12
13     add();
14     cout << a << endl;
15
16     return 0;
17 }
```

函式：~ 函式 / 變數作用域 / 範例三 \$

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a=0;
5
6  void add(){
7      a++;
8  }
9
10 int main(){
11     int a=0;
12
13     add();
14     cout << a << endl;
15
16     return 0;
17 }
```

函式：~ 函式 / 變數作用域 \$

```
variable-scope1.cpp > main()
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 // declare
5 int a=0;
6
7 // function
8 void add(int a){
9     a++;
10 }
11
12 int main(){
13
14     add(a);
15     cout << a << endl;
16
17     return 0;
18 }

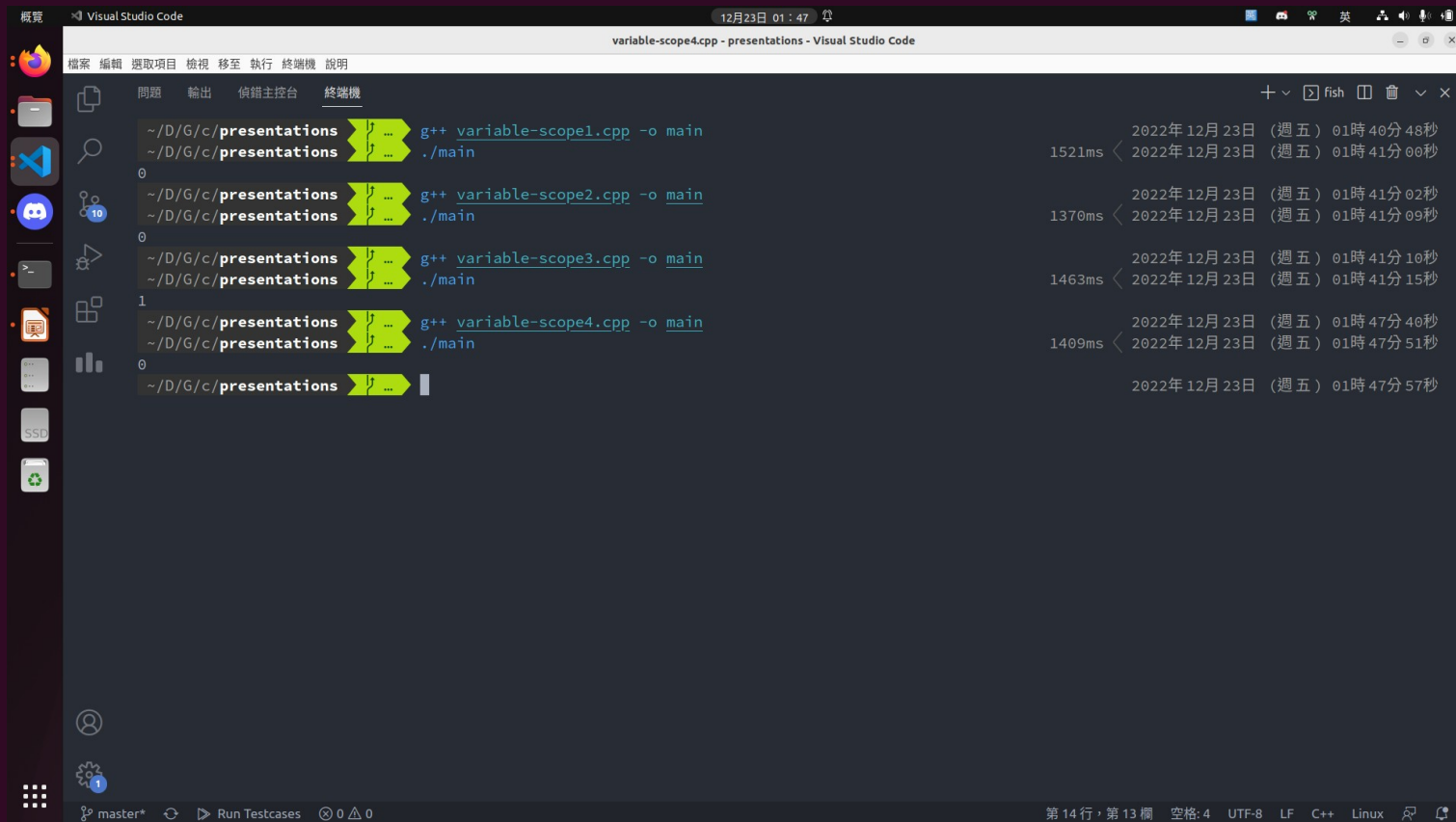
variable-scope2.cpp > main()
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 // function
5 void add(int a){
6     a++;
7 }
8
9 int main(){
10
11     int a=0;
12     add(a);
13     cout << a << endl;
14
15     return 0;
16 }

variable-scope3.cpp > main()
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 // declare
5 int a=0;
6
7 // function
8 void add(int num){
9     a++;
10 }
11
12 int main(){
13
14     add(a);
15     cout << a << endl;
16
17     return 0;
18 }

variable-scope4.cpp > main()
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 // declare
5 int a=0;
6
7 // function
8 void add(int num){
9     a++;
10 }
11
12 int main(){
13
14     int a=0;
15     add(a);
16     cout << a << endl;
17
18     return 0;
19 }
```

