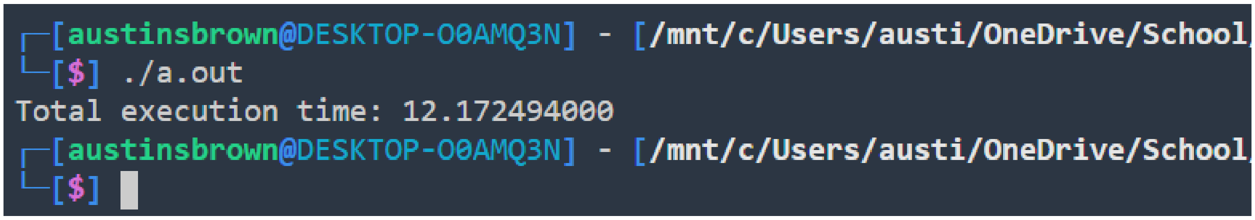
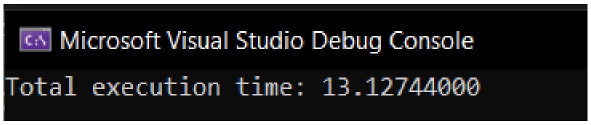
**1.**

```
#include <iostream>
#include <time.h>
using namespace std;

#define ARRSZE 20000

int main()
{
    struct timespec start, stop;
    clock_gettime(CLOCK_REALTIME, &start);

    int** array = new int*[ARRSZE];

    for(int i = 0; i < ARRSZE; ++i)
        array[i] = new int[ARRSZE];

    int i,j;
    for(i=0;i<ARRSZE;i++)
        for(j=0;j<ARRSZE;j++)
            array[j][i]=0; // cause cache misses

    for(int i = 0; i < ARRSZE; i++)

        delete[] array[i];
    delete[] array;

    clock_gettime(CLOCK_REALTIME, &stop);
    unsigned long long totalSeconds = (long long)(stop.tv_sec - start
.tv_sec);
    unsigned long totalNanoseconds = stop.tv_nsec - start.tv_nsec;

    printf("Total execution time: %llu.%.9lu\n", totalSeconds, totalN
anoseconds);
    return 0;
}
```



