

Take Home Test 3:

OPTIMIZATION OF DOT PRODUCT COMPUTATION OF TWO VECTORS USING VECTOR INSTRUCTIONS

MdShahid Bin Emdad

May 15th, 2022

CSC 34200

I will neither give nor receive
unauthorized assistance on this test.
I will use only one computing device
to perform this TEST, I will not use
cell while performing this test

Shahid

Start Time and date: 9:00 am, 5/13/2022

End Time and date: 8:45 pm. 5/13/2022

TABLE OF CONTENTS

Objective:	2
Description of Specifications, and Functionality:	2
Intel x86 (Visual Studio Code):	2
CPU-Z specifications:.....	2
Dot Product [Non-Optimized]:.....	3
Dot Product [Compiler Generated Optimization]:	6
Dot Product [Manually Optimized]:.....	9
VDPPS Vector Instruction:	10
Intel x64 (Linux g++):.....	12
CPU-X specifications:	12
Dot Product [Non-Optimized]:.....	12
Dot Product [Compiler Generated Optimization]:	13
Simulation:	15
Intel x86 (Visual Studio Code):	15
Dot Product [Non-Optimized]:.....	15
Dot Product [Compiler Generated Optimization]:	25
Dot Product [Manually Optimized]:.....	36
VDPPS:.....	47
Intel x64 (Linux g++):.....	58
Dot Product [Non-Optimized]:.....	58
Dot Product [Compiler Generated Optimization]:	60
Dot Product [Manually Optimized]:.....	61
VDPPS:.....	63
Explanation:	65
Intel x86 (Visual Studio Code):	65
Intel x64 (Linux g++):.....	66
Conclusion:	67

Objective:

The objective of this test is to explore, understand and ability to compute and demonstrate dot products using vector instructions and optimize the code by compiler generate in both Intel x86(MS VS) and Intel x64(Linux) environment. The compilers use parallelization, vectorization to optimize the code. We were also instructed to optimate the code by ourselves for better execution runtime. Eventually, we compared the run time from both environments and check which is the most optimal way to execute the program faster.

Description of Specifications, and Functionality:

The digital system I used in this assignment is Microsoft Visual Studio 2019 to get the average time for each array execution time. After that, I used Ubuntu Linux to run all the c files in the same way and got the average time as well for each array size. I did $2^3(8)$ to $2^{12}(4096)$.

Intel x86 (Visual Studio Code):

CPU-Z specifications:

In figure 1, we can see the CPU-Z window with specifications, the highest instructions I can use the AVX2 for asm file generating. We needed to know this to specify generating the asm file.

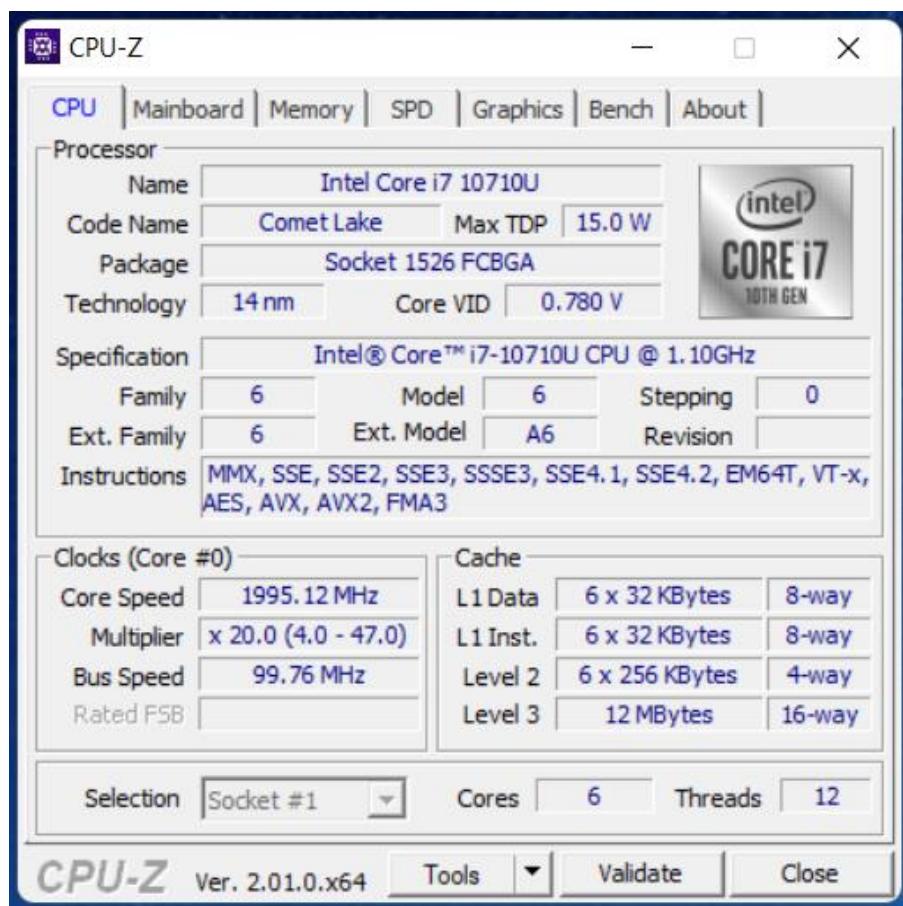


Figure 1: CPU-Z Specs window

In figure 2, we can see the project directory for this project.

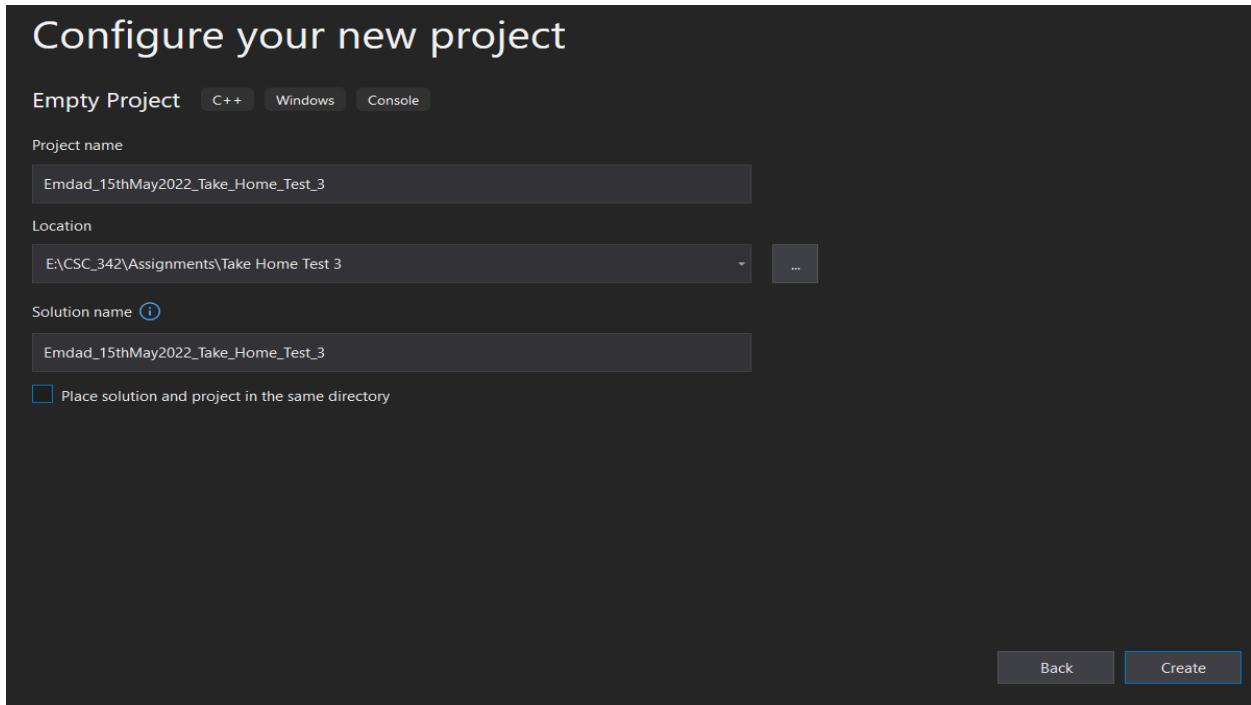


Figure 2: project directory

Dot Product [Non-Optimized]:

In figure 2, we can see the project directory for Emdad_Source.cpp file.

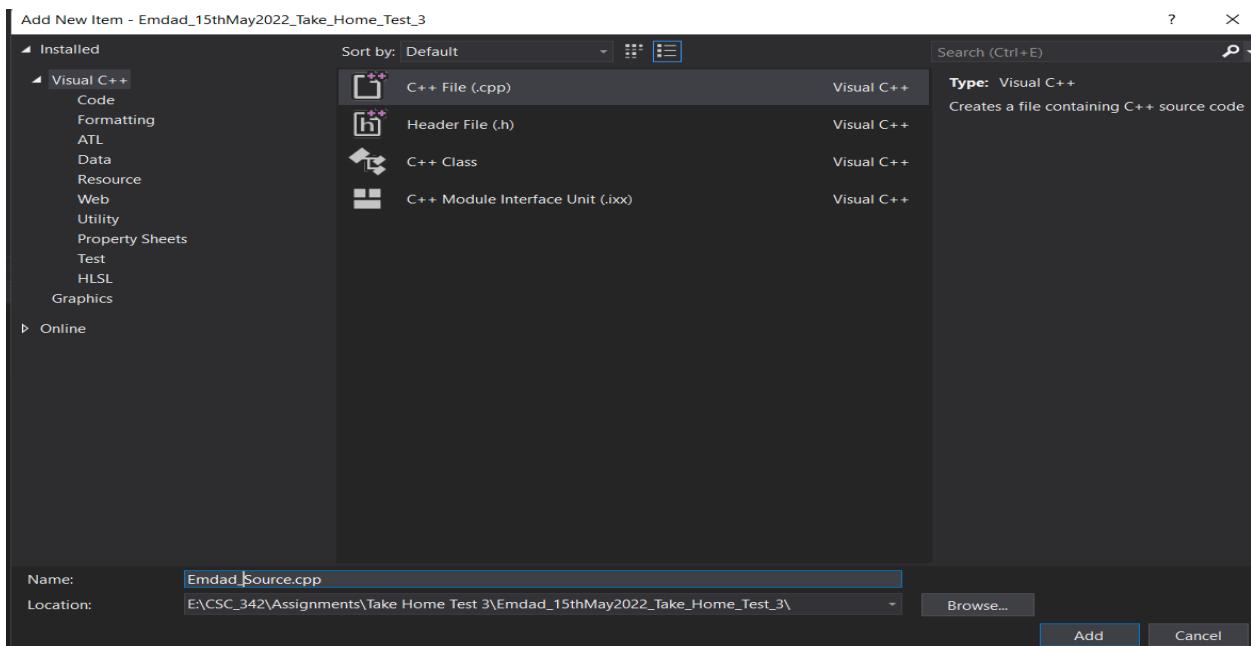
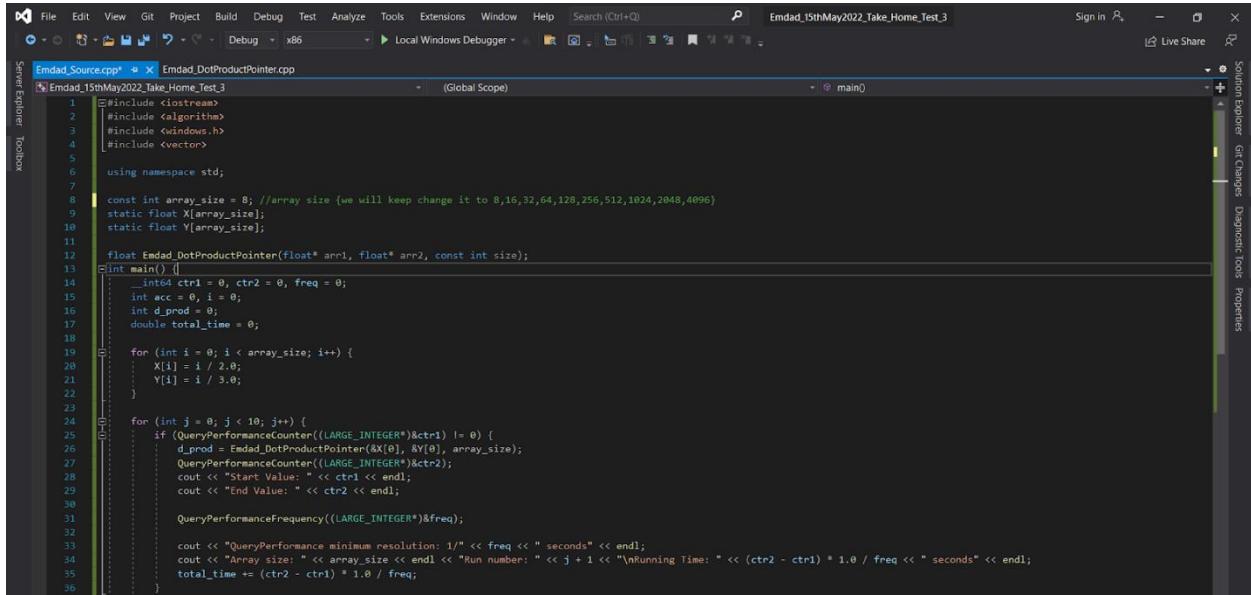


Figure 3: project directory for Emdad_Source.cpp file.

In figure 3, we can see that non optimized c++ code for dot pointer, source code.



```

1 #include <iostream>
2 #include <algorithm>
3 #include <windows.h>
4 #include <vector>
5
6 using namespace std;
7
8 const int array_size = 8; //array size [we will keep change it to 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096]
9 static float X[array_size];
10 static float Y[array_size];
11
12 float Emdad_DotProductPointer(float* arr1, float* arr2, const int size);
13
14 int main() {
15     _int64 ctr1 = 0, ctr2 = 0, freq = 0;
16     int acc = 0, i = 0;
17     int d_prod = 0;
18     double total_time = 0;
19
20     for (int i = 0; i < array_size; i++) {
21         X[i] = i / 2.0;
22         Y[i] = i / 3.0;
23     }
24
25     for (int j = 0; j < 10; j++) {
26         if (QueryPerformanceCounter((LARGE_INTEGER*)&ctr1) != 0) {
27             d_prod = Emdad_DotProductPointer(&X[0], &Y[0], array_size);
28             QueryPerformanceCounter((LARGE_INTEGER*)&ctr2);
29             cout << "Start Value: " << ctr1 << endl;
30             cout << "End Value: " << ctr2 << endl;
31             QueryPerformanceFrequency((LARGE_INTEGER*)&freq);
32
33             cout << "QueryPerformance minimum resolution: " << freq << " seconds" << endl;
34             cout << "Array size: " << array_size << endl << "Run number: " << j + 1 << "\nRunning Time: " << (ctr2 - ctr1) * 1.0 / freq << " seconds" << endl;
35             total_time += (ctr2 - ctr1) * 1.0 / freq;
36         }
37     }
38
39     else {
40         DWORD dwError = GetLastError();
41         cout << "Error value = " << dwError << endl;
42     }
43
44     cout << "Average time: " << (total_time / 10) << endl;
45
46     cout << "-----" << endl;
47
48     system("pause");
49     return 0;
50 }

```

Figure 4: Source code.cpp (Part 1)



```

37
38     else {
39         DWORD dwError = GetLastError();
40         cout << "Error value = " << dwError << endl;
41     }
42
43
44     cout << "Average time: " << (total_time / 10) << endl;
45
46     cout << "-----" << endl;
47
48     system("pause");
49     return 0;
50 }

```

Figure 5: Source code.cpp (Part 2)

I used array pointer arithmetic dot production calculation to get the runtime for each run and different array sizes. I manually changed the array size starting from 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096. I also have an average time calculator to find the most accurate time to take from the 10 different runtimes. I used “**QueryPerformanceCounter**” for the runtime calculation.

In figure 6, we can see the project directory for Emdad_DotProductPointer.cpp file.

