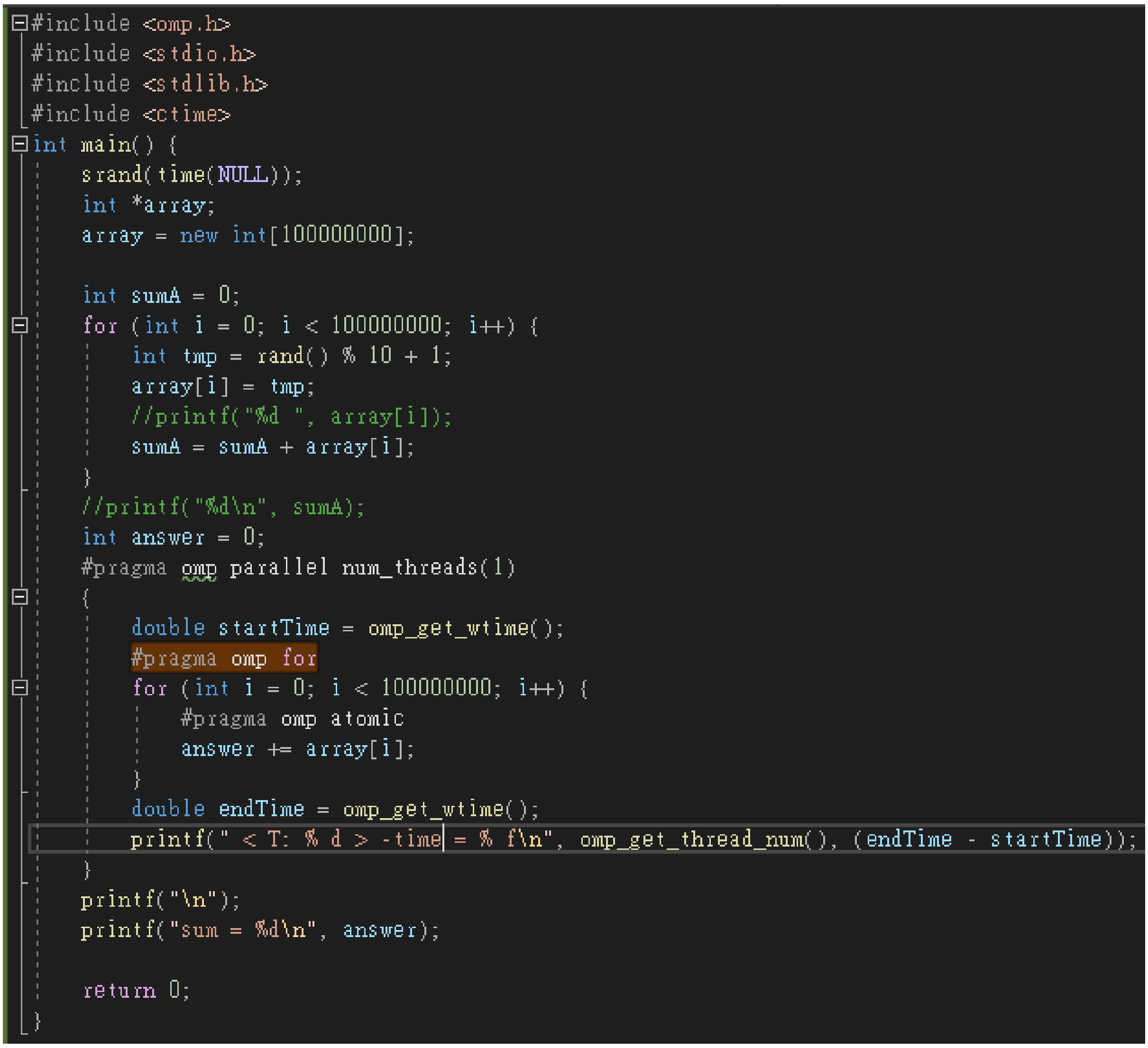
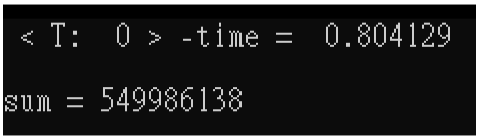
P=1



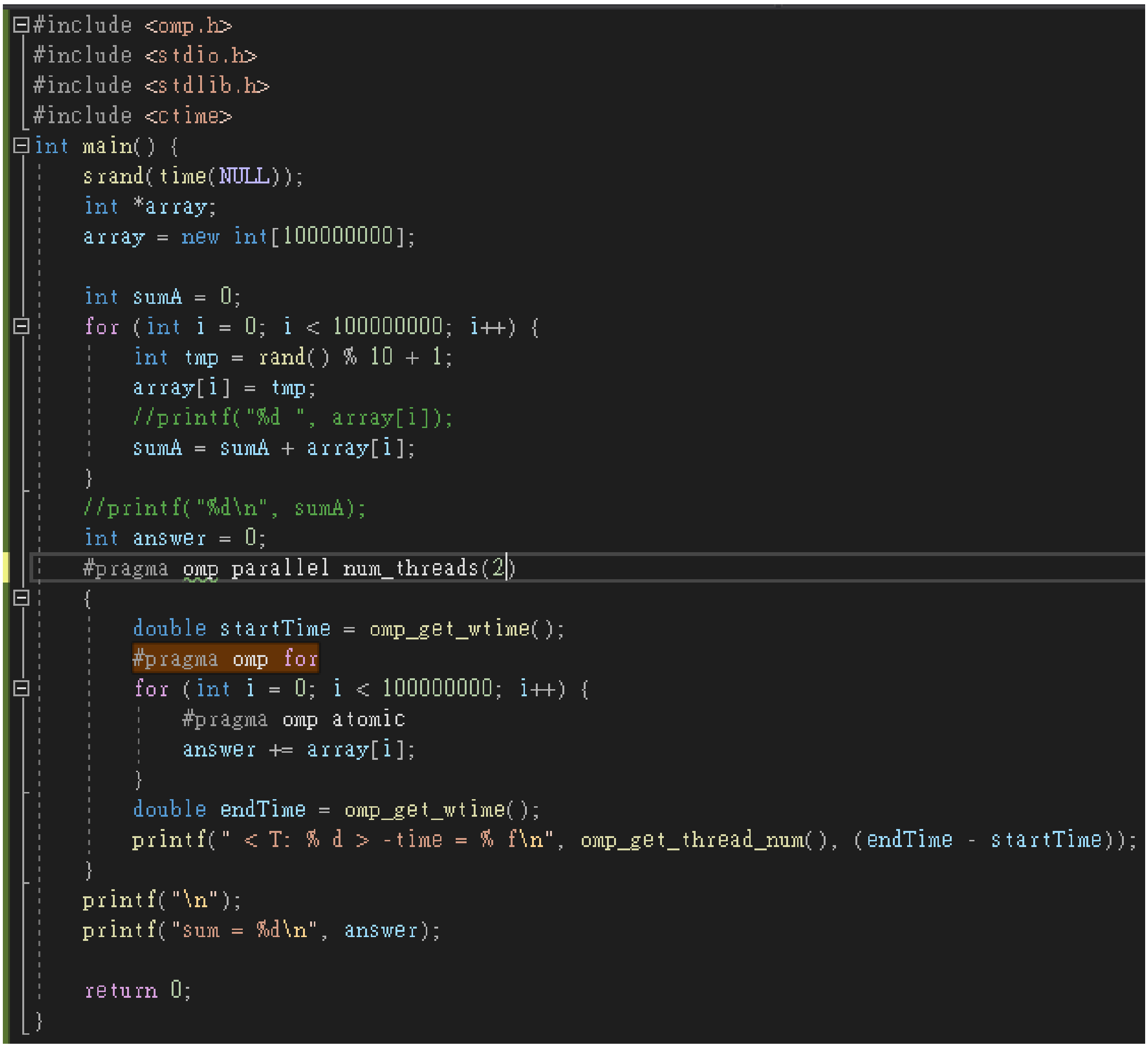
```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
int main() {
    srand(time(NULL));
    int *array;
    array = new int[100000000];

    int sumA = 0;
    for (int i = 0; i < 100000000; i++) {
        int tmp = rand() % 10 + 1;
        array[i] = tmp;
        //printf("%d ", array[i]);
        sumA = sumA + array[i];
    }
    //printf("%d\n", sumA);
    int answer = 0;
    #pragma omp parallel num_threads(1)
    {
        double startTime = omp_get_wtime();
        #pragma omp for
        for (int i = 0; i < 100000000; i++) {
            #pragma omp atomic
            answer += array[i];
        }
        double endTime = omp_get_wtime();
        printf(" < T: % d > -time = % f\n", omp_get_thread_num(), (endTime - startTime));
    }
    printf("\n");
    printf("sum = %d\n", answer);

    return 0;