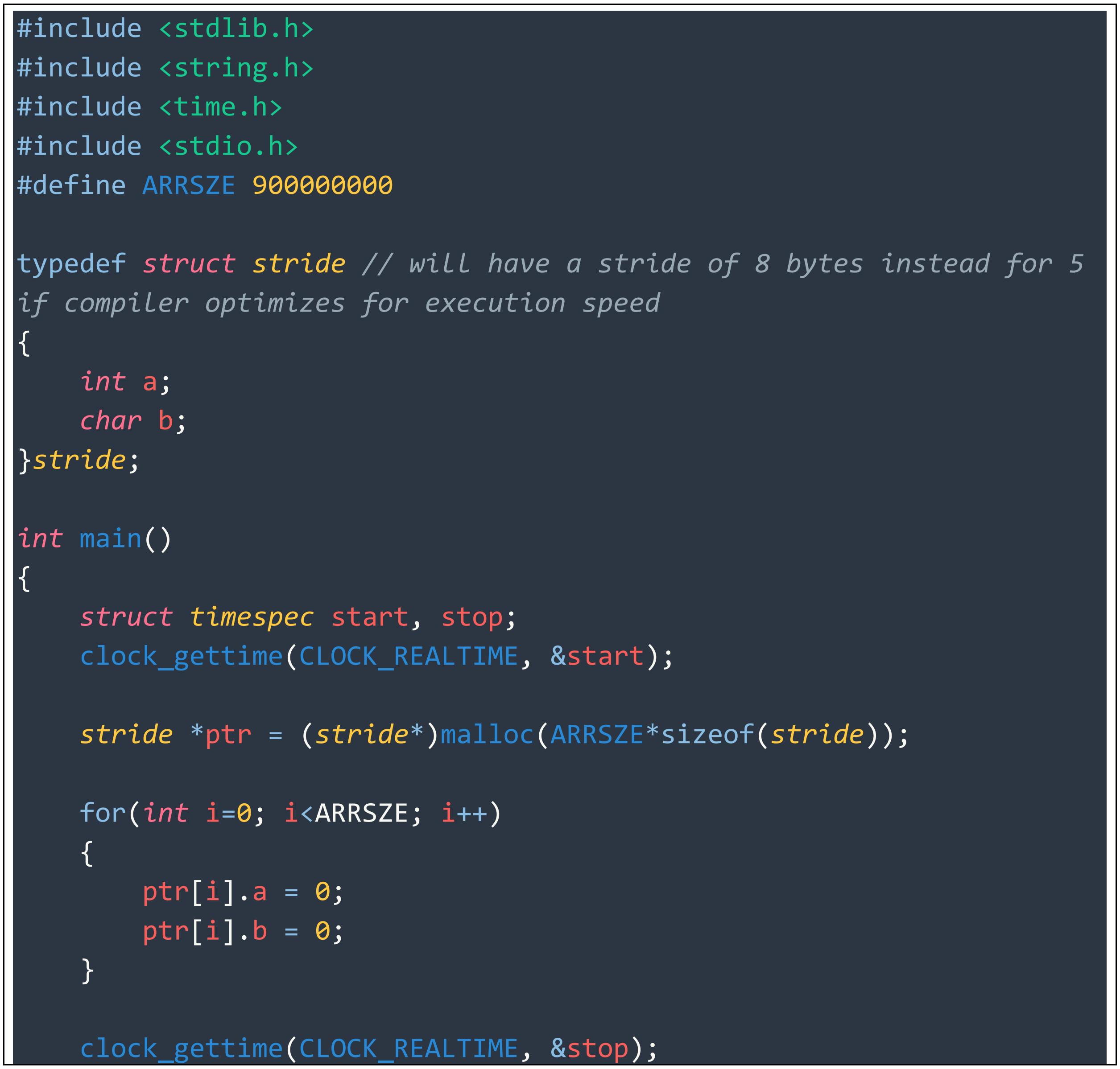
```
┌─[austinsbrown@DESKTOP-O0AMQ3N] - [/mnt/c/Users/austi/OneDrive/School
└─[$] ./a.out
Total execution time: 12.172494000
┌─[austinsbrown@DESKTOP-O0AMQ3N] - [/mnt/c/Users/austi/OneDrive/School
└─[$] 
```

**1b.**

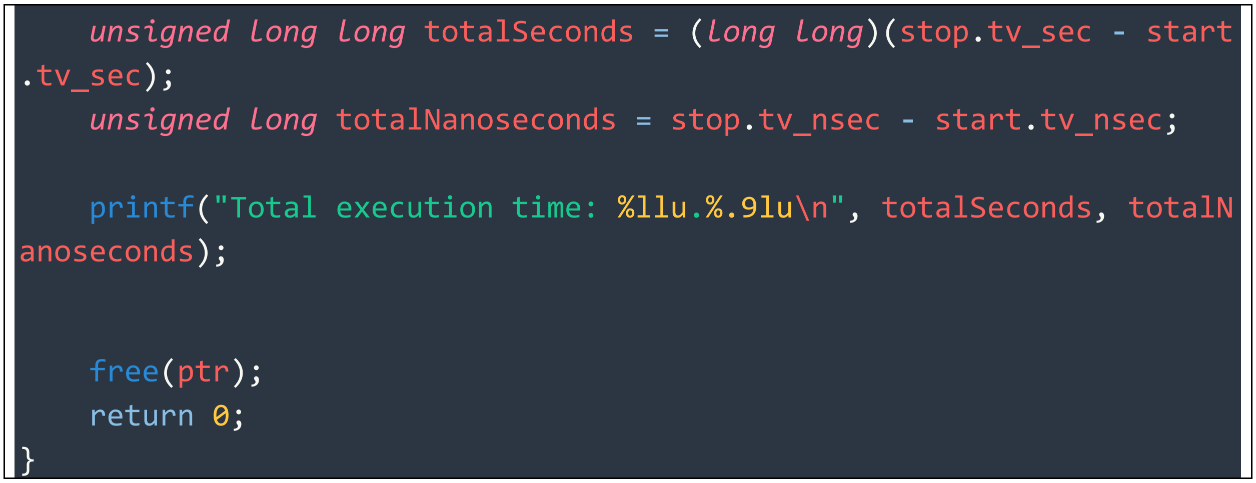


```
Total execution time: 13.12744000
```