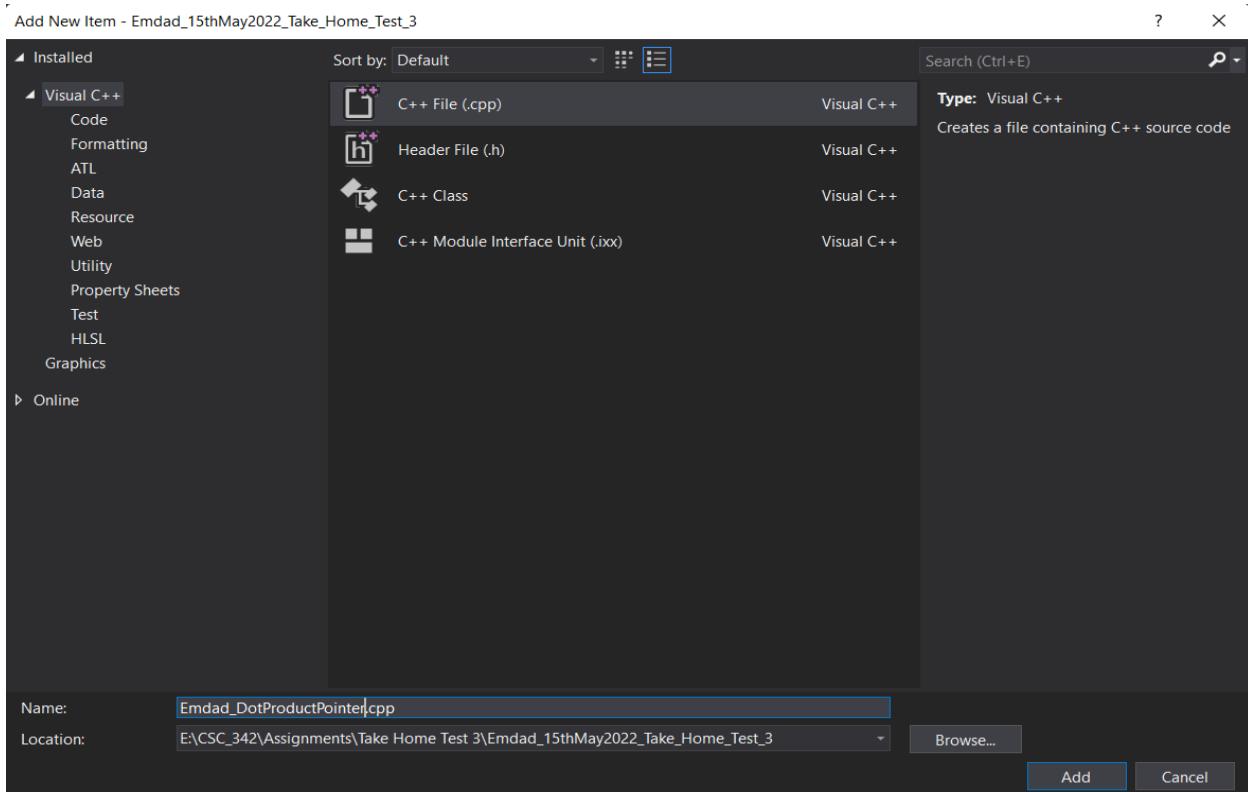


Figure 6: project directory for Emdad_DotProductPointer.cpp

In figure 6, we can see that non optimized c++ code for dot pointer constructor.

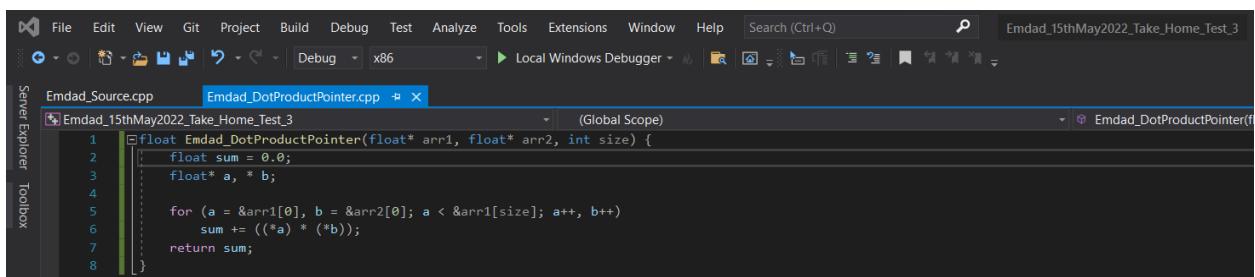


Figure 7: Emdad_DotProductPointer.cpp

Pointer's arithmetic is faster than indexing because indices require multiplication of 4 for each increment which can cause more runtime. It also skips the multiplication step.

In figure 8, we can see the programs compiled without any issues.

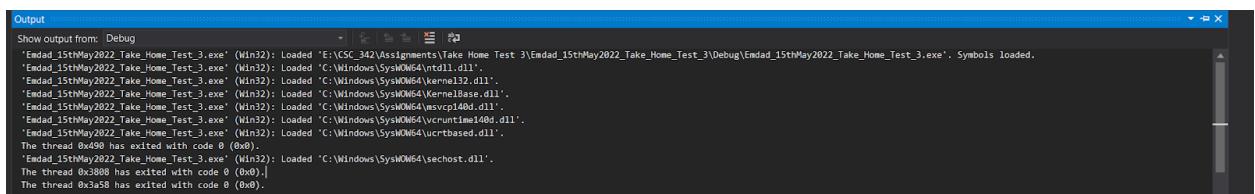


Figure 8: Program compiled

Dot Product [Compiler Generated Optimization]:

In figure 9 and 10, we can see that the file setup to generate asm file from the dotpointer c++ file.

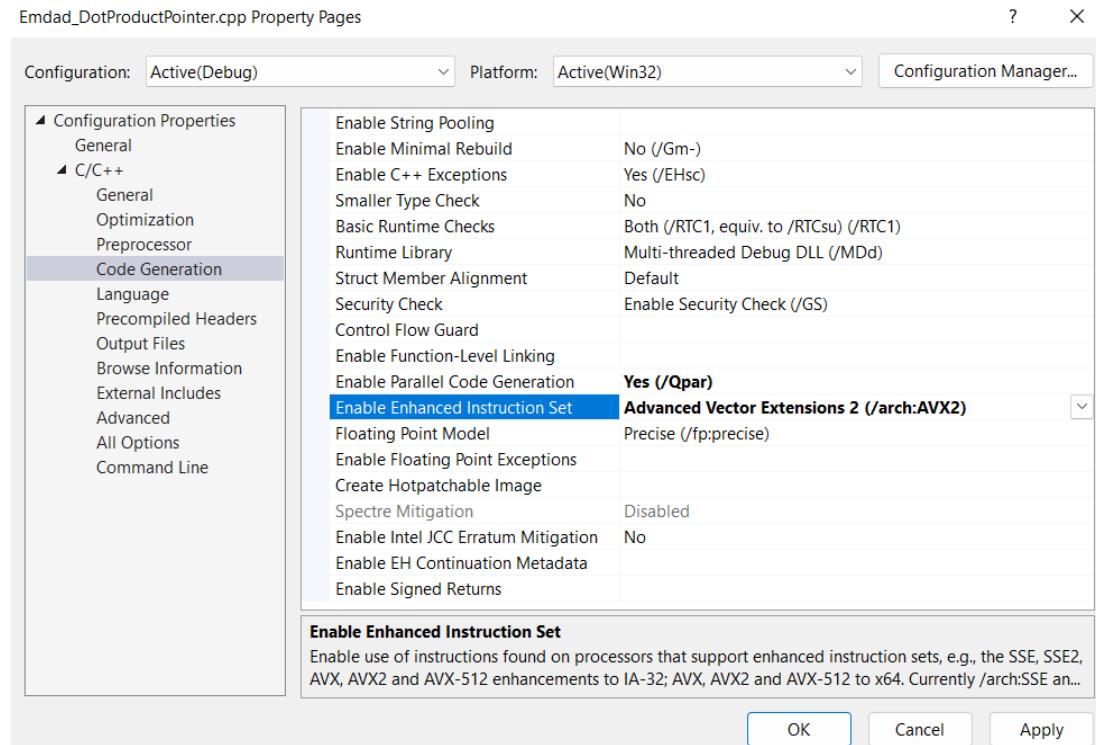


Figure 9: Code Generation Setup

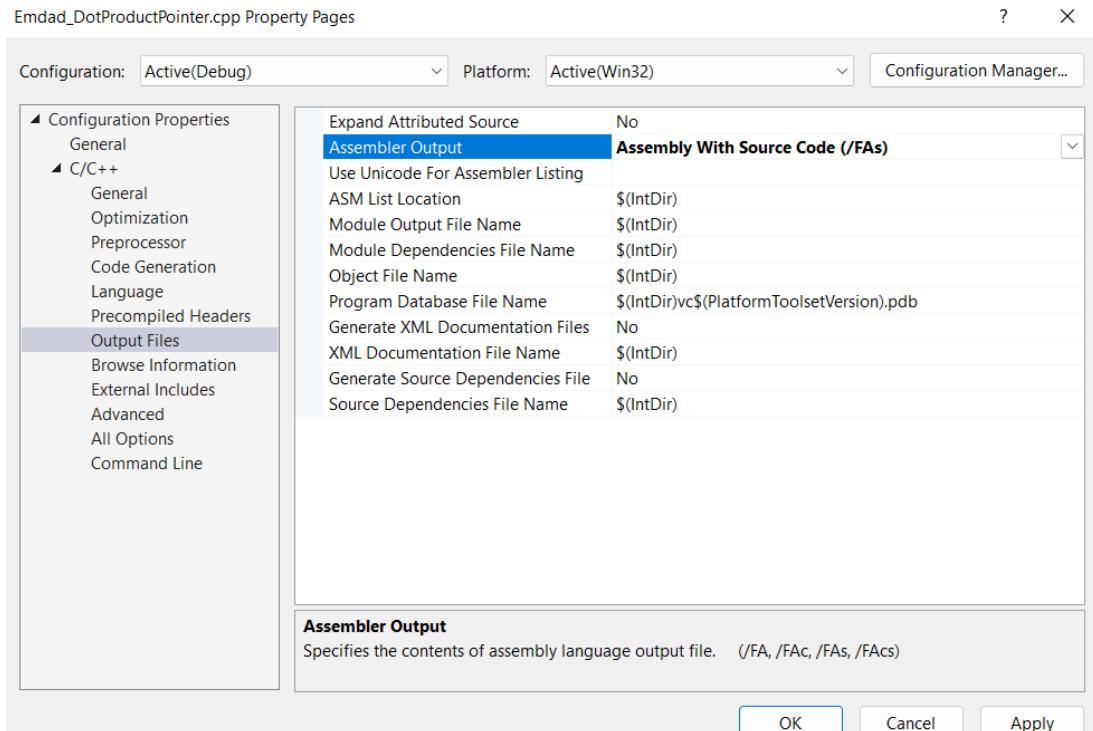


Figure 10: Output files extension

In figure 11, we can see the generated asm file from the source code by the program.

```

1 ; Listing generated by Microsoft (R) Optimizing Compiler Version 19.29.30142.1
2
3 TITLE E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_DotProductPointer.obj
4 .686P
5 .XMM
6 include listing.inc
7 .model flat
8
9 INCLUDELIB MSVCRTD
10 INCLUDELIB OLDNAMES
11
12 msvcjmc SEGMENT
13 _FCD7C082_Emadad_DotProductPointer@cpp DB 01H
14 MSVCJMC ENDS
15 PUBLIC ?Emdad_DotProductPointer@@YAMPAM0H@Z ; Emdad_DotProductPointer
16 EXTRN _JustMyCode_Default ; JustMyCode_Default
17 EXTRN __RTC_CheckEsp@4:PROC
18 EXTRN __RTC_InitBase@PROC
19 EXTRN __RTC_Shutdown@PROC
20 EXTRN __fusedWORD
21 EXTRN __COMDAT rtc$TMZ
22 ; COMDAT rtc$IMZ
23 rtc$TMZ SEGMENT
24 __RTC_Shutdown rtc$TMZ DD FLAT:_RTC_Shutdown
25 rtc$TMZ ENDS
26 ; COMDAT rtc$IMZ
27 rtc$IMZ SEGMENT
28 __RTC_InitBase rtc$IMZ DD FLAT:_RTC_InitBase
29 rtc$IMZ ENDS
30 ; Function compile flags: /Odt
31 ; COMDAT _JustMyCode_Default
32 _TEXT SEGMENT
33 _JustMyCode_Default PROC ; COMDAT
34 push ebp
35 mov ebp, esp
36 pop ebp
37 ret 0
38 _JustMyCode_Default ENDP
39 _TEXT ENDS
40 ; Function compile flags: /Odt /RTCsu /ZI
41 ; File E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Emdad_DotProductPointer.cpp
42 ; COMDAT ?Emdad_DotProductPointer@@YAMPAM0H@Z

```

Figure 11: Emdad_DotProductPointer.asm (Part 1)

```

43 _TEXT SEGMENT
44 _b$ = -32 ; size = 4
45 _a$ = -20 ; size = 4
46 _sum$ = -8 ; size = 4
47 _arr1$ = 8 ; size = 4
48 _arr2$ = 12 ; size = 4
49 _size$ = 16 ; size = 4
50 ?Emdad_DotProductPointer@@YAMPAM0H@Z PROC ; Emdad_DotProductPointer, COMDAT
51 ; 1 : float Emdad_DotProductPointer(float* arr1, float* arr2, int size) {
52
53     push    ebp
54     mov     ebp, esp
55     sub     esp, 228 ; 000000e4H
56     push    ebx
57     push    esi
58     push    edi
59     lea     edi, DWORD PTR [ebp-36]
60     mov     eax, 9
61     mov     eax, -858993460 ; cccccccH
62     rep     stosd
63     mov     ecx, OFFSET __FCD7CD82_Emdad_DotProductPointer@cpp
64     call    @_CheckForDebuggerJustMyCode@4
65
66 ; 2 : float sum = 0.0;
67
68     vxorps xmm0, xmm0, xmm0
69     vmovss DWORD PTR _sum$[ebp], xmm0
70
71 ; 3 : float* a, * b;
72 ; 4 :
73 ; 5 : for (a = &arr1[0], b = &arr2[0]; a < &arr1[size]; a++, b++)
74
75     mov     eax, 4
76     imul    ecx, eax, 0
77     add     ecx, DWORD PTR _arr1$[ebp]
78     mov     edx, DWORD PTR _a$[ebp], ecx
79     mov     edx, 4
80     imul    eax, edx, 0
81     add     eax, DWORD PTR _arr2$[ebp]
82     mov     DWORD PTR _b$[ebp], eax
83     jmp     SHORT $LN4@Emdad_DotP
84

```

Figure 12: Emdad_DotProductPointer.asm (Part 2)

```

85     $LN2@Emdad_DotP:
86     mov     eax, DWORD PTR _a$[ebp]
87     add     eax, 4
88     mov     DWORD PTR _a$[ebp], eax
89     mov     ecx, DWORD PTR _b$[ebp]
90     add     ecx, 4
91     mov     DWORD PTR _b$[ebp], ecx
92     $LN4@Emdad_DotP:
93     mov     eax, DWORD PTR _size$[ebp]
94     mov     ecx, DWORD PTR _arr1$[ebp]
95     lea     edx, DWORD PTR [ecx+eax*4]
96     cmp     DWORD PTR _a$[ebp], edx
97     jae    SHORT $LN2@Emdad_DotP
98
99 ; 6 : sum += (*a) * (*b);
100
101    mov     eax, DWORD PTR _a$[ebp]
102    mov     ecx, DWORD PTR _b$[ebp]
103    vmovss xmm0, DWORD PTR [eax]
104    vmulss xmm0, xmm0, DWORD PTR [ecx]
105    vmoveps xmm0, DWORD PTR [sum$[ebp]]
106    vaddss xmm0, xmm0, xmm0
107    vmoveps DWORD PTR _sum$[ebp], xmm0
108    jmp     SHORT $LN2@Emdad_DotP
109
110     $LN3@Emdad_DotP:
111
112 ; 7 : return sum;
113
114     fild   DWORD PTR _sum$[ebp]
115
116 ; 8 : }
117
118     pop     edi
119     pop     esi
120     pop     ebx
121     add     esp, 228 ; 000000e4H
122     cmp     esp, esp
123     call    _RTC_CheckEsp
124     mov     esp, ebp
125     pop     ebp
126     ret     0
127 ?Emdad_DotProductPointer@@YAMPAM0H@Z ENDP ; Emdad_DotProductPointer

```

Figure 13: Emdad_DotProductPointer.asm (Part 3)

```

127 _TEXT ENDS
128 END

```

Figure 14: Emdad_DotProductPointer.asm (Part 4)

In figure 15, we can see the programs compiled without any issues.

```

100% ▶ 0 issues found
Output ▶
Show output from: Debug
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe'. Symbols loaded.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\kernel32.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\kernelbase.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\msvcp140d.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\msvcrtd.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\user32.dll'.
The thread 0x2374 has exited with code 0 (0x0).
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\sechost.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\kernel32.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SySO64\msvcrtd.dll'.
The thread 0x1c08 has exited with code -1073741510 (0xc000013a).
The thread 0x3878 has exited with code -1073741510 (0xc000013a).
The thread 0x4cc0 has exited with code -1073741510 (0xc000013a).
The program '[17896] Emdad_15thMay2022_Take_Home_Test_3.exe' has exited with code -1073741510 (0xc000013a).