}
```

結果:



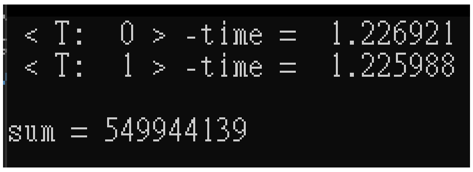
```
 < T:  0 > -time =  0.804129

sum = 549986138
```

P=2



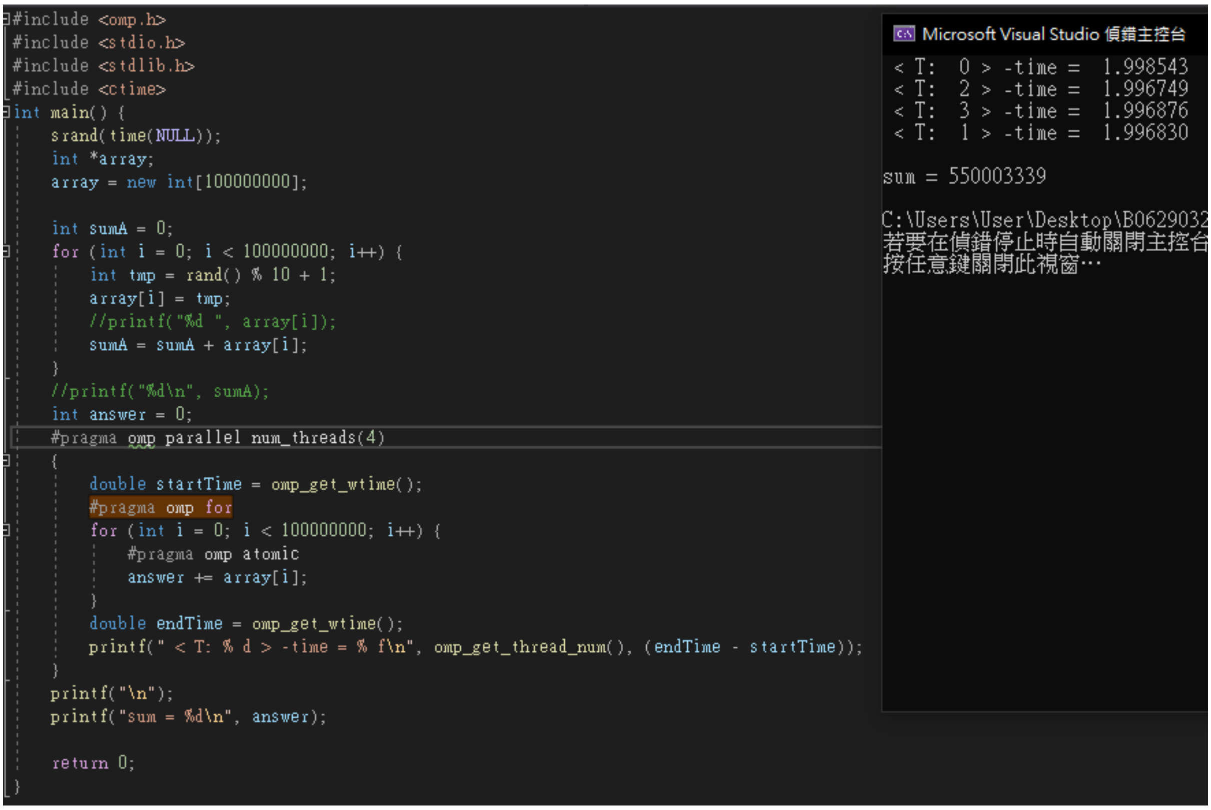
```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
int main() {
    srand(time(NULL));
    int *array;
    array = new int[100000000];

    int sumA = 0;
    for (int i = 0; i < 100000000; i++) {
        int tmp = rand() % 10 + 1;
        array[i] = tmp;
        //printf("%d ", array[i]);
        sumA = sumA + array[i];
    }
    //printf("%d\n", sumA);
    int answer = 0;
    #pragma omp parallel num_threads(2)
    {
        double startTime = omp_get_wtime();
        #pragma omp for
        for (int i = 0; i < 100000000; i++) {
            #pragma omp atomic
            answer += array[i];
        }
        double endTime = omp_get_wtime();
        printf(" < T: % d > -time = % f\n", omp_get_thread_num(), (endTime - startTime));
    }
    printf("\n");
    printf("sum = %d\n", answer);