第 19 行, 第 2 欄 空格: 4 UTF-8 LF C++ Linux

函式：~ 函式 / 變數作用域 \$



The screenshot shows the Visual Studio Code interface with a terminal window open. The terminal displays the compilation and execution of four C++ programs, each demonstrating a different variable scope concept. The programs are named `variable-scope1.cpp` through `variable-scope4.cpp`. Each program is compiled using `g++` and executed using `./main`. The output of each execution is shown on the right side of the terminal, including the date and time of execution.

```
~/D/G/c/presentations } ... g++ variable-scope1.cpp -o main
~/D/G/c/presentations } ... ./main
0
2022年 12月 23日 (週五) 01時 40分 48秒
1521ms < 2022年 12月 23日 (週五) 01時 41分 00秒

~/D/G/c/presentations } ... g++ variable-scope2.cpp -o main
~/D/G/c/presentations } ... ./main
0
2022年 12月 23日 (週五) 01時 41分 02秒
1370ms < 2022年 12月 23日 (週五) 01時 41分 09秒

~/D/G/c/presentations } ... g++ variable-scope3.cpp -o main
~/D/G/c/presentations } ... ./main
1
2022年 12月 23日 (週五) 01時 41分 10秒
1463ms < 2022年 12月 23日 (週五) 01時 41分 15秒

~/D/G/c/presentations } ... g++ variable-scope4.cpp -o main
~/D/G/c/presentations } ... ./main
0
2022年 12月 23日 (週五) 01時 47分 40秒
1409ms < 2022年 12月 23日 (週五) 01時 47分 51秒

~/D/G/c/presentations } ... 
2022年 12月 23日 (週五) 01時 47分 57秒
```

Visual Studio Code status bar at the bottom shows: master* | Run Testcases | 0 0 0 | 第 14 行, 第 13 欄 | 空格: 4 | UTF-8 | LF | C++ | Linux |

競賽概論：~\$

我們終於完成所有語法課程了：D

明天到下一次社課前就會開放社內賽，這次的社內賽是偏競程向的
所以這裡介紹一些競程的賽制以及寫題技巧

競賽概論：~ 賽制說明 / IOI\$

本次的賽制是 IOI 制：有以下的特色

- 有部份分數
- 沒有罰時
- 即時反饋

例：市賽、全國賽（多為高中端）

競賽概論：~ 賽制說明 / ICPC\$

另外有兩個主要的賽制為 ICPC 制和 OI 制

ICPC：

- 沒有部份分數
- 有罰時
- 即時反饋

例： ICPC、codeforces （多為大學端比賽）

競賽概論：~ 賽制說明 /OI\$

OI：

- 有部份分數
- 沒有罰時
- 非即時反饋

例：APCS

競賽概論：~ 寫題技巧 \$

不一定要全部寫完，子任務的分數總和也是相當可觀

1	11102021 (高雄高級中學)			70%	71%	73%	48%	24%	6%	57.9	2	1	1
2	11102014 (高雄高級中學)			7%		72%	24%	75%	6%	57.3	3	8	2
3	11102027 (高雄高級中學)			7%			-	36%	6%	51.7	4	224	3
4	11102019 (高雄高級中學)			7%			-	0%	6%	45.9	4	109	4
5	11102030 (高雄高級中學)			7%	57%		0%	36%	6%	45.7	3	158	5
6	11102026 (新莊高中)			7%		66%	6%	12%	6%	43.4	3	198	6
7	11102042 (高雄高級中學)			-		94%	-	-	-	43	3	92	7
8	11102024 (高雄女中)			0%		73%	-	-	0%	39.7	3	94	8
9	11102034 (高雄高級中學)			0%		73%	-	0%	-	39.7	3	109	9
10	11102009 (高雄高級中學)			7%		54%	0%	-	-	37.6	3	199	10