```

Figure 15: program compilation

Dot Product [Manually Optimized]:

In figure 16, we can see the optimized code for the Emdad_DotProductPointer.asm file. In this code, I removed some of the unnecessary lines which was costing more run time. I removed some repeated lines and moved the code out of loop for less runtime. I used parallelization and vectorization for this.

```

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Emdad_15thMay2022_Take_Home_Test_3 Sign in Live Share
Server Explorer Solution Explorer Git Changes Diagnostic Tools Properties
Emdad_DotProductPointer.asm Emdad_Source.cpp Emdad_DotProductPointer.cpp
1 ; Listing generated by Microsoft (R) Optimizing Compiler Version 19.29.30142.1
2
3 TITLE E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_DotProductPointer.obj
4 .686P
5 .XMM
6 include listing.inc
7 .model flat
8
9 INCLUDELIB MSVCRTD
10 INCLUDELIB OLDNAMES
11 PUBLIC ?Emdad_DotProductPointer@@YAMPAM@Z ; Emdad_DotProductPointer
12 ; Function compile flags: /Otp /RTCsu /ZI
13 ; COMDAT ?Emdad_DotProductPointer@@YAMPAM@Z
14
15 _TEXT SEGMENT
16 _b5 = -32 ; size = 4
17 _a5 = -20 ; size = 4
18 _sum$ = -8 ; size = 4
19 _arr1$ = 8 ; size = 4
20 _arr2$ = 12 ; size = 4
21 _size$ = 16 ; size = 4
22
23 ?Emdad_DotProductPointer@@YAMPAM@Z PROC ; Emdad_DotProductPointer, COMDAT
24
25 ; 1 : float Emdad_DotProductPointer(float* arr1, float* arr2, int size) {
26
27     push ebp
28     mov ebp, esp
29     sub esp, 228 ; 000000e4H
30     push ebx
31     push esi
32     push edi
33
34 ; 2 : float sum = 0.0;
35 ; 3 :     vMOVSS DWORD PTR _sum$[ebp], xmm0
36 ; 4 :     vMULSS float* a, * b;
37 ; 5 :     for (a = &arr1[0], b = &arr2[0]; a < &arr1[size]; a++, b++)
38
39
40
41
42

```

Figure 16: Emdad_DotProductPointer.asm (Part 1)

```

43     mov eax, 4
44     imul edx, eax, 0
45     add ecx, DWORD PTR _arr1$[ebp]
46     mov DWORD PTR _arr2$[ebp], ecx
47     mov edx, 4
48     imul eax, edx, 0
49     add eax, DWORD PTR _arr2$[ebp]
50     mov DWORD PTR _b$[ebp], eax
51     jmp SHORT $LN4@Emdad_Dotp
52 $LN2@Emdad_Dotp:
53     mov eax, DWORD PTR _a$[ebp]
54     add eax, 4
55     mov DWORD PTR _a$[ebp], eax
56     mov ecx, DWORD PTR _b$[ebp]
57     add ecx, 4
58     mov DWORD PTR _b$[ebp], ecx
59 $LN4@Emdad_Dotp:
60     mov eax, DWORD PTR _size$[ebp]
61     mov ecx, DWORD PTR _arr1$[ebp]
62     lea edx, DWORD PTR [ecx+eax*4]
63     cmp DWORD PTR _a$[ebp], edx
64     jae SHORT $LN3@Emdad_Dotp
65
66 ; 6 :      sum += ((*) * (*b));
67
68     mov eax, DWORD PTR _a$[ebp]
69     mov ecx, DWORD PTR _b$[ebp]
70     vmovss xmm0, DWORD PTR [eax]
71     vmulss xmm0, xmm0, DWORD PTR [ecx]
72     vmovss xmm1, DWORD PTR _sum$[ebp]
73     vaddss xmm0, xmm1, xmm0
74     vmovss DWORD PTR _sum$[ebp], xmm0
75     jmp SHORT $LN2@Emdad_Dotp
76 $LN3@Emdad_Dotp:
77
78 ; 7 :      return sum;
79
80     fld DWORD PTR _sum$[ebp]
81
82 ; 8 : }
83
84     pop edi

```

Figure 17: Emdad_DotProductPointer.asm (Part 2)

```

85     pop esi
86     pop ebx
87     add esp, 228          ; 000000e4H
88     cmp ebp, esp
89     call _RTC_CheckEsp
90     mov esp, ebp
91     pop ebp
92     ret 0
93 ?Emdad_DotProductPointer@@YAMPAM0@Z ENDP      ; Emdad_DotProductPointer
94 _TEXT ENDS
95 END

```

100 % □ Ready ✓ No issues found

Figure 18: Emdad_DotProductPointer.asm (Part 3)

VDPPS Vector Instruction:

In figure 19, we can see the code for VDPPS cpp file.

```

Emdad_VDPPS_DotProductPointer.cpp  Emdad_DotProductPointer.asm  Emdad_Source.cpp  Emda
Emdad_15thMay2022_Take_Home_Test_3  (Global Scope)

1 float Emdad_VDPPS_DotProductPointer(float* a, float* b, int i) {
2     float x = 0.0;
3     _asm {
4         vxorps ymm3, ymm3, ymm3
5         mov eax, dword ptr[a]
6         mov ebx, dword ptr[b]
7         mov ecx, i
8
9         $mainloop:
10        vmovups ymm0, [eax]
11        vmovups ymm1, [ebx]
12        vdpps ymm2, ymm0, ymm1, 0xFF
13        vaddps ymm3, ymm2, ymm3
14
15        add eax, 32
16        add ebx, 32
17        sub ecx, 8
18        jnz $mainloop
19
20        vperm2f128 ymm0, ymm3, ymm3, 1
21        vaddps ymm3, ymm0, ymm3
22
23        vextractI128 xmm3, ymm3, 1
24        movss dword ptr[x], xmm3
25    }
26    return x;
27 }

```

Figure 19: Emdad_VDPPS_DotProductPointer.cpp

I used DPPS vector instruction to optimize the dot product pointer code further. DPPS do not require any unnecessary addition or multiplication to execute the programs in multiple runs. DPPS does all the operations at once and give the output at the end.

In figure 20, we can see the programs compiled without any issues.

```

Output
Show output from: Debug

'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe'. Symbols loaded.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\ntdll.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\kernel32.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\kernelBase.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\msvcpi40d.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\vcruntime140d.dll'.
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\ucrtbased.dll'.
The thread 0x34e4 has exited with code 0 (0x0).
'Emdad_15thMay2022_Take_Home_Test_3.exe' (Win32): Loaded 'C:\Windows\SysWOW64\sehost.dll'.

```

Figure 20: Program Compilation

Intel x64 (Linux g++):**CPU-X specifications:**

In figure 21, we can see the CPU-X window with specifications, the highest instructions I can use the AVX2 for asm file generating.

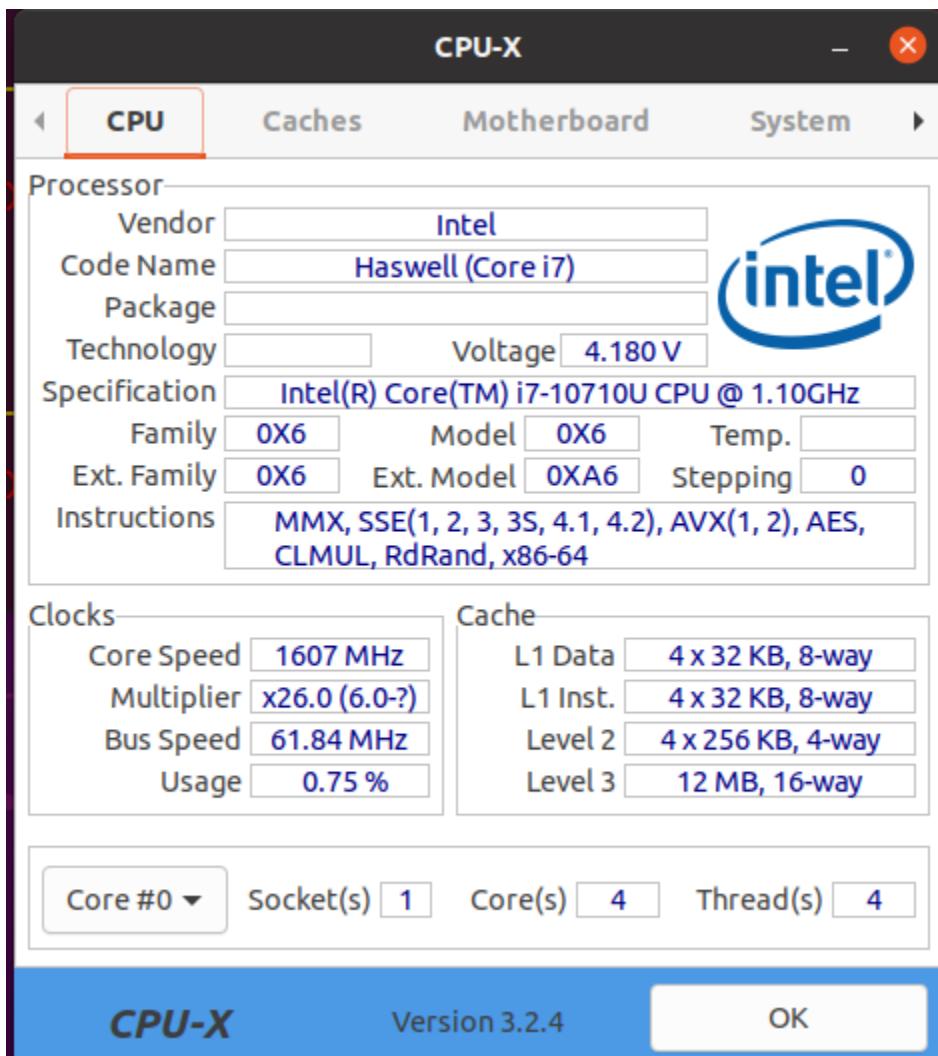


Figure 21: CPU-X Specs window.

I am running Ubuntu in VirtualBox which has different specifications. Therefore, I wanted to have a check by using CPU-X to see if I can use the same instruction in my Linux computer as well.

Dot Product [Non-Optimized]:

In figure 22, we can see the code for Emdad_Source.cpp file.

```

Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Text Editor
Documents Open Emdad_Source.cpp
Emdad_Source.cpp
Emdad_DotProductPointer.cpp
May 15 00:16 •
Emdad_Source.cpp
~/Desktop/cp-543/takeHomeTest3
Save
Emdad_DotProductPointer.cpp
C++ Tab Width: 8 Ln 17, Col 33 INS
1 #include <iostream>
2 #include <time.h>
3 #include <stdint.h>
4 #include <stdlib.h>
5
6 #define NANO 1000000000LL
7
8 using namespace std;
9
10 const int array_size = 8; //array size (we will keep change it to 8,16,32,64,128,256,512,1024,2048,4096)
11 static float *X{array_size};
12 static float *Y{array_size};
13
14 float Emdad_DotProductPointer(float* arr1, float* arr2, const int size);
15
16 int main()
17 {
18     uint64_t elapsedTime;
19     uint64_t total_time = 0;
20     int d_prod;
21     struct timespec startTime, endTime;
22
23     for (int i = 0; i < array_size; i++) {
24         X[i] = i / 2.0;
25         Y[i] = i / 5.0;
26     }
27
28     cout << "Array size: " << array_size << endl;
29
30     for (int n = 0; n < 10; n++) {
31         clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &startTime);
32         d_prod = Emdad_DotProductPointer(&X[0], &Y[0], array_size);
33         clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &endTime);
34
35         elapsedTime = NANO * (endTime.tv_sec - startTime.tv_sec) + (endTime.tv_nsec - startTime.tv_nsec);
36
37         total_time += elapsedTime;
38     }
39
40     cout << "Average time: " << (total_time / 10) << " ns" << endl;
41
42     return 0;
43 }

```

Figure 22: Emdad_Source.cpp

In figure 23, we can see the code for Emdad_DotProductPointer.cpp file.

```

Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Text Editor
Documents Open Emdad_Source.cpp
Emdad_Source.cpp
Emdad_DotProductPointer.cpp
May 15 00:16 •
Emdad_DotProductPointer.cpp
~/Desktop/cp-543/takeHomeTest3
Save
Emdad_DotProductPointer.cpp
C++ Tab Width: 8 Ln 17, Col 33 INS
1 float Emdad_DotProductPointer(float* arr1, float* arr2, const int size) {
2     float sum = 0.0;
3     float* a, * b;
4
5     for (a = &arr1[0], b = &arr2[0]; a < &arr1[size]; a++, b++)
6         sum += (*a) * (*b);
7
8     return sum;
9 }

```

Figure 23: Emdad_DotProductPointer.cpp

Dot Product [Compiler Generated Optimization]:

In figure 24, we can see the Emdad_DotProductPointer.s file which generated from the Emdad_Source.cpp file.

The screenshot shows the GDB debugger interface on an Ubuntu system. The assembly code is displayed in the central window, with file names like Emdad_DotProductPointers.s, Emdad_Source.cpp, and Emdad_DotProductPointer.cpp visible at the top. The assembly code includes labels such as .LFB0, .L3, and .L2, and various instructions like movq, pushq, and addq.

```

.LFB0:
    .cf_startproc
    .text
    .globl _Z23Emdad_DotProductPointerPFS_L
    .type _Z23Emdad_DotProductPointerPFS_L, @function
_Z23Emdad_DotProductPointerPFS_L:
.LFBB0:
    .cf_startproc
    .text
    .pushq %rbp
    .cf_offset %rbp, -16
    .movq %rsp, %rbp
    .cf_def_cfa register 6
    .movq %rdi, -40(%rbp)
    .movq %rsi, -48(%rbp)
    .movl %rcx, -52(%rbp)
    .popq %rdx
    .movss %xmm0, -20(%rbp)
    .movq %rax, -40(%rbp), %rax
    .movq %rax, -16(%rbp)
    .movq %rax, -48(%rbp), %rax
    .movq %rax, -8(%rbp)
    .movl -52(%rbp), %eax
    .cltq
    .leaq 0(%rax, 4,%rdx)
    .movq -40(%rbp), %rax
    .addq %rax, %rax
    .addq %rax, -16(%rbp)
    .jnb .L2
    .movq -16(%rbp), %rax
    .movss (%rax), %xmm1
    .movq -8(%rbp), %rax
    .movss (%rax), %xmm0
    .notb %mm1, %mm0
    .movss -20(%rbp), %xmm1
    .addss %xmm1, %xmm0
    .movss %xmm0, -20(%rbp)
    .addq $4, -16(%rbp)
    .addq $4, -8(%rbp)
    .jmp .L3
.L3:
    .movss -26(%rbp), %xmm0
    .popq %rbp
    .cf_endproc
    .cf_endproc

```

Figure 24: Emdad_DotProductPointers.s (Part 1)

This screenshot continues the assembly code from Figure 24. It shows the .LFE80 section, which includes notes about the stack and properties, and ends with a string section containing "GNU". The assembly code uses labels like .L1, .L2, and .L3.

```

.LFE80:
    .size _Z23Emdad_DotProductPointerPFS_L, .-_Z23Emdad_DotProductPointerPFS_L
    .ident "GCC: (Ubuntu 9.4.0-1ubuntu1-20.04.1) 9.4.0"
    .section .note.GNU-stack,"@progbits"
    .section .note.gnu.property,"a"
    .align 8
    .long 1f - 0f
    .long 4f - 1f
    .long 5
.L1:
    .string "GNU"
.L2:
    .align 8
    .long 0xc0000002
.L3:
    .long 3f - 2f
.L4:
    .long 0x3
    .align 8

```

Figure 25: Emdad_DotProductPointers.s (Part 2)

I ran `g++ -S Emdad_DotProductPointer.cpp` to generate the asm file.