    return 0;
}
```

結果:



```
 < T:  0 > -time =  1.226921
 < T:  1 > -time =  1.225988

sum = 549944139
```

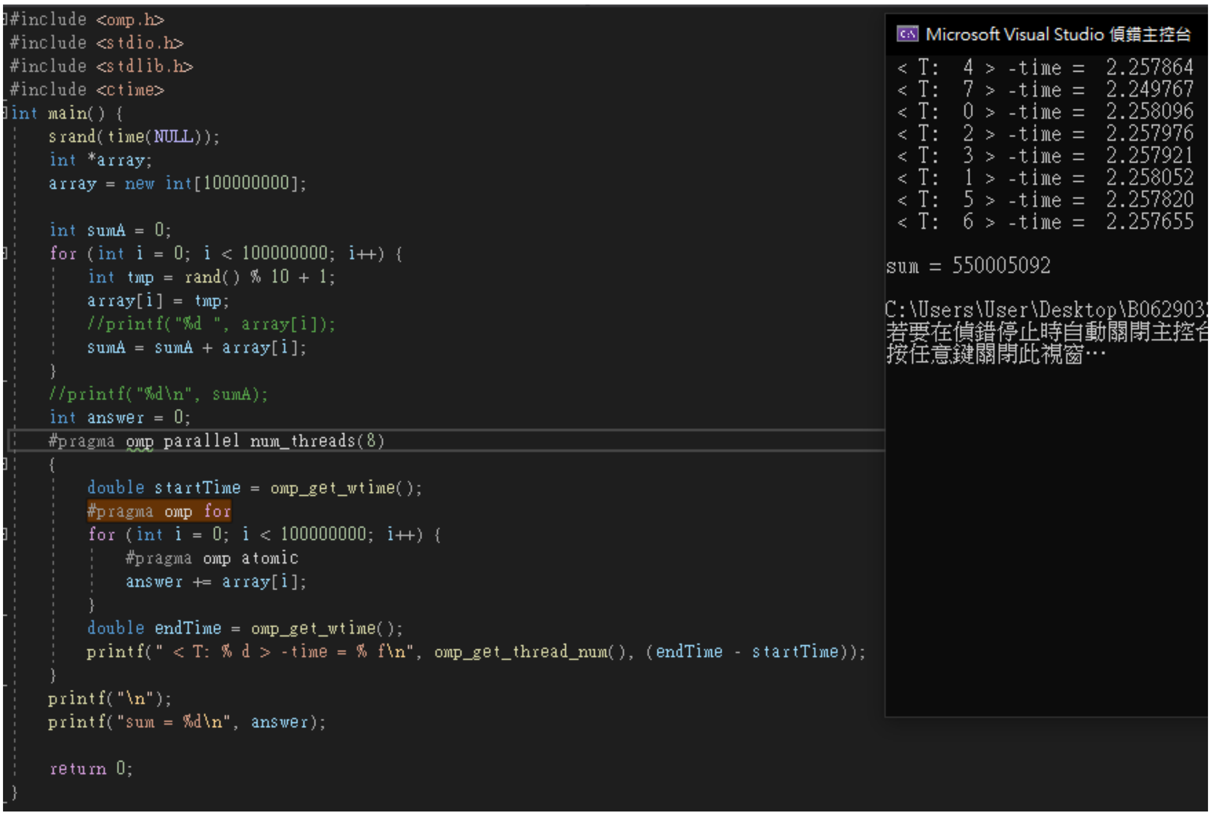
P=4(右方是結果圖)



```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
int main() {
    srand(time(NULL));
    int *array;
    array = new int[100000000];

    int sumA = 0;
    for (int i = 0; i < 100000000; i++) {
        int tmp = rand() % 10 + 1;
        array[i] = tmp;
        //printf("%d ", array[i]);
        sumA = sumA + array[i];
    }
    //printf("%d\n", sumA);
    int answer = 0;
    #pragma omp parallel num_threads(4)
    {
        double startTime = omp_get_wtime();
        #pragma omp for
        for (int i = 0; i < 100000000; i++) {
            #pragma omp atomic
            answer += array[i];
        }
        double endTime = omp_get_wtime();
        printf(" < T: % d > -time = % f\n", omp_get_thread_num(), (endTime - startTime));
    }
    printf("\n");