**1c.**

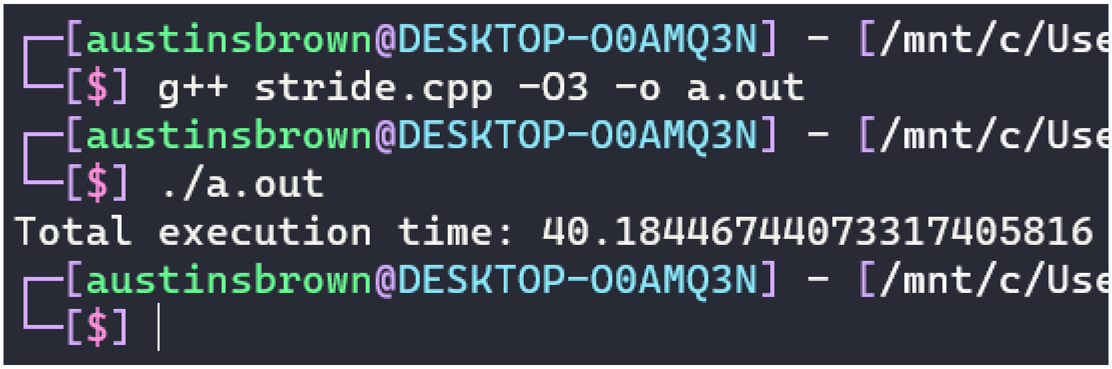


```c
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <stdio.h>
#define ARRSZE 900000000

typedef struct stride // will have a stride of 8 bytes instead for 5
if compiler optimizes for execution speed
{
    int a;
    char b;
}stride;

int main()
{
    struct timespec start, stop;
    clock_gettime(CLOCK_REALTIME, &start);

    stride *ptr = (stride*)malloc(ARRSZE*sizeof(stride));

    for(int i=0; i<ARRSZE; i++)
    {
        ptr[i].a = 0;
        ptr[i].b = 0;
    }

    clock_gettime(CLOCK_REALTIME, &stop);
```



```
    unsigned long long totalSeconds = (long long)(stop.tv_sec - start
.tv_sec);
    unsigned long totalNanoseconds = stop.tv_nsec - start.tv_nsec;

    printf("Total execution time: %llu.%.9lu\n", totalSeconds, totalN
anoseconds);


    free(ptr);
    return 0;
}
```

I used the -03 optimization flag because it can cause the stride struct to take up more space than it is supposed to due to alignment.



```
┌─[austinsbrown@DESKTOP-O0AMQ3N] - [/mnt/c/Use
└─[$] g++ stride.cpp -O3 -o a.out
┌─[austinsbrown@DESKTOP-O0AMQ3N] - [/mnt/c/Use
└─[$] ./a.out
Total execution time: 40.18446744073317405816
┌─[austinsbrown@DESKTOP-O0AMQ3N] - [/mnt/c/Use
└─[$] |
```

| | Virtual Machine (Seconds) | Local Time (Seconds) |
|---|---|---|
| 1 | 2.18446744073272407916 | 1.319018000 |
| 2 | 1.463540000 | 2.18446744073439963416 |
| 3 | 2.18446744073451718116 | 1.395076500 |
| 4 | 2.18446744073393724516 | 1.676363200 |
| 5 | 2.18446744073374848116 | 1.362660400 |
| 6 | 2.590348000 | 1.416361900 |
| 7 | 2.18446744073492724116 | 2.18446744073605025116 |
| 8 | 1.654541200 | 1.439958800 |
| 9 | 2.18446744073328217216 | 1.674886400 |
| 10 | 1.610063400 | 1.404980300 |

The program on executed faster on the local machine for the most part.

**2.**

You could give 4 extra bits to the top level followed by an extra bit for the second and third level. This would be the optimal solution as it is enough space for the program, but little is wasted.