Simulation:

Intel x86 (Visual Studio Code):

Dot Product [Non-Optimized]:

In figure 26, we can see the array size average execution running time for array size 8.

```
E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1250924103385
End Value: 1250924103392
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 1
Running Time: 7e-07 seconds
Start Value: 1250924128278
End Value: 1250924128280
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 2
Running Time: 2e-07 seconds
Start Value: 1250924136759
End Value: 1250924136761
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1250924144847
End Value: 1250924144850
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 4
Running Time: 3e-07 seconds
Start Value: 1250924151728
End Value: 1250924151730
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 5
Running Time: 2e-07 seconds
Start Value: 1250924160064
End Value: 1250924160066
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1250924172289
End Value: 1250924172291
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 7
Running Time: 2e-07 seconds
Start Value: 1250924179014
End Value: 1250924179015
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 8
Running Time: 1e-07 seconds
Start Value: 1250924187405
End Value: 1250924187408
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 9
Running Time: 3e-07 seconds
Start Value: 1250924195705
End Value: 1250924195707
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 10
Running Time: 2e-07 seconds
Average time: 2.6e-07
-----
Press any key to continue . . .
```

Figure 26: average execution running time for array size 8.

In figure 27, we can see the array size average execution running time for array size 16.

```
E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1251443052501
End Value: 1251443052507
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 1
Running Time: 6e-07 seconds
Start Value: 1251443079446
End Value: 1251443079449
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 2
Running Time: 3e-07 seconds
Start Value: 1251443087031
End Value: 1251443087033
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1251443094200
End Value: 1251443094202
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1251443101276
End Value: 1251443101277
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 5
Running Time: 1e-07 seconds
Start Value: 1251443108757
End Value: 1251443108759
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1251443116019
End Value: 1251443116023
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1251443124477
End Value: 1251443124480
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 8
Running Time: 3e-07 seconds
Start Value: 1251443131752
End Value: 1251443131754
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 9
Running Time: 2e-07 seconds
Start Value: 1251443138967
End Value: 1251443138969
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 10
Running Time: 2e-07 seconds
Average time: 2.7e-07
-----
Press any key to continue . . .
```

Figure 27: average execution running time for array size 16

In figure 28, we can see the array size average execution running time for array size 32.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1251698423215
End Value: 1251698423222
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 1
Running Time: 7e-07 seconds
Start Value: 1251698439286
End Value: 1251698439290
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 2
Running Time: 4e-07 seconds
Start Value: 1251698448905
End Value: 1251698448908
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1251698459199
End Value: 1251698459204
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 4
Running Time: 5e-07 seconds
Start Value: 1251698472185
End Value: 1251698472189
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 5
Running Time: 4e-07 seconds
Start Value: 1251698482960
End Value: 1251698482963
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1251698493097
End Value: 1251698493101
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1251698503664
End Value: 1251698503667
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 8
Running Time: 3e-07 seconds
Start Value: 1251698511314
End Value: 1251698511318
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 9
Running Time: 4e-07 seconds
Start Value: 1251698519536
End Value: 1251698519539
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 10
Running Time: 3e-07 seconds
Average time: 4e-07
-----
Press any key to continue . . .

```

Figure 28: execution running time for array size 32.

In figure 29, we can see the array size average execution running time for array size 64.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1252694666585
End Value: 1252694666608
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 1
Running Time: 2.3e-06 seconds
Start Value: 1252694701466
End Value: 1252694701470
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 2
Running Time: 4e-07 seconds
Start Value: 1252694711760
End Value: 1252694711763
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1252694722113
End Value: 1252694722117
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 4
Running Time: 4e-07 seconds
Start Value: 1252694735362
End Value: 1252694735366
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 5
Running Time: 4e-07 seconds
Start Value: 1252694745623
End Value: 1252694745626
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1252694753649
End Value: 1252694753653
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1252694762250
End Value: 1252694762255
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 8
Running Time: 5e-07 seconds
Start Value: 1252694770423
End Value: 1252694770426
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 9
Running Time: 3e-07 seconds
Start Value: 1252694778752
End Value: 1252694778756
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 10
Running Time: 4e-07 seconds
Average time: 5.7e-07
-----
Press any key to continue . . .

```

Figure 29: execution running time for array size 64

In figure 30, we can see the array size average execution running time for array size 128.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1253376993618
End Value: 1253376993633
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 1
Running Time: 1.5e-06 seconds
Start Value: 1253377027598
End Value: 1253377027606
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 2
Running Time: 8e-07 seconds
Start Value: 1253377035564
End Value: 1253377035569
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 3
Running Time: 5e-07 seconds
Start Value: 1253377043474
End Value: 1253377043481
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 4
Running Time: 7e-07 seconds
Start Value: 1253377052461
End Value: 1253377052467
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 5
Running Time: 6e-07 seconds
Start Value: 1253377061006
End Value: 1253377061012
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 6
Running Time: 6e-07 seconds
Start Value: 1253377068402
End Value: 1253377068408
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 7
Running Time: 6e-07 seconds
Start Value: 1253377075465
End Value: 1253377075471
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 8
Running Time: 6e-07 seconds
Start Value: 1253377083874
End Value: 1253377083879
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 9
Running Time: 5e-07 seconds
Start Value: 1253377091950
End Value: 1253377091955
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 10
Running Time: 5e-07 seconds
Average time: 6.9e-07
-----
Press any key to continue . . .

```

Figure 30: execution running time for array size 128.

In figure 31, we can see the array size average execution running time for array size 256.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1253643508003
End Value: 1253643508017
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 1
Running Time: 1.4e-06 seconds
Start Value: 1253643535663
End Value: 1253643535673
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 2
Running Time: 1e-06 seconds
Start Value: 1253643543474
End Value: 1253643543483
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 3
Running Time: 9e-07 seconds
Start Value: 1253643550845
End Value: 1253643550856
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 4
Running Time: 1.1e-06 seconds
Start Value: 1253643558920
End Value: 1253643558929
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 5
Running Time: 9e-07 seconds
Start Value: 1253643566150
End Value: 1253643566160
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 6
Running Time: 1e-06 seconds
Start Value: 1253643573372
End Value: 1253643573381
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 7
Running Time: 9e-07 seconds
Start Value: 1253643580577
End Value: 1253643580585
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 8
Running Time: 8e-07 seconds
Start Value: 1253643589062
End Value: 1253643589071
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 9
Running Time: 9e-07 seconds
Start Value: 1253643597526
End Value: 1253643597535
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 10
Running Time: 9e-07 seconds
Average time: 9.8e-07
-----
Press any key to continue . . .

```

Figure 31: execution running time for array size 256.

In figure 32, we can see the array size average execution running time for array size 512.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1254001027224
End Value: 1254001027251
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 1
Running Time: 2.7e-06 seconds
Start Value: 1254001052627
End Value: 1254001052648
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 2
Running Time: 2.1e-06 seconds
Start Value: 1254001062897
End Value: 1254001062917
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 3
Running Time: 2e-06 seconds
Start Value: 1254001071315
End Value: 1254001071335
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 4
Running Time: 2e-06 seconds
Start Value: 1254001079965
End Value: 1254001079984
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 5
Running Time: 1.9e-06 seconds
Start Value: 1254001090437
End Value: 1254001090454
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 6
Running Time: 1.7e-06 seconds
Start Value: 1254001102133
End Value: 1254001102150
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 7
Running Time: 1.7e-06 seconds
Start Value: 1254001110835
End Value: 1254001110862
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 8
Running Time: 2.7e-06 seconds
Start Value: 1254001119163
End Value: 1254001119190
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 9
Running Time: 2.7e-06 seconds
Start Value: 1254001129579
End Value: 1254001129620
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 10
Running Time: 4.1e-06 seconds
Average time: 2.36e-06
-----
Press any key to continue . . .

```

Figure 32: execution running time for array size 512.

In figure 33, we can see the array size average execution running time for array size 1024.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1254222077348
End Value: 1254222077385
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 1
Running Time: 3.7e-06 seconds
Start Value: 1254222098568
End Value: 1254222098601
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 2
Running Time: 3.3e-06 seconds
Start Value: 1254222113236
End Value: 1254222113269
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 3
Running Time: 3.3e-06 seconds
Start Value: 1254222121828
End Value: 1254222121861
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 4
Running Time: 3.3e-06 seconds
Start Value: 1254222130777
End Value: 1254222130812
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 5
Running Time: 3.5e-06 seconds
Start Value: 1254222139204
End Value: 1254222139239
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 6
Running Time: 3.5e-06 seconds
Start Value: 1254222149313
End Value: 1254222149350
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 7
Running Time: 3.7e-06 seconds
Start Value: 1254222157996
End Value: 1254222158030
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 8
Running Time: 3.4e-06 seconds
Start Value: 1254222166758
End Value: 1254222166791
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 9
Running Time: 3.3e-06 seconds
Start Value: 1254222174682
End Value: 1254222174715
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 10
Running Time: 3.3e-06 seconds
Average time: 3.43e-06
-----
Press any key to continue . . .

```

Figure 33: execution running time for array size 1024.

In figure 34, we can see the array size average execution running time for array size 2048.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1254576388484
End Value: 1254576388578
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 1
Running Time: 9.4e-06 seconds
Start Value: 1254576413709
End Value: 1254576413795
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 2
Running Time: 8.6e-06 seconds
Start Value: 1254576424858
End Value: 1254576424941
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 3
Running Time: 8.3e-06 seconds
Start Value: 1254576434645
End Value: 1254576434723
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 4
Running Time: 7.8e-06 seconds
Start Value: 1254576443761
End Value: 1254576443836
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 5
Running Time: 7.5e-06 seconds
Start Value: 1254576452958
End Value: 1254576453031
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 6
Running Time: 7.3e-06 seconds
Start Value: 1254576462479
End Value: 1254576462549
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 7
Running Time: 7e-06 seconds
Start Value: 1254576470959
End Value: 1254576471029
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 8
Running Time: 7e-06 seconds
Start Value: 1254576480010
End Value: 1254576480076
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 9
Running Time: 6.6e-06 seconds
Start Value: 1254576487879
End Value: 1254576487942
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 10
Running Time: 6.3e-06 seconds
Average time: 7.58e-06
-----
Press any key to continue . . .

```

Figure 34: execution running time for array size 2048.

In figure 35, we can see the array size average execution running time for array size 4096.

```

Select E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1254815221361
End Value: 1254815221841
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 1
Running Time: 4.8e-05 seconds
Start Value: 1254815258503
End Value: 1254815258629
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 2
Running Time: 1.26e-05 seconds
Start Value: 1254815266713
End Value: 1254815266837
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 3
Running Time: 1.24e-05 seconds
Start Value: 1254815274054
End Value: 1254815274173
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 4
Running Time: 1.19e-05 seconds
Start Value: 1254815282278
End Value: 1254815282397
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 5
Running Time: 1.19e-05 seconds
Start Value: 1254815289893
End Value: 1254815290012
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 6
Running Time: 1.19e-05 seconds
Start Value: 1254815299672
End Value: 1254815299799
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 7
Running Time: 1.27e-05 seconds
Start Value: 1254815307873
End Value: 1254815307996
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 8
Running Time: 1.23e-05 seconds
Start Value: 1254815316218
End Value: 1254815316349
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 9
Running Time: 1.31e-05 seconds
Start Value: 1254815324526
End Value: 1254815324655
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 10
Running Time: 1.29e-05 seconds
Average time: 1.597e-05
-----
Press any key to continue . . .

```

Figure 35: execution running time for array size 4096.

In figure 36, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (seconds)
8	2.60E-07
16	2.70E-07
32	4.00E-07
64	5.70E-07
128	6.90E-07
256	9.80E-07
512	2.36E-06
1024	3.43E-06
2048	7.58E-06
4096	1.60E-05

Figure 36: Average Runtime Table(Non-Optimized)

In figure 37, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

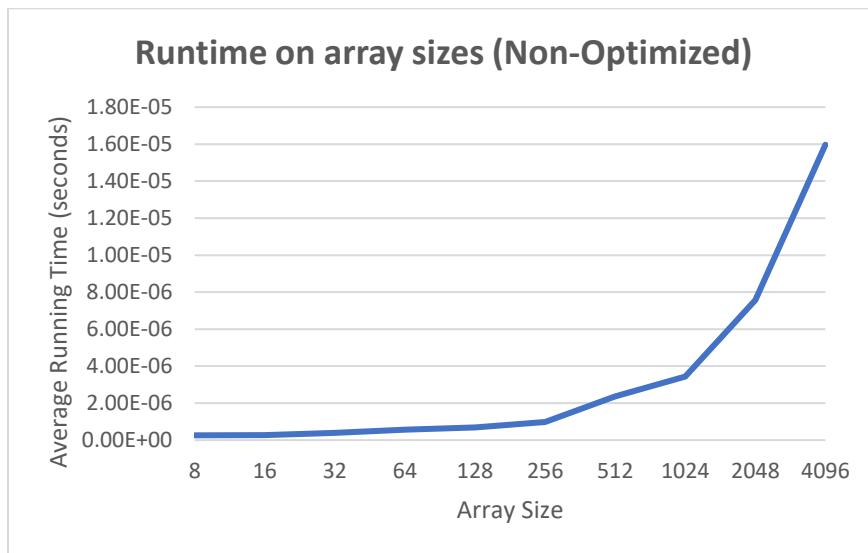


Figure 37: Runtime graph(non-Optimized)

Dot Product [Compiler Generated Optimization]:

In figure 38, we can see the array size average execution running time for array size 8.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1272130125135
End Value: 1272130125143
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 1
Running Time: 8e-07 seconds
Start Value: 1272130144775
End Value: 1272130144779
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 2
Running Time: 4e-07 seconds
Start Value: 1272130159589
End Value: 1272130159593
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 3
Running Time: 4e-07 seconds
Start Value: 1272130174887
End Value: 1272130174892
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 4
Running Time: 5e-07 seconds
Start Value: 1272130187492
End Value: 1272130187494
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 5
Running Time: 2e-07 seconds
Start Value: 1272130200503
End Value: 1272130200506
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1272130214180
End Value: 1272130214184
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1272130231921
End Value: 1272130231927
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 8
Running Time: 6e-07 seconds
Start Value: 1272130253746
End Value: 1272130253751
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 9
Running Time: 5e-07 seconds
Start Value: 1272130313077
End Value: 1272130313081
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 10
Running Time: 4e-07 seconds
Average time: 4.5e-07
-----
Press any key to continue . . .

```

Figure 38: average execution running time for array size 8.

