Name: Manasi. B. Kshirsagar

PRN: S17111009

Roll No.: 06

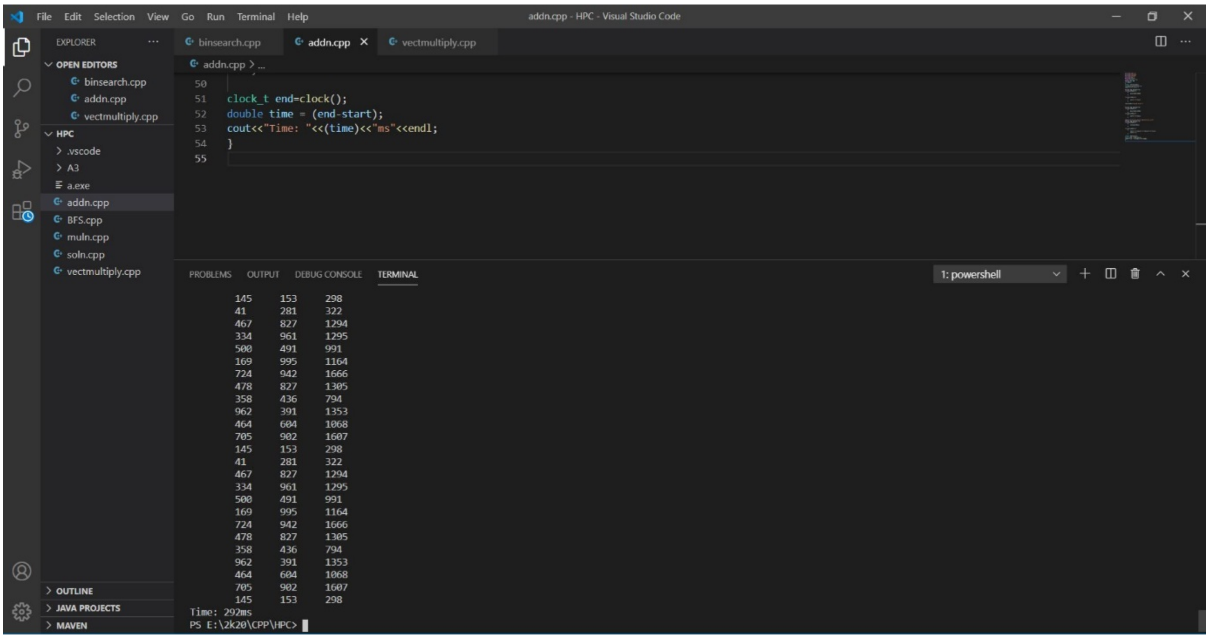
Class: BE Comp SS

Code (Addition of Two Large vectors):

```
#include<stdio.h>
#include<iostream>
#include<cstdlib>
#include<omp.h>
#include<time.h>
using namespace std;
#define MAX 100
int main()
{
clock_t start=clock();
int a[MAX],b[MAX],c[MAX],i;
cout<<"First Vector:";
#pragma omp parallel for
for(i=0;i<MAX;i++)
	{
		a[i]=rand()%1000;
	}
for(i=0;i<MAX;i++)
	{
        cout<<"\t"<<a[i];
    }

cout<<endl<<"Second Vector:";


#pragma omp parallel for
for(i=0;i<MAX;i++)
    {
        b[i]=rand()%1000;
    }
for(i=0;i<MAX;i++)
    {
        cout<<"\t"<<b[i];
    }

cout<<"\n Parallel-Vector Addition:(a,b,c)\t";
#pragma omp parallel for
for(i=0;i<MAX;i++)
    {
        c[i]=a[i]+b[i];
    }

for(i=0;i<MAX;i++)
    {
        cout<<"\t"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i];
```

```
                    cout<<"\n";
          }

clock_t end=clock();
double time =(end-start);
cout<<"Time: "<<(time)<<"ms"<<endl;
}
```

Output:



```
clock_t end=clock();
double time = (end-start);
cout<<"Time: "<<(time)<<"ms"<<endl;
}
```