    printf("sum = %d\n", answer);

    return 0;
}
```

```
 < T:  0 > -time =  1.998543
 < T:  2 > -time =  1.996749
 < T:  3 > -time =  1.996876
 < T:  1 > -time =  1.996830

sum = 550003339

C:\Users\User\Desktop\B0629032
若要在偵錯停止時自動關閉主控台
按任意鍵關閉此視窗…
```

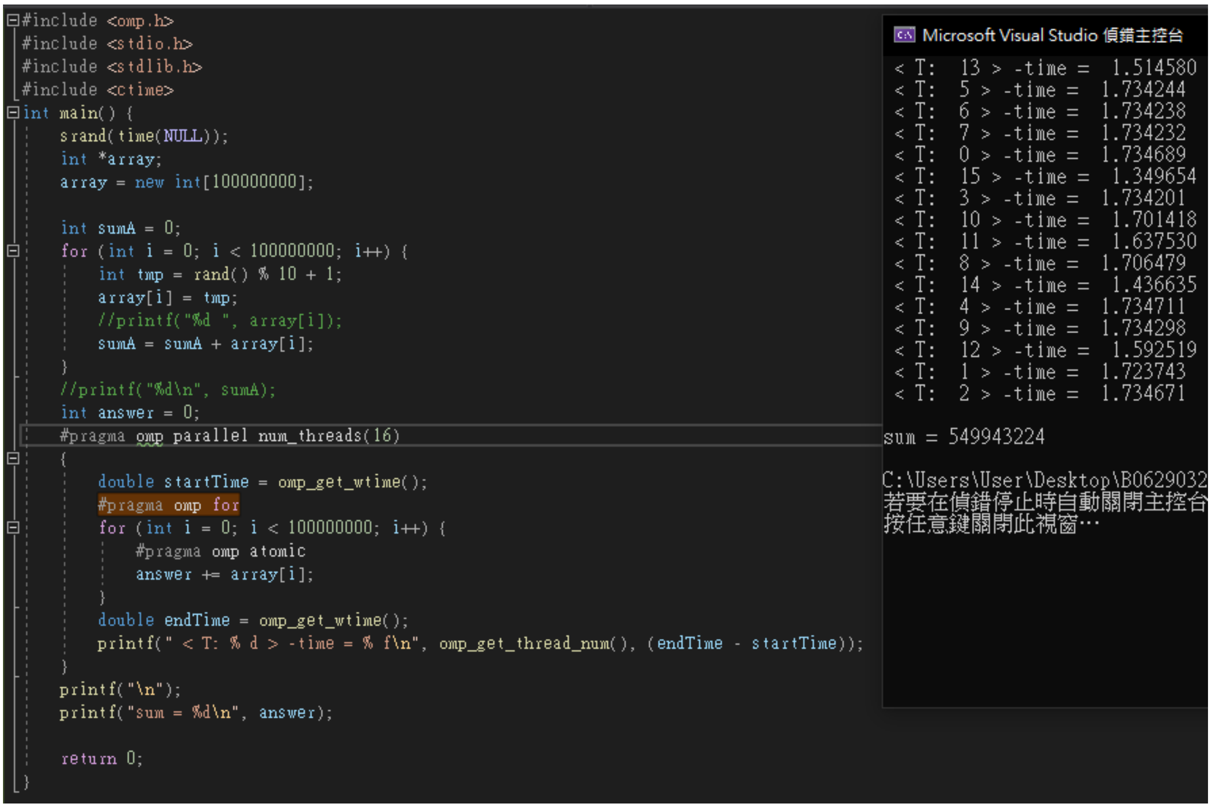
P=8



```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
int main() {
    srand(time(NULL));
    int *array;
    array = new int[100000000];

    int sumA = 0;
    for (int i = 0; i < 100000000; i++) {
        int tmp = rand() % 10 + 1;
        array[i] = tmp;
        //printf("%d ", array[i]);
        sumA = sumA + array[i];
    }
    //printf("%d\n", sumA);
    int answer = 0;
    #pragma omp parallel num_threads(8)
    {
        double startTime = omp_get_wtime();
        #pragma omp for
        for (int i = 0; i < 100000000; i++) {
            #pragma omp atomic
            answer += array[i];
        }
        double endTime = omp_get_wtime();