In figure 39, we can see the array size average execution running time for array size 16.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1273331021116
End Value: 1273331021125
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 1
Running Time: 9e-07 seconds
Start Value: 1273331050964
End Value: 1273331050969
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 2
Running Time: 5e-07 seconds
Start Value: 1273331074898
End Value: 1273331074902
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 3
Running Time: 4e-07 seconds
Start Value: 1273331092307
End Value: 1273331092310
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 4
Running Time: 3e-07 seconds
Start Value: 1273331107140
End Value: 1273331107143
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 5
Running Time: 3e-07 seconds
Start Value: 1273331121121
End Value: 1273331121124
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1273331136341
End Value: 1273331136345
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1273331153044
End Value: 1273331153049
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 8
Running Time: 5e-07 seconds
Start Value: 1273331213359
End Value: 1273331213366
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 9
Running Time: 7e-07 seconds
Start Value: 1273331238620
End Value: 1273331238627
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 10
Running Time: 7e-07 seconds
Average time: 5e-07
-----
Press any key to continue . . .

```

Figure 39: execution running time for array size 16.

In figure 40, we can see the array size average execution running time for array size 32.

```

Select E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1273864379004
End Value: 1273864379009
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 1
Running Time: 5e-07 seconds
Start Value: 1273864396812
End Value: 1273864396816
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 2
Running Time: 4e-07 seconds
Start Value: 1273864409754
End Value: 1273864409756
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1273864421609
End Value: 1273864421613
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 4
Running Time: 4e-07 seconds
Start Value: 1273864440716
End Value: 1273864440720
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 5
Running Time: 4e-07 seconds
Start Value: 1273864454247
End Value: 1273864454251
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 6
Running Time: 4e-07 seconds
Start Value: 1273864466200
End Value: 1273864466203
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 7
Running Time: 3e-07 seconds
Start Value: 1273864482422
End Value: 1273864482428
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 8
Running Time: 6e-07 seconds
Start Value: 1273864507224
End Value: 1273864507230
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 9
Running Time: 6e-07 seconds
Start Value: 1273864531115
End Value: 1273864531126
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 10
Running Time: 1.1e-06 seconds
Average time: 4.9e-07
-----
Press any key to continue . . .

```

Figure 40: execution running time for array size 32

In figure 41, we can see the array size average execution running time for array size 64.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1274811181381
End Value: 1274811181392
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 1
Running Time: 1.1e-06 seconds
Start Value: 1274811227820
End Value: 1274811227832
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 2
Running Time: 1.2e-06 seconds
Start Value: 1274811247668
End Value: 1274811247676
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 3
Running Time: 8e-07 seconds
Start Value: 1274811262055
End Value: 1274811262062
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 4
Running Time: 7e-07 seconds
Start Value: 1274811275382
End Value: 1274811275388
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 5
Running Time: 6e-07 seconds
Start Value: 1274811287734
End Value: 1274811287739
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 6
Running Time: 5e-07 seconds
Start Value: 1274811299453
End Value: 1274811299459
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 7
Running Time: 6e-07 seconds
Start Value: 1274811321338
End Value: 1274811321346
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 8
Running Time: 8e-07 seconds
Start Value: 1274811366439
End Value: 1274811366451
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 9
Running Time: 1.2e-06 seconds
Start Value: 1274811383600
End Value: 1274811383607
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 10
Running Time: 7e-07 seconds
Average time: 8.2e-07
-----
Press any key to continue . . . .

```

Figure 41: execution running time for array size 64

In figure 42, we can see the array size average execution running time for array size 128.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1275592694937
End Value: 1275592694957
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 1
Running Time: 2e-06 seconds
Start Value: 1275592741505
End Value: 1275592741521
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 2
Running Time: 1.6e-06 seconds
Start Value: 1275592793521
End Value: 1275592793537
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 3
Running Time: 1.6e-06 seconds
Start Value: 1275592822637
End Value: 1275592822649
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 4
Running Time: 1.2e-06 seconds
Start Value: 1275592865260
End Value: 1275592865278
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 5
Running Time: 1.8e-06 seconds
Start Value: 1275592899671
End Value: 1275592899685
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 6
Running Time: 1.4e-06 seconds
Start Value: 1275592931330
End Value: 1275592931345
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 7
Running Time: 1.5e-06 seconds
Start Value: 1275592951813
End Value: 1275592951824
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 8
Running Time: 1.1e-06 seconds
Start Value: 1275592972429
End Value: 1275592972440
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 9
Running Time: 1.1e-06 seconds
Start Value: 1275592993384
End Value: 1275592993397
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 10
Running Time: 1.3e-06 seconds
Average time: 1.46e-06
-----
Press any key to continue . . .

```

Figure 42: execution running time for array size 128

In figure 43, we can see the array size average execution running time for array size 256.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1275854813246
End Value: 1275854813259
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 1
Running Time: 1.3e-06 seconds
Start Value: 1275854835764
End Value: 1275854835775
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 2
Running Time: 1.1e-06 seconds
Start Value: 1275854844248
End Value: 1275854844255
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 3
Running Time: 7e-07 seconds
Start Value: 1275854851327
End Value: 1275854851335
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 4
Running Time: 8e-07 seconds
Start Value: 1275854858478
End Value: 1275854858487
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 5
Running Time: 9e-07 seconds
Start Value: 1275854868396
End Value: 1275854868408
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 6
Running Time: 1.2e-06 seconds
Start Value: 1275854884271
End Value: 1275854884282
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 7
Running Time: 1.1e-06 seconds
Start Value: 1275854894875
End Value: 1275854894886
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 8
Running Time: 1.1e-06 seconds
Start Value: 1275854905704
End Value: 1275854905719
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 9
Running Time: 1.5e-06 seconds
Start Value: 1275854920126
End Value: 1275854920142
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 10
Running Time: 1.6e-06 seconds
Average time: 1.13e-06
-----
Press any key to continue . . .

```

Figure 43: execution running time for array size 256

In figure 44, we can see the array size average execution running time for array size 512.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1276168728003
End Value: 1276168728045
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 1
Running Time: 4.2e-06 seconds
Start Value: 1276168741594
End Value: 1276168741617
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 2
Running Time: 2.3e-06 seconds
Start Value: 1276168754143
End Value: 1276168754170
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 3
Running Time: 2.7e-06 seconds
Start Value: 1276168774252
End Value: 1276168774280
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 4
Running Time: 2.8e-06 seconds
Start Value: 1276168786749
End Value: 1276168786772
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 5
Running Time: 2.3e-06 seconds
Start Value: 1276168799041
End Value: 1276168799066
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 6
Running Time: 2.5e-06 seconds
Start Value: 1276168810019
End Value: 1276168810044
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 7
Running Time: 2.5e-06 seconds
Start Value: 1276168822400
End Value: 1276168822423
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 8
Running Time: 2.3e-06 seconds
Start Value: 1276168836807
End Value: 1276168836835
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 9
Running Time: 2.8e-06 seconds
Start Value: 1276168849344
End Value: 1276168849370
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 10
Running Time: 2.6e-06 seconds
Average time: 2.7e-06
-----
Press any key to continue . . .

```

Figure 44: execution running time for array size 512

In figure 45, we can see the array size average execution running time for array size 1024.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1276461858173
End Value: 1276461858220
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 1
Running Time: 4.7e-06 seconds
Start Value: 1276461879519
End Value: 1276461879553
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 2
Running Time: 3.4e-06 seconds
Start Value: 1276461894545
End Value: 1276461894573
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 3
Running Time: 2.8e-06 seconds
Start Value: 1276461902285
End Value: 1276461902314
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 4
Running Time: 2.9e-06 seconds
Start Value: 1276461912272
End Value: 1276461912307
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 5
Running Time: 3.5e-06 seconds
Start Value: 1276461923042
End Value: 1276461923080
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 6
Running Time: 3.8e-06 seconds
Start Value: 1276461937731
End Value: 1276461937772
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 7
Running Time: 4.1e-06 seconds
Start Value: 1276461950904
End Value: 1276461950950
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 8
Running Time: 4.6e-06 seconds
Start Value: 1276461963122
End Value: 1276461963165
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 9
Running Time: 4.3e-06 seconds
Start Value: 1276461976454
End Value: 1276461976499
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 10
Running Time: 4.5e-06 seconds
Average time: 3.86e-06
-----
Press any key to continue . . .

```

Figure 45: execution running time for array size 1024

In figure 46, we can see the array size average execution running time for array size 2048.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1276732986422
End Value: 1276732986495
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 1
Running Time: 7.3e-06 seconds
Start Value: 1276733007324
End Value: 1276733007399
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 2
Running Time: 7.5e-06 seconds
Start Value: 1276733019863
End Value: 1276733019943
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 3
Running Time: 8e-06 seconds
Start Value: 1276733033340
End Value: 1276733033424
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 4
Running Time: 8.4e-06 seconds
Start Value: 1276733048109
End Value: 1276733048247
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 5
Running Time: 1.38e-05 seconds
Start Value: 1276733060694
End Value: 1276733060778
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 6
Running Time: 8.4e-06 seconds
Start Value: 1276733073185
End Value: 1276733073271
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 7
Running Time: 8.6e-06 seconds
Start Value: 1276733088876
End Value: 1276733088974
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 8
Running Time: 9.8e-06 seconds
Start Value: 1276733103012
End Value: 1276733103105
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 9
Running Time: 9.3e-06 seconds
Start Value: 1276733116001
End Value: 1276733116093
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 10
Running Time: 9.2e-06 seconds
Average time: 9.03e-06
-----
Press any key to continue . . .

```

Figure 46: execution running time for array size 2048

In figure 47, we can see the array size average execution running time for array size 4096.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1277470472487
End Value: 1277470472628
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 1
Running Time: 1.41e-05 seconds
Start Value: 1277470501525
End Value: 1277470501670
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 2
Running Time: 1.45e-05 seconds
Start Value: 1277470514294
End Value: 1277470514488
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 3
Running Time: 1.94e-05 seconds
Start Value: 1277470533296
End Value: 1277470533502
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 4
Running Time: 2.06e-05 seconds
Start Value: 1277470549558
End Value: 1277470549763
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 5
Running Time: 2.05e-05 seconds
Start Value: 1277470566266
End Value: 1277470566470
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 6
Running Time: 2.04e-05 seconds
Start Value: 1277470583741
End Value: 1277470583931
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 7
Running Time: 1.9e-05 seconds
Start Value: 1277470598073
End Value: 1277470598287
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 8
Running Time: 2.14e-05 seconds
Start Value: 1277470611851
End Value: 1277470612033
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 9
Running Time: 1.82e-05 seconds
Start Value: 1277470644598
End Value: 1277470644763
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 10
Running Time: 1.65e-05 seconds
Average time: 1.846e-05
-----
Press any key to continue . . .

```

Figure 47: execution running time for array size 4096

In figure 48, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (seconds)
8	4.50E-07
16	5.00E-07
32	4.90E-07
64	8.20E-07
128	1.46E-06
256	1.13E-06
512	2.70E-06
1024	3.86E-06
2048	9.03E-06
4096	1.85E-05

Figure 48: Average Runtime Table (Compiler Generated Optimization)

In figure 49, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

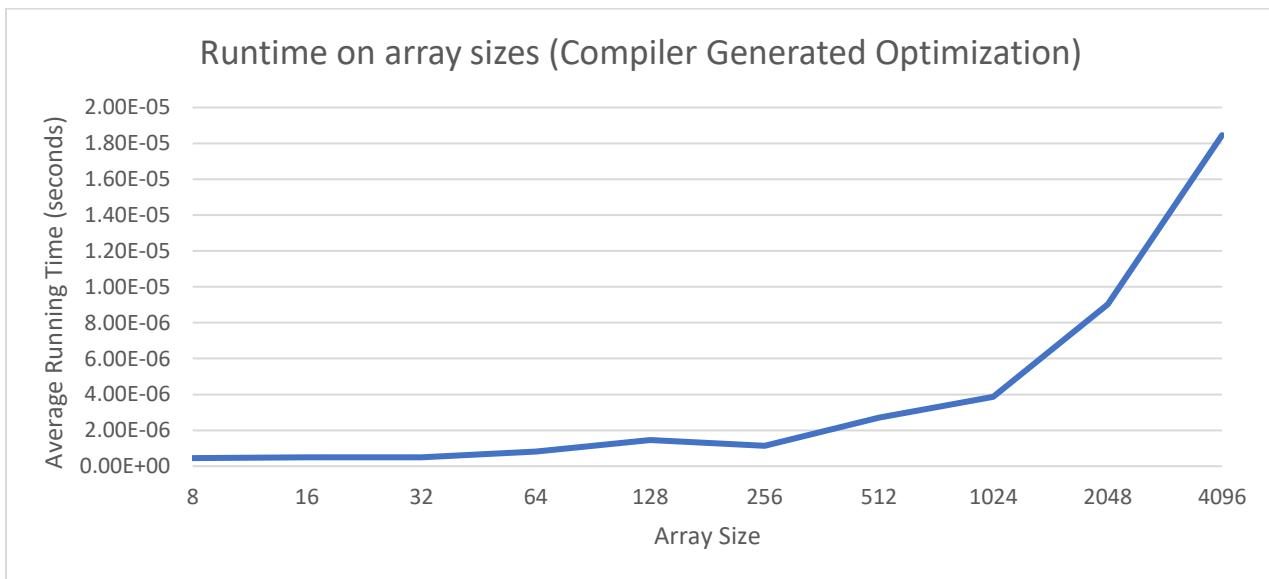


Figure 49: Runtime graph (Compiler Generated Optimization)

Dot Product [Manually Optimized]:

In figure 50, we can see the array size average execution running time for array size 8.

```

Select E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1294779580445
End Value: 1294779580451
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 1
Running Time: 6e-07 seconds
Start Value: 1294779600336
End Value: 1294779600339
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 2
Running Time: 3e-07 seconds
Start Value: 1294779609158
End Value: 1294779609161
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1294779616666
End Value: 1294779616668
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1294779623470
End Value: 1294779623471
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 5
Running Time: 1e-07 seconds
Start Value: 1294779630390
End Value: 1294779630391
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 6
Running Time: 1e-07 seconds
Start Value: 1294779637318
End Value: 1294779637319
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 7
Running Time: 1e-07 seconds
Start Value: 1294779643565
End Value: 1294779643567
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1294779651435
End Value: 1294779651437
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 9
Running Time: 2e-07 seconds
Start Value: 1294779659576
End Value: 1294779659577
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 10
Running Time: 1e-07 seconds
Average time: 2.2e-07
-----
Press any key to continue . . .

```

Figure 50: execution running time for array size 8.

In figure 51, we can see the array size average execution running time for array size 16.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 130898628834
End Value: 1308986288323
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 1
Running Time: 9e-07 seconds
Start Value: 1308986317868
End Value: 1308986317870
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 2
Running Time: 2e-07 seconds
Start Value: 1308986327742
End Value: 1308986327745
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1308986336780
End Value: 1308986336782
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1308986345823
End Value: 1308986345826
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 5
Running Time: 3e-07 seconds
Start Value: 1308986355114
End Value: 1308986355116
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1308986365077
End Value: 1308986365079
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 7
Running Time: 2e-07 seconds
Start Value: 1308986373174
End Value: 1308986373176
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1308986387382
End Value: 1308986387384
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 9
Running Time: 2e-07 seconds
Start Value: 1308986394952
End Value: 1308986394954
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 10
Running Time: 2e-07 seconds
Average time: 2.9e-07
-----
Press any key to continue . . .

```

Figure 51: execution running time for array size 16

In figure 52, we can see the array size average execution running time for array size 32.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1309394835912
End Value: 1309394835921
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 1
Running Time: 9e-07 seconds
Start Value: 1309394856371
End Value: 1309394856374
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 2
Running Time: 3e-07 seconds
Start Value: 1309394864230
End Value: 1309394864232
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1309394872702
End Value: 1309394872704
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1309394880962
End Value: 1309394880965
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 5
Running Time: 3e-07 seconds
Start Value: 1309394889720
End Value: 1309394889722
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1309394896885
End Value: 1309394896889
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1309394903828
End Value: 1309394903830
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1309394910520
End Value: 1309394910522
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 9
Running Time: 2e-07 seconds
Start Value: 1309394917517
End Value: 1309394917519
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 10
Running Time: 2e-07 seconds
Average time: 3.1e-07
-----
Press any key to continue . . . .

```

Figure 52: execution running time for array size 32

In figure 53, we can see the array size average execution running time for array size 64.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1309599133566
End Value: 1309599133578
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 1
Running Time: 1.2e-06 seconds
Start Value: 1309599161458
End Value: 1309599161463
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 2
Running Time: 5e-07 seconds
Start Value: 1309599170175
End Value: 1309599170178
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1309599182153
End Value: 1309599182157
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 4
Running Time: 4e-07 seconds
Start Value: 1309599188906
End Value: 1309599188910
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 5
Running Time: 4e-07 seconds
Start Value: 1309599195826
End Value: 1309599195830
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 6
Running Time: 4e-07 seconds
Start Value: 1309599202446
End Value: 1309599202450
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 7
Running Time: 4e-07 seconds
Start Value: 1309599211296
End Value: 1309599211300
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 8
Running Time: 4e-07 seconds
Start Value: 1309599219898
End Value: 1309599219901
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 9
Running Time: 3e-07 seconds
Start Value: 1309599227140
End Value: 1309599227144
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 10
Running Time: 4e-07 seconds
Average time: 4.7e-07
-----
Press any key to continue . . .

```

Figure 53: execution running time for array size 64

In figure 54, we can see the array size average execution running time for array size 128.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1309854148854
End Value: 1309854148864
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 1
Running Time: 1e-06 seconds
Start Value: 1309854174116
End Value: 1309854174125
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 2
Running Time: 9e-07 seconds
Start Value: 1309854181036
End Value: 1309854181040
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 3
Running Time: 4e-07 seconds
Start Value: 1309854188699
End Value: 1309854188704
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 4
Running Time: 5e-07 seconds
Start Value: 1309854195975
End Value: 1309854195981
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 5
Running Time: 6e-07 seconds
Start Value: 1309854203309
End Value: 1309854203313
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 6
Running Time: 4e-07 seconds
Start Value: 1309854210309
End Value: 1309854210314
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 7
Running Time: 5e-07 seconds
Start Value: 1309854219544
End Value: 1309854219550
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 8
Running Time: 6e-07 seconds
Start Value: 1309854227645
End Value: 1309854227650
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 9
Running Time: 5e-07 seconds
Start Value: 1309854234504
End Value: 1309854234509
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 10
Running Time: 5e-07 seconds
Average time: 5.9e-07
-----
Press any key to continue . . .

```

Figure 54: execution running time for array size 128

In figure 55, we can see the array size average execution running time for array size 256.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1310295857946
End Value: 1310295857959
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 1
Running Time: 1.3e-06 seconds
Start Value: 1310295885843
End Value: 1310295885857
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 2
Running Time: 1.4e-06 seconds
Start Value: 1310295894443
End Value: 1310295894455
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 3
Running Time: 1.2e-06 seconds
Start Value: 1310295902284
End Value: 1310295902294
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 4
Running Time: 1e-06 seconds
Start Value: 1310295909924
End Value: 1310295909934
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 5
Running Time: 1e-06 seconds
Start Value: 1310295919230
End Value: 1310295919239
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 6
Running Time: 9e-07 seconds
Start Value: 1310295929178
End Value: 1310295929195
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 7
Running Time: 1.7e-06 seconds
Start Value: 1310295936656
End Value: 1310295936666
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 8
Running Time: 1e-06 seconds
Start Value: 1310295947105
End Value: 1310295947125
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 9
Running Time: 2e-06 seconds
Start Value: 1310295961195
End Value: 1310295961204
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 10
Running Time: 9e-07 seconds
Average time: 1.24e-06
-----
Press any key to continue . . .

```

Figure 55: execution running time for array size 256

In figure 56, we can see the array size average execution running time for array size 512.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1310545060275
End Value: 1310545060521
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 1
Running Time: 2.46e-05 seconds
Start Value: 1310545092216
End Value: 1310545092234
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 2
Running Time: 1.8e-06 seconds
Start Value: 1310545103992
End Value: 1310545104012
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 3
Running Time: 2e-06 seconds
Start Value: 1310545112517
End Value: 1310545112536
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 4
Running Time: 1.9e-06 seconds
Start Value: 1310545123450
End Value: 1310545123468
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 5
Running Time: 1.8e-06 seconds
Start Value: 1310545138267
End Value: 1310545138291
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 6
Running Time: 2.4e-06 seconds
Start Value: 1310545147602
End Value: 1310545147637
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 7
Running Time: 3.5e-06 seconds
Start Value: 1310545157374
End Value: 1310545157407
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 8
Running Time: 3.3e-06 seconds
Start Value: 1310545166291
End Value: 1310545166309
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 9
Running Time: 1.8e-06 seconds
Start Value: 1310545174298
End Value: 1310545174313
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 10
Running Time: 1.5e-06 seconds
Average time: 4.46e-06
-----
Press any key to continue . . .

```

Figure 56: execution running time for array size 512

In figure 57, we can see the array size average execution running time for array size 1024.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1310795409227
End Value: 1310795409265
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 1
Running Time: 3.8e-06 seconds
Start Value: 1310795432684
End Value: 1310795432721
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 2
Running Time: 3.7e-06 seconds
Start Value: 1310795441749
End Value: 1310795441780
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 3
Running Time: 3.1e-06 seconds
Start Value: 1310795455652
End Value: 1310795455682
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 4
Running Time: 3e-06 seconds
Start Value: 1310795462796
End Value: 1310795462825
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 5
Running Time: 2.9e-06 seconds
Start Value: 1310795470079
End Value: 1310795471008
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 6
Running Time: 2.9e-06 seconds
Start Value: 1310795479195
End Value: 1310795479222
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 7
Running Time: 2.7e-06 seconds
Start Value: 1310795486575
End Value: 1310795486602
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 8
Running Time: 2.7e-06 seconds
Start Value: 1310795496480
End Value: 1310795496508
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 9
Running Time: 2.8e-06 seconds
Start Value: 1310795503733
End Value: 1310795503760
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 10
Running Time: 2.7e-06 seconds
Average time: 3.03e-06
-----
Press any key to continue . . .

```

Figure 57: execution running time for array size 1024

In figure 58, we can see the array size average execution running time for array size 2048.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1311288932997
End Value: 1311288933056
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 1
Running Time: 5.9e-06 seconds
Start Value: 1311288959277
End Value: 1311288959335
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 2
Running Time: 5.8e-06 seconds
Start Value: 1311288968204
End Value: 1311288968261
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 3
Running Time: 5.7e-06 seconds
Start Value: 1311288976371
End Value: 1311288976430
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 4
Running Time: 5.9e-06 seconds
Start Value: 1311288984504
End Value: 1311288984564
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 5
Running Time: 6e-06 seconds
Start Value: 1311288993015
End Value: 1311288993075
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 6
Running Time: 6e-06 seconds
Start Value: 1311289001657
End Value: 1311289001718
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 7
Running Time: 6.1e-06 seconds
Start Value: 1311289011468
End Value: 1311289011529
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 8
Running Time: 6.1e-06 seconds
Start Value: 1311289023472
End Value: 1311289023535
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 9
Running Time: 6.3e-06 seconds
Start Value: 1311289031972
End Value: 1311289032030
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 10
Running Time: 5.8e-06 seconds
Average time: 5.96e-06
-----
Press any key to continue . . .

```

Figure 58: execution running time for array size 2048

In figure 59, we can see the array size average execution running time for array size 4096.

```

Select E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1311619091150
End Value: 1311619091299
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 1
Running Time: 1.49e-05 seconds
Start Value: 1311619117117
End Value: 1311619117238
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 2
Running Time: 1.21e-05 seconds
Start Value: 1311619127539
End Value: 1311619127650
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 3
Running Time: 1.11e-05 seconds
Start Value: 1311619135021
End Value: 1311619135133
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 4
Running Time: 1.12e-05 seconds
Start Value: 1311619142740
End Value: 1311619142849
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 5
Running Time: 1.09e-05 seconds
Start Value: 1311619150423
End Value: 1311619150528
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 6
Running Time: 1.05e-05 seconds
Start Value: 1311619158761
End Value: 1311619158874
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 7
Running Time: 1.13e-05 seconds
Start Value: 1311619165998
End Value: 1311619166108
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 8
Running Time: 1.1e-05 seconds
Start Value: 1311619181063
End Value: 1311619181183
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 9
Running Time: 1.2e-05 seconds
Start Value: 1311619193035
End Value: 1311619193149
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 10
Running Time: 1.14e-05 seconds
Average time: 1.164e-05
-----
Press any key to continue . . .

```

Figure 59: execution running time for array size 4096

In figure 60, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (seconds)
8	2.20E-07
16	2.90E-07
32	3.10E-07
64	4.70E-07
128	5.90E-07
256	1.24E-06
512	2.46E-06
1024	3.03E-06
2048	5.96E-06
4096	1.16E-05

Figure 60: Average Runtime Table (Manually Optimized)

In figure 61, we can see that I made a graph for the average runtime based on the array sizes. We can see how the runtime went as the array size increased.

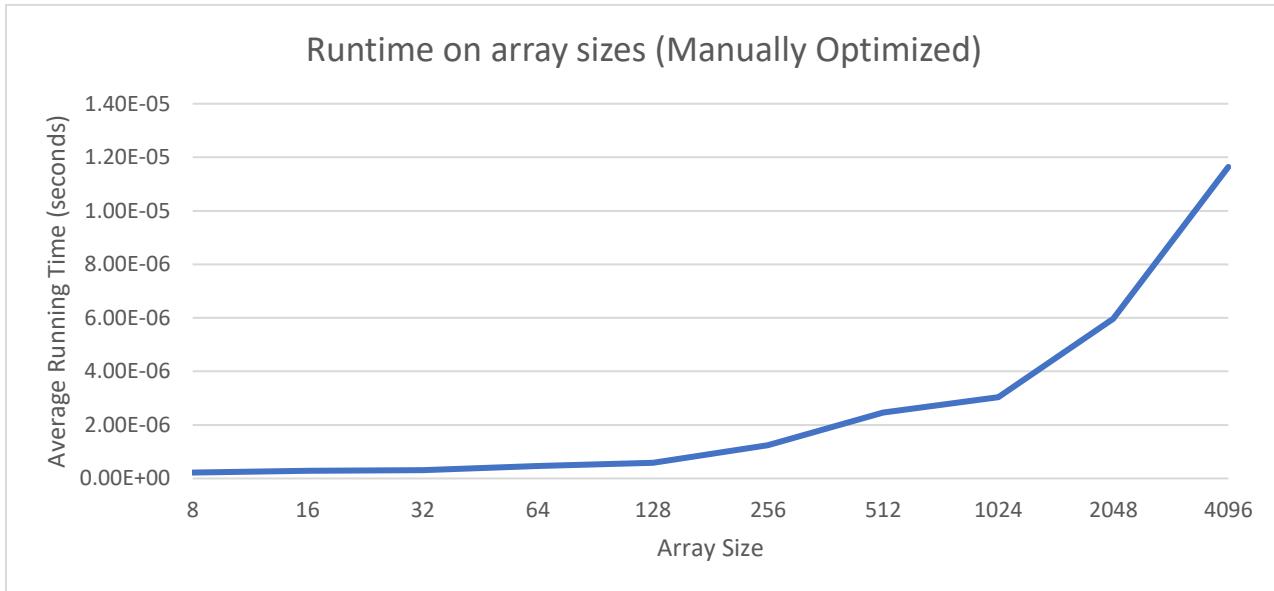


Figure 61: Runtime graph (Manually Optimized)

VDPPS:

In figure 62, we can see the array size average execution running time for array size 8.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1423313622308
End Value: 1423313622317
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 1
Running Time: 9e-07 seconds
Start Value: 1423313658253
End Value: 1423313658256
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 2
Running Time: 3e-07 seconds
Start Value: 1423313669763
End Value: 1423313669766
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1423313680158
End Value: 1423313680160
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1423313688479
End Value: 1423313688481
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 5
Running Time: 2e-07 seconds
Start Value: 1423313695779
End Value: 1423313695781
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1423313703250
End Value: 1423313703252
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 7
Running Time: 2e-07 seconds
Start Value: 1423313714375
End Value: 1423313714377
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1423313725695
End Value: 1423313725699
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 9
Running Time: 4e-07 seconds
Start Value: 1423313732599
End Value: 1423313732601
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 8
Run number: 10
Running Time: 2e-07 seconds
Average time: 3.1e-07
-----
Press any key to continue . . .

```

Figure 62: execution running time for array size 8

In figure 63, we can see the array size average execution running time for array size 16.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1427447941602
End Value: 1427447941608
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 1
Running Time: 6e-07 seconds
Start Value: 1427447964929
End Value: 1427447964931
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 2
Running Time: 2e-07 seconds
Start Value: 1427447972663
End Value: 1427447972665
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1427447987196
End Value: 1427447987198
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1427447994570
End Value: 1427447994571
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 5
Running Time: 1e-07 seconds
Start Value: 1427448006908
End Value: 1427448006910
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 6
Running Time: 2e-07 seconds
Start Value: 1427448016657
End Value: 1427448016659
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 7
Running Time: 2e-07 seconds
Start Value: 1427448024834
End Value: 1427448024836
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1427448031342
End Value: 1427448031344
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 9
Running Time: 2e-07 seconds
Start Value: 1427448039201
End Value: 1427448039202
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 16
Run number: 10
Running Time: 1e-07 seconds
Average time: 2.2e-07
-----
Press any key to continue . . .

```

Figure 63: execution running time for array size 16

In figure 64, we can see the array size average execution running time for array size 32.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1427759365151
End Value: 1427759365158
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 1
Running Time: 7e-07 seconds
Start Value: 1427759395850
End Value: 1427759395855
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 2
Running Time: 5e-07 seconds
Start Value: 1427759407647
End Value: 1427759407649
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 3
Running Time: 2e-07 seconds
Start Value: 1427759419197
End Value: 1427759419199
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 4
Running Time: 2e-07 seconds
Start Value: 1427759428280
End Value: 1427759428282
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 5
Running Time: 2e-07 seconds
Start Value: 1427759435906
End Value: 1427759435909
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1427759443080
End Value: 1427759443083
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 7
Running Time: 3e-07 seconds
Start Value: 1427759450026
End Value: 1427759450028
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 8
Running Time: 2e-07 seconds
Start Value: 1427759459055
End Value: 1427759459059
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 9
Running Time: 4e-07 seconds
Start Value: 1427759474174
End Value: 1427759474176
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 32
Run number: 10
Running Time: 2e-07 seconds
Average time: 3.2e-07
-----
Press any key to continue . . . .

```

Figure 64: execution running time for array size 32

In figure 65, we can see the array size average execution running time for array size 64.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1427971429097
End Value: 1427971429110
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 1
Running Time: 1.3e-06 seconds
Start Value: 1427971466536
End Value: 1427971466541
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 2
Running Time: 5e-07 seconds
Start Value: 1427971477499
End Value: 1427971477502
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 3
Running Time: 3e-07 seconds
Start Value: 1427971489436
End Value: 1427971489439
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 4
Running Time: 3e-07 seconds
Start Value: 1427971503368
End Value: 1427971503371
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 5
Running Time: 3e-07 seconds
Start Value: 1427971515581
End Value: 1427971515584
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 6
Running Time: 3e-07 seconds
Start Value: 1427971525207
End Value: 1427971525210
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 7
Running Time: 3e-07 seconds
Start Value: 1427971532505
End Value: 1427971532509
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 8
Running Time: 4e-07 seconds
Start Value: 1427971539515
End Value: 1427971539518
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 9
Running Time: 3e-07 seconds
Start Value: 1427971547421
End Value: 1427971547424
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 64
Run number: 10
Running Time: 3e-07 seconds
Average time: 4.3e-07
-----
Press any key to continue . . .