        printf(" < T: % d > -time = % f\n", omp_get_thread_num(), (endTime - startTime));
    }
    printf("\n");
    printf("sum = %d\n", answer);

    return 0;
}
```

```
 < T:  4 > -time =  2.257864
 < T:  7 > -time =  2.249767
 < T:  0 > -time =  2.258096
 < T:  2 > -time =  2.257976
 < T:  3 > -time =  2.257921
 < T:  1 > -time =  2.258052
 < T:  5 > -time =  2.257820
 < T:  6 > -time =  2.257655

sum = 550005092

C:\Users\User\Desktop\B062903
若要在偵錯停止時自動關閉主控台
按任意鍵關閉此視窗…
```

P=16(右方是結果圖)



```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
int main() {
    srand(time(NULL));
    int *array;
    array = new int[100000000];

    int sumA = 0;
    for (int i = 0; i < 100000000; i++) {
        int tmp = rand() % 10 + 1;
        array[i] = tmp;
        //printf("%d ", array[i]);
        sumA = sumA + array[i];
    }
    //printf("%d\n", sumA);
    int answer = 0;
    #pragma omp parallel num_threads(16)
    {
        double startTime = omp_get_wtime();
        #pragma omp for
        for (int i = 0; i < 100000000; i++) {
            #pragma omp atomic
            answer += array[i];
        }
        double endTime = omp_get_wtime();
        printf(" < T: % d > -time = % f\n", omp_get_thread_num(), (endTime - startTime));
    }
    printf("\n");
    printf("sum = %d\n", answer);

    return 0;
}
```

```
 < T:  13 > -time =  1.514580
 < T:  5 > -time =  1.734244
 < T:  6 > -time =  1.734238
 < T:  7 > -time =  1.734232
 < T:  0 > -time =  1.734689
 < T:  15 > -time =  1.349654
 < T:  3 > -time =  1.734201
 < T:  10 > -time =  1.701418
 < T:  11 > -time =  1.637530
 < T:  8 > -time =  1.706479
 < T:  14 > -time =  1.436635
 < T:  4 > -time =  1.734711
 < T:  9 > -time =  1.734298
 < T:  12 > -time =  1.592519
 < T:  1 > -time =  1.723743
 < T:  2 > -time =  1.734671

sum = 549943224

C:\Users\User\Desktop\B0629032
若要在偵錯停止時自動關閉主控台
按任意鍵關閉此視窗…
```