```

Figure 65: execution running time for array size 64

In figure 66, we can see the array size average execution running time for array size 128.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1428487413024
End Value: 1428487413034
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 1
Running Time: 1e-06 seconds
Start Value: 1428487442017
End Value: 1428487442026
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 2
Running Time: 9e-07 seconds
Start Value: 1428487453601
End Value: 1428487453606
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 3
Running Time: 5e-07 seconds
Start Value: 1428487461571
End Value: 1428487461576
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 4
Running Time: 5e-07 seconds
Start Value: 1428487473999
End Value: 1428487474005
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 5
Running Time: 6e-07 seconds
Start Value: 1428487480990
End Value: 1428487480995
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 6
Running Time: 5e-07 seconds
Start Value: 1428487489665
End Value: 1428487489670
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 7
Running Time: 5e-07 seconds
Start Value: 1428487501503
End Value: 1428487501508
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 8
Running Time: 5e-07 seconds
Start Value: 1428487511086
End Value: 1428487511093
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 9
Running Time: 7e-07 seconds
Start Value: 1428487521286
End Value: 1428487521293
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 128
Run number: 10
Running Time: 7e-07 seconds
Average time: 6.4e-07
-----
Press any key to continue . . . .

```

Figure 66: execution running time for array size 128

In figure 67, we can see the array size average execution running time for array size 256.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1428777188843
End Value: 1428777188856
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 1
Running Time: 1.3e-06 seconds
Start Value: 1428777214545
End Value: 1428777214558
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 2
Running Time: 1.3e-06 seconds
Start Value: 1428777225741
End Value: 1428777225750
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 3
Running Time: 9e-07 seconds
Start Value: 1428777243382
End Value: 1428777243391
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 4
Running Time: 9e-07 seconds
Start Value: 1428777250138
End Value: 1428777250146
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 5
Running Time: 8e-07 seconds
Start Value: 1428777257116
End Value: 1428777257123
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 6
Running Time: 7e-07 seconds
Start Value: 1428777265606
End Value: 1428777265613
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 7
Running Time: 7e-07 seconds
Start Value: 1428777274893
End Value: 1428777274901
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 8
Running Time: 8e-07 seconds
Start Value: 1428777281699
End Value: 1428777281707
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 9
Running Time: 8e-07 seconds
Start Value: 1428777288492
End Value: 1428777288500
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 256
Run number: 10
Running Time: 8e-07 seconds
Average time: 9e-07
-----
Press any key to continue . . .

```

Figure 67: execution running time for array size 256

In figure 68, we can see the array size average execution running time for array size 512.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1428979259004
End Value: 1428979259025
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 1
Running Time: 2.1e-06 seconds
Start Value: 1428979292573
End Value: 1428979292597
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 2
Running Time: 2.4e-06 seconds
Start Value: 1428979303399
End Value: 1428979303418
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 3
Running Time: 1.9e-06 seconds
Start Value: 1428979311312
End Value: 1428979311327
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 4
Running Time: 1.5e-06 seconds
Start Value: 1428979319155
End Value: 1428979319170
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 5
Running Time: 1.5e-06 seconds
Start Value: 1428979332094
End Value: 1428979332110
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 6
Running Time: 1.6e-06 seconds
Start Value: 1428979340120
End Value: 1428979340136
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 7
Running Time: 1.6e-06 seconds
Start Value: 1428979353079
End Value: 1428979353115
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 8
Running Time: 3.6e-06 seconds
Start Value: 1428979361444
End Value: 1428979361461
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 9
Running Time: 1.7e-06 seconds
Start Value: 1428979375729
End Value: 1428979375744
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 512
Run number: 10
Running Time: 1.5e-06 seconds
Average time: 1.94e-06
-----
Press any key to continue . . .

```

Figure 68: execution running time for array size 512

In figure 69, we can see the array size average execution running time for array size 1024.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1432494660729
End Value: 1432494660767
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 1
Running Time: 3.8e-06 seconds
Start Value: 1432494688629
End Value: 1432494688659
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 2
Running Time: 3e-06 seconds
Start Value: 1432494700330
End Value: 1432494700359
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 3
Running Time: 2.9e-06 seconds
Start Value: 1432494718078
End Value: 1432494718106
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 4
Running Time: 2.8e-06 seconds
Start Value: 1432494725663
End Value: 1432494725692
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 5
Running Time: 2.9e-06 seconds
Start Value: 1432494732957
End Value: 1432494732984
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 6
Running Time: 2.7e-06 seconds
Start Value: 1432494740414
End Value: 1432494740442
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 7
Running Time: 2.8e-06 seconds
Start Value: 1432494747922
End Value: 1432494747994
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 8
Running Time: 7.2e-06 seconds
Start Value: 1432494758163
End Value: 1432494758193
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 9
Running Time: 3e-06 seconds
Start Value: 1432494765955
End Value: 1432494765986
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 1024
Run number: 10
Running Time: 3.1e-06 seconds
Average time: 3.42e-06
-----
Press any key to continue . . .

```

Figure 69: execution running time for array size 1024

In figure 70, we can see the array size average execution running time for array size 2048.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1434182295443
End Value: 1434182295526
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 1
Running Time: 8.3e-06 seconds
Start Value: 1434182323128
End Value: 1434182323191
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 2
Running Time: 6.3e-06 seconds
Start Value: 1434182333527
End Value: 1434182333584
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 3
Running Time: 5.7e-06 seconds
Start Value: 1434182344110
End Value: 1434182344166
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 4
Running Time: 5.6e-06 seconds
Start Value: 1434182363778
End Value: 1434182363880
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 5
Running Time: 1.02e-05 seconds
Start Value: 1434182372466
End Value: 1434182372518
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 6
Running Time: 5.2e-06 seconds
Start Value: 1434182383853
End Value: 1434182384086
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 7
Running Time: 2.33e-05 seconds
Start Value: 1434182397086
End Value: 1434182397143
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 8
Running Time: 5.7e-06 seconds
Start Value: 1434182413951
End Value: 1434182414003
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 9
Running Time: 5.2e-06 seconds
Start Value: 1434182432131
End Value: 1434182432194
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 2048
Run number: 10
Running Time: 6.3e-06 seconds
Average time: 8.18e-06
-----
Press any key to continue . . .

```

Figure 70: execution running time for array size 2048

In figure 71, we can see the array size average execution running time for array size 4096.

```

E:\CSC_342\Assignments\Take Home Test 3\Emdad_15thMay2022_Take_Home_Test_3\Debug\Emdad_15thMay2022_Take_Home_Test_3.exe
Start Value: 1438297327206
End Value: 1438297327327
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 1
Running Time: 1.21e-05 seconds
Start Value: 1438297352837
End Value: 1438297352953
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 2
Running Time: 1.16e-05 seconds
Start Value: 1438297363872
End Value: 1438297363981
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 3
Running Time: 1.09e-05 seconds
Start Value: 1438297373422
End Value: 1438297373526
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 4
Running Time: 1.04e-05 seconds
Start Value: 1438297383020
End Value: 1438297383126
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 5
Running Time: 1.06e-05 seconds
Start Value: 1438297392468
End Value: 1438297392579
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 6
Running Time: 1.11e-05 seconds
Start Value: 1438297404842
End Value: 1438297404950
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 7
Running Time: 1.08e-05 seconds
Start Value: 1438297412839
End Value: 1438297412944
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 8
Running Time: 1.05e-05 seconds
Start Value: 1438297421363
End Value: 1438297421464
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 9
Running Time: 1.01e-05 seconds
Start Value: 1438297433410
End Value: 1438297433514
QueryPerformance minimum resolution: 1/10000000 seconds
Array size: 4096
Run number: 10
Running Time: 1.04e-05 seconds
Average time: 1.085e-05
-----
Press any key to continue . . .

```

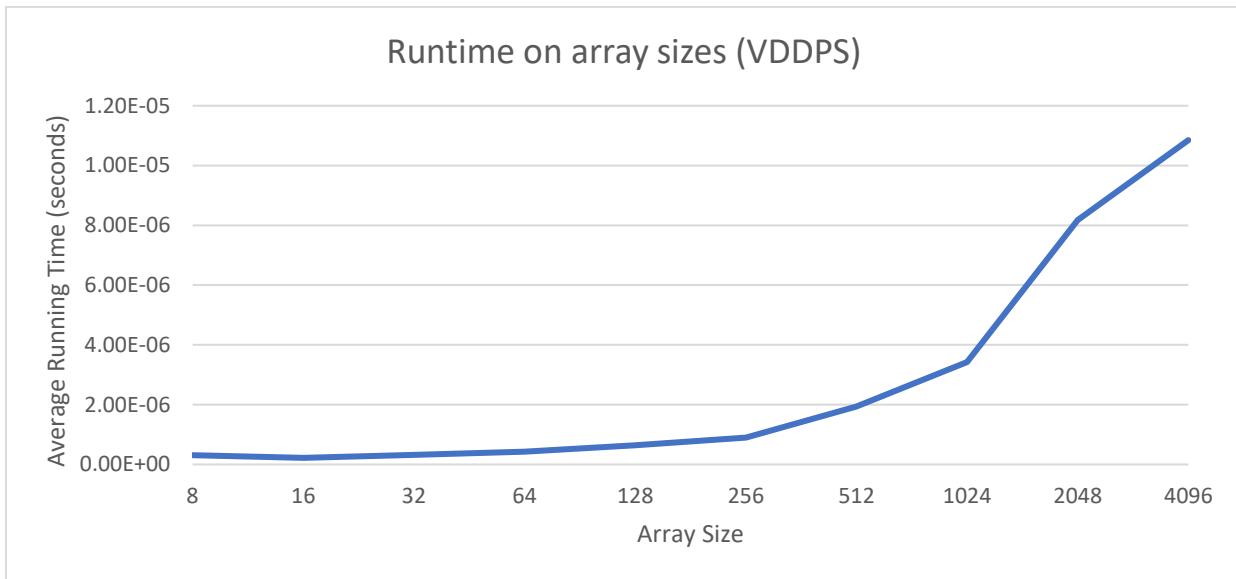
Figure 71: execution running time for array size 4096

In figure 72, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (seconds)
8	3.10E-07
16	2.20E-07
32	3.20E-07
64	4.30E-07
128	6.40E-07
256	9.00E-07
512	1.94E-06
1024	3.42E-06
2048	8.18E-06
4096	1.09E-05

Figure 72: : Average Runtime Table (VDPPS)

In figure 73, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

*Figure 73: Runtime graph (VDDPS)*

Intel x64 (Linux g++):

Dot Product [Non-Optimized]:

In figure 74, we can see the average time for array size {8,16,32,64,128,256,512,1024,2048,4096}. We did the execution in order. We ran **g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out** to compile the files. It generated an a.out output file. Then we did **./a.out** to execute the file.

```
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 8
Average time: 995 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 16
Average time: 1023 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 32
Average time: 1111 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 64
Average time: 669 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 128
Average time: 1981 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 256
Average time: 3003 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 512
Average time: 5162 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 1024
Average time: 9408 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 2048
Average time: 28550 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o0 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 4096
Average time: 34675 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ █
```

Figure 74: Runtime on array sizes(non-Optimized)

In figure 75, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (ns)
8	995
16	1023
32	1111
64	669
128	1981
256	3003
512	5162
1024	9408
2048	28550
4096	34675

Figure 75: : Average Runtime Table (Non-Optimized)

In figure 76, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

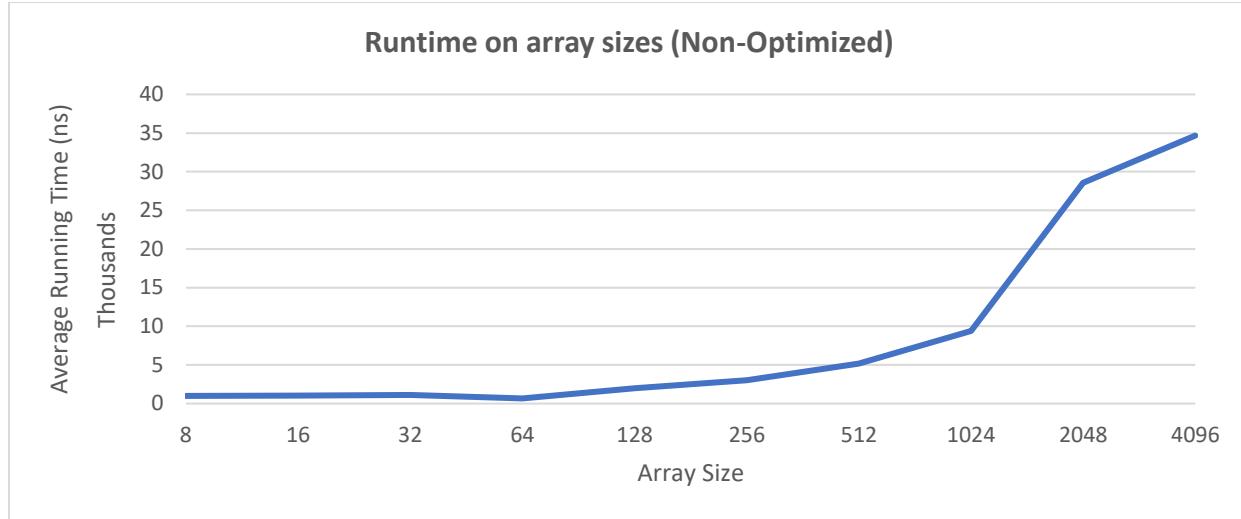


Figure 76: Runtime graph (non-Optimized)

Dot Product [Compiler Generated Optimization]:

In figure 77, we can see the average time for array size $\{8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096\}$. We did the execution in order. We ran **g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out** to compile the files. It generated an a.out output file. Then we did **./a.out** to execute the file.

```
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 8
Average time: 654 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 16
Average time: 744 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 32
Average time: 1202 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 64
Average time: 1425 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 128
Average time: 1943 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 256
Average time: 2778 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 512
Average time: 3256 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 1024
Average time: 9433 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 2048
Average time: 17757 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -O1 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 4096
Average time: 47222 ns
```

Figure 77: Runtime on array sizes (Compiler Generated Optimization)

In figure 78, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (ns)
8	654
16	744
32	1202
64	1425
128	1943
256	2778
512	3256
1024	9433
2048	17757
4096	47222

Figure 78: Average Runtime Table (Compiler Generated Optimization)

In figure 79, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

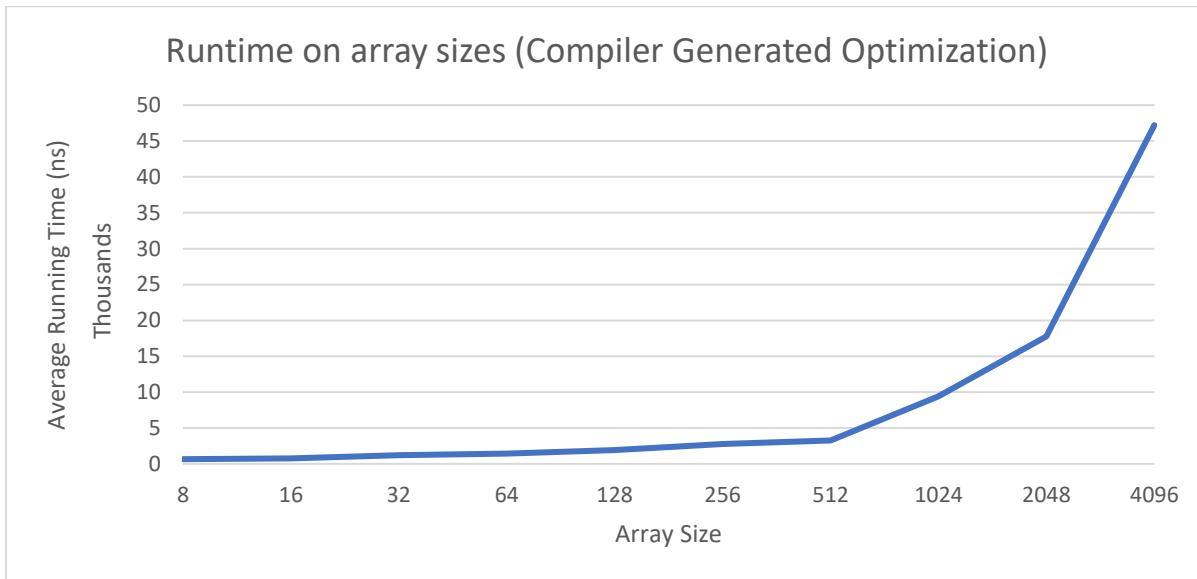


Figure 79: Runtime graph (Compiler Generated Optimization)

Dot Product [Manually Optimized]:

In figure , we can see the average time for array size {8,16,32,64,128,256,512,1024,2048,4096}. We did the execution in order. We ran **g++ -o2 Emdad_Source.cpp**

Emdad_DotProductPointer.cpp -o a.out to compile the files. It generated an a.out output file. Then we did **./a.out** to execute the file.

```
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 8
Average time: 503 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 16
Average time: 1108 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 32
Average time: 774 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 64
Average time: 1425 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 128
Average time: 1474 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 256
Average time: 2935 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 512
Average time: 5085 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 1024
Average time: 8520 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 2048
Average time: 17682 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o2 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 4096
Average time: 10596 ns
```

Figure 80: Runtime on array sizes (Manually Optimized)

In figure 81, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (ns)
8	503
16	1108
32	774
64	1425
128	1474
256	2935
512	5085
1024	8520
2048	10682
4096	12596

Figure 81: Average Runtime Table (Manually Optimized)

In figure 82, we can see that I made a graph for the average runtime based on the array sizes. We can see how the runtime went as the array size increased.

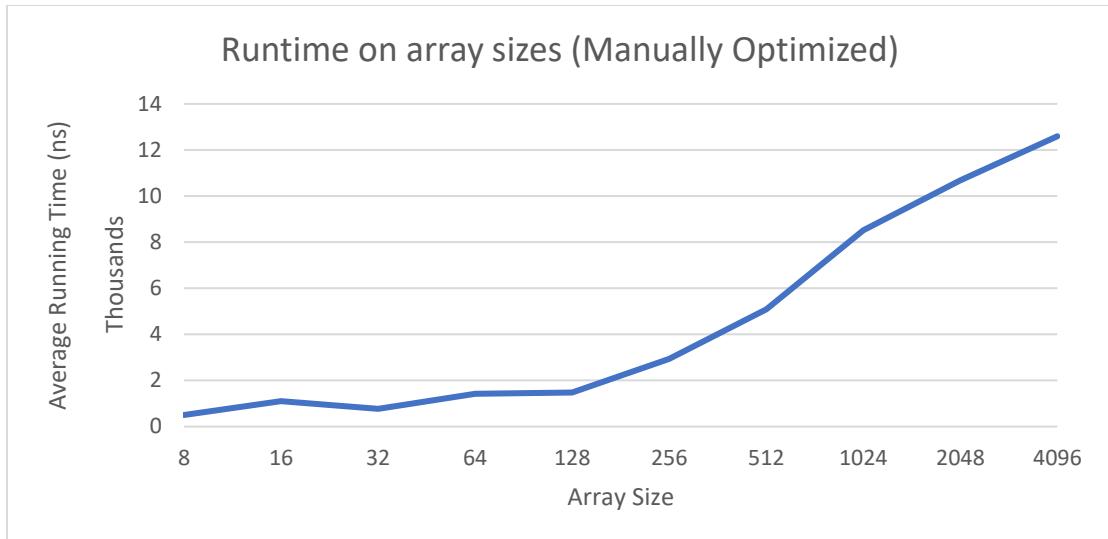


Figure 82: Runtime graph (Manually Optimized)

VDPPS:

In figure , we can see the average time for array size {8,16,32,64,128,256,512,1024,2048,4096}. We did the execution in order. We ran **g++ -o3 Emdad_Source.cpp**

Emdad_DotProductPointer.cpp -o a.out to compile the files. It generated an a.out output file. Then we did **./a.out** to execute the file.

```
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 8
Average time: 1030 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 16
Average time: 1042 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 32
Average time: 1177 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 64
Average time: 1430 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 128
Average time: 2079 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 256
Average time: 2951 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 512
Average time: 5126 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 1024
Average time: 6852 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 2048
Average time: 17720 ns
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ g++ -o3 Emdad_Source.cpp Emdad_DotProductPointer.cpp -o a.out
main@main-VirtualBox:~/Desktop/cs-343/takeHomeTest-3$ ./a.out
Array size: 4096
Average time: 53067 ns
```

Figure 83: Runtime on array sizes (VDPPS)

In figure 84, we can see that I made a table for the average runtime based on the array sizes.

Array Size	Average Running Time (ns)
8	1030
16	1042
32	1177
64	1430
128	2079
256	2951
512	5126
1024	6852
2048	17720
4096	53067

Figure 84: Average Runtime Table (VDPPS)

In figure 85, we can see that I made a graph for the average runtime based on the array sizes. We can how the runtime went as the array size increased.

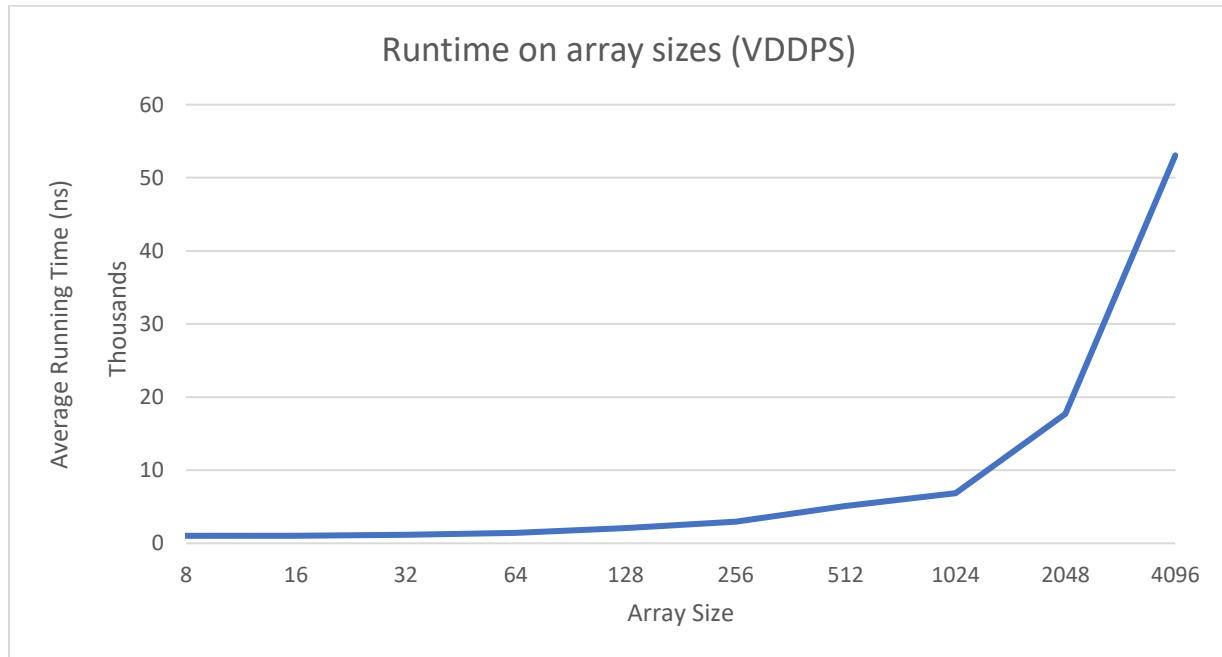


Figure 85: Runtime graph (VDPPS)

Explanation:

Intel x86 (Visual Studio Code):

In figure 86, we can see that I took the average time from all the methods to run the program for each vector array size and put them in a single table. I am going to make a graph out of this table to visualize to check which method is faster than others.

Array Size	Non-Optimized (seconds)	Compiler Generated Optimization (seconds)	Manually Optimized (seconds)	VDDPS (seconds)
8	2.60E-07	4.50E-07	2.20E-07	3.10E-07
16	2.70E-07	5.00E-07	2.90E-07	2.20E-07
32	4.00E-07	4.90E-07	3.10E-07	3.20E-07
64	5.70E-07	8.20E-07	4.70E-07	4.30E-07
128	6.90E-07	1.46E-06	5.90E-07	6.40E-07
256	9.80E-07	1.13E-06	1.24E-06	9.00E-07
512	2.36E-06	2.70E-06	2.46E-06	1.94E-06
1024	3.43E-06	3.86E-06	3.03E-06	3.42E-06
2048	7.58E-06	9.03E-06	5.96E-06	8.18E-06
4096	1.60E-05	1.85E-05	1.16E-05	1.09E-05

Figure 86: Average Runtime Table (all methods)

In figure 87, we can see the generated graph from the table(figure 86).

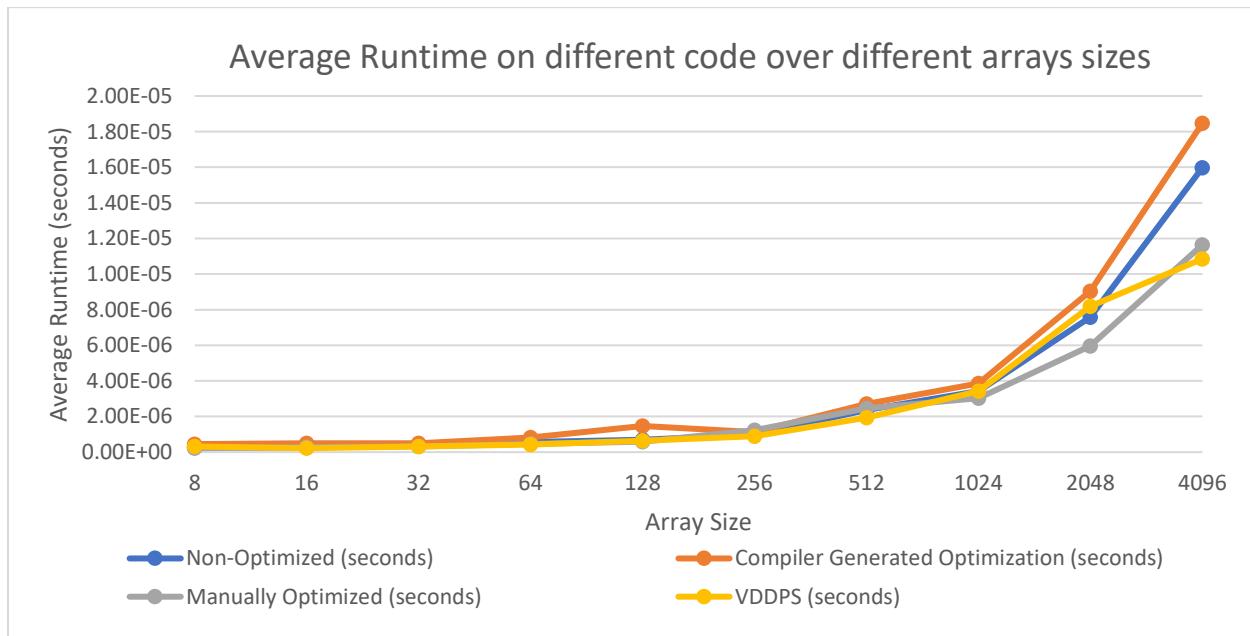


Figure 87: Average Runtime graph (all methods)

The more we optimized the code the more runtime decreased. We can see that VDPPS instruction has the lowest runtime meaning it ran faster than other methods. It is because VDPPS instruction used vectorization and parallelization which is more efficient for the larger array sizes and can be noticeable.

Intel x64 (Linux g++):

In figure 88, we can see that I took the average time from all the methods to run the program for each vector array size and put them in a single table. I am going to make a graph out of this table to visualize to check which method is faster than others.

Array Size	Non-Optimized (ns)	Compiler Generated Optimization (ns)	Manually Optimized (ns)	VDDPS (ns)
8	995	654	503	1030
16	1023	744	1108	1042
32	1111	1202	774	1177
64	669	1425	1425	1430
128	1981	1943	1474	2079
256	3003	2778	2935	2951
512	5162	3256	5085	5126
1024	9408	9433	8520	6852
2048	28550	17757	10682	17720
4096	34675	47222	12596	53067

Figure 88: Average Runtime Table (all methods)

In figure 89, we can see the generated graph from the table(figure 88).

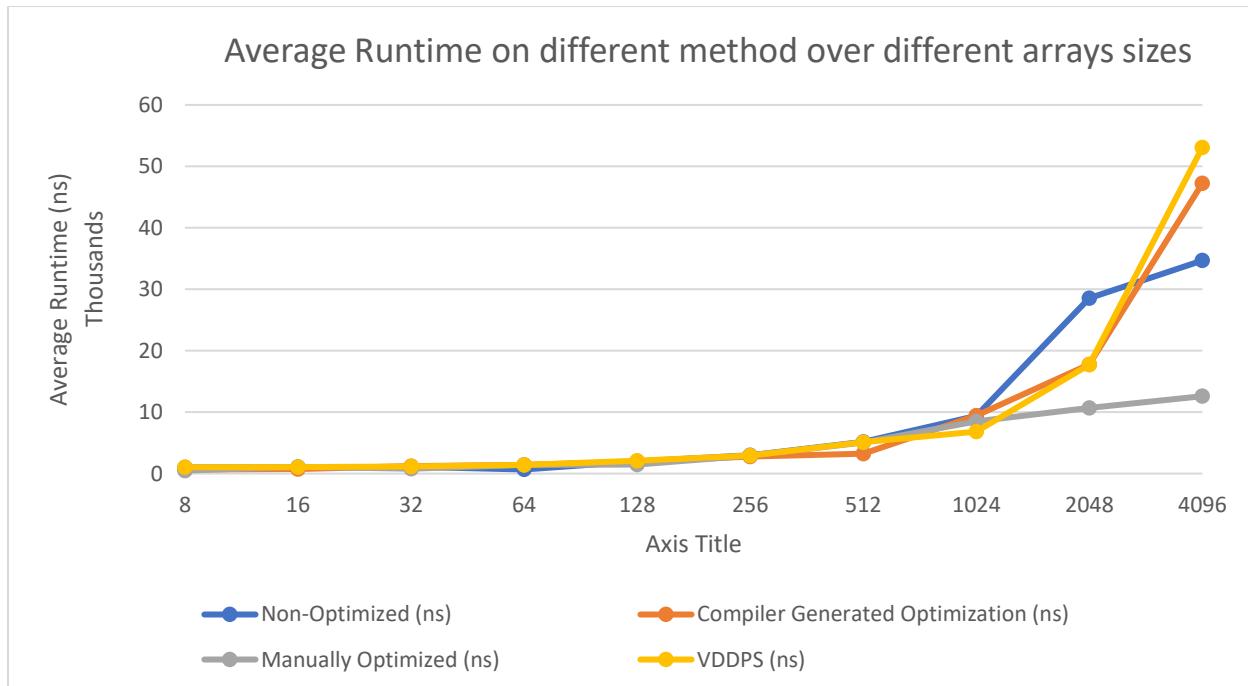


Figure 89: Average Runtime graph (all methods)

The more we optimized the code the more runtime decreased. We can see that our manually optimized code is executing faster than any other methods. Interestingly, VDPPS instruction is taking longer here than any other method. I am assuming because I used virtual box which has limited processor and depend on the parent computer, there is a drastic difference runtime.

If we compare both graphs from both environments how drastically optimizing code can reduce the runtime. It is more noticeable while working with larger array sizes. The manual optimized runtime changed more than other methods. The vector instruction, VDPPS made the program running faster.

Conclusion:

This take-home test taught me how to compute and demonstrate dot products using vector instructions and optimize the code by compiler generate in both Intel x86(MS VS) and Intel x64(Linux) environment. I also learned to use parallelization, vectorization to optimize the code. I also manually optimized the code. I learned to see how much optimal code matter while working with a large data set. I will remember than for my future code projects.