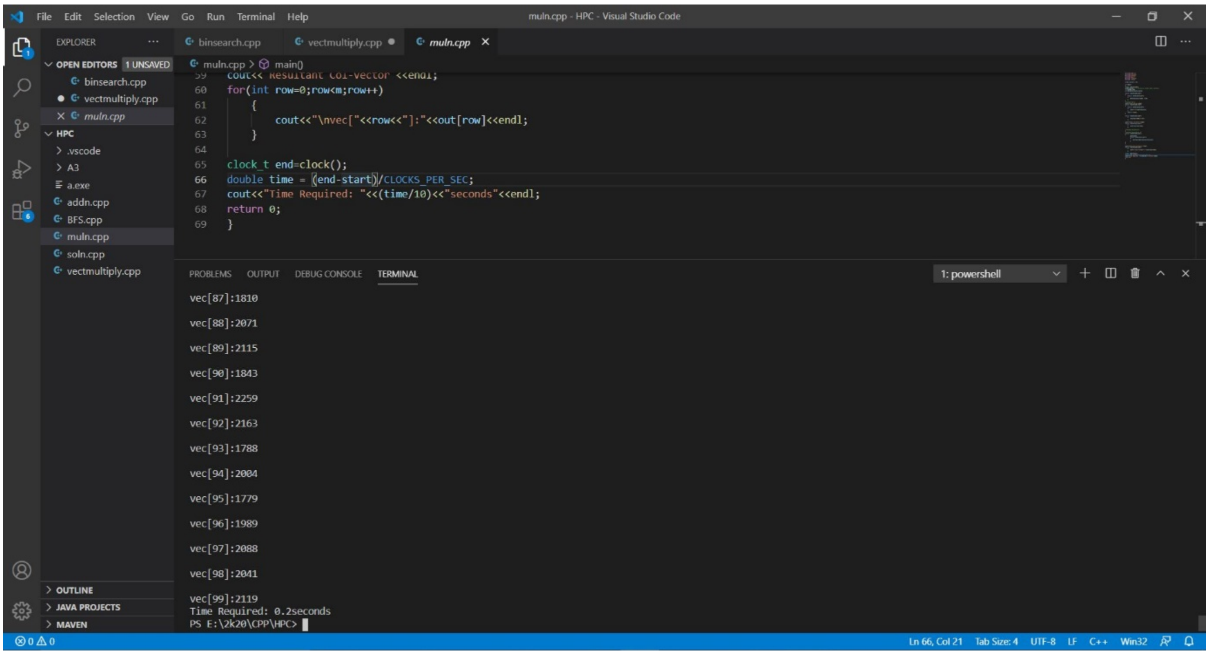
```
        145     153     298
        41      281     322
        467     827     1294
        334     961     1295
        500     491     991
        169     995     1164
        724     942     1666
        478     827     1305
        358     436     794
        962     391     1353
        464     604     1068
        705     902     1607
        145     153     298
        41      281     322
        467     827     1294
        334     961     1295
        500     491     991
        169     995     1164
        724     942     1666
        478     827     1305
        358     436     794
        962     391     1353
        464     604     1068
        705     902     1607
        145     153     298
Time: 292ms
PS E:\2k20\CPP\HPC> 
```

Code (Multiplication of two large arrays):

```
#include<stdio.h>
#include<iostream>
#include<cstdlib>
#include<omp.h>
#include <cstdlib>
#include <ctime>
using namespace std;
int main()
{
clock_t start=clock();
srand(time(0));   // Initialize random number generator.
int m=100,n=100;
int mat[m][n],vec[n],out[m];
```

```
for (int row=0; row<m;row++)
{
        for (int col=0; col<n;col++)
        {
                mat[row][col]=(rand () % 10);
        }
}
cout<<"Input Matrix"<<endl;
for (int row=0; row<m;row++)
{
        For (int col=0; col<n;col++)
        {
                cout<<"\t"<<mat[row][col];
        }
        cout<<""<<endl;
}

for(int row=0;row<n;row++)
        {
                vec[row]=(rand() % 10);
        }

cout<<"Input Col-Vector"<<endl;
for(int row=0;row<n;row++)
        {
                cout<<vec[row]<<endl;
        }

#pragma omp parallel
{
#pragma omp parallel for
For (int row=0;row<m;row++)
{
                out[row]=0;
                for(int col=0;col<n;col++)
                {
                        out[row]+=mat[row][col]*vec[col];
                }
}
}
cout<<"Resultant Col-Vector"<<endl;
for(int row=0;row<m;row++)
        {
                cout<<"\nvec["<<row<<"]:"<<out[row]<<endl;
        }
clock_t end=clock();
double time = (end-start)/CLOCKS_PER_SEC;
cout<<"Time Required: "<<(time/10)<<"seconds"<<endl;
return 0;
}
```

Output:



```
cout<<"Resultant Col-vector"<<endl;
for(int row=0;row<m;row++)
    {
        cout<<"\nvec["<<row<<"]:"<<out[row]<<endl;
    }

clock_t end=clock();
double time = (end-start)/CLOCKS_PER_SEC;
cout<<"Time Required: "<<(time/10)<<"seconds"<<endl;
return 0;
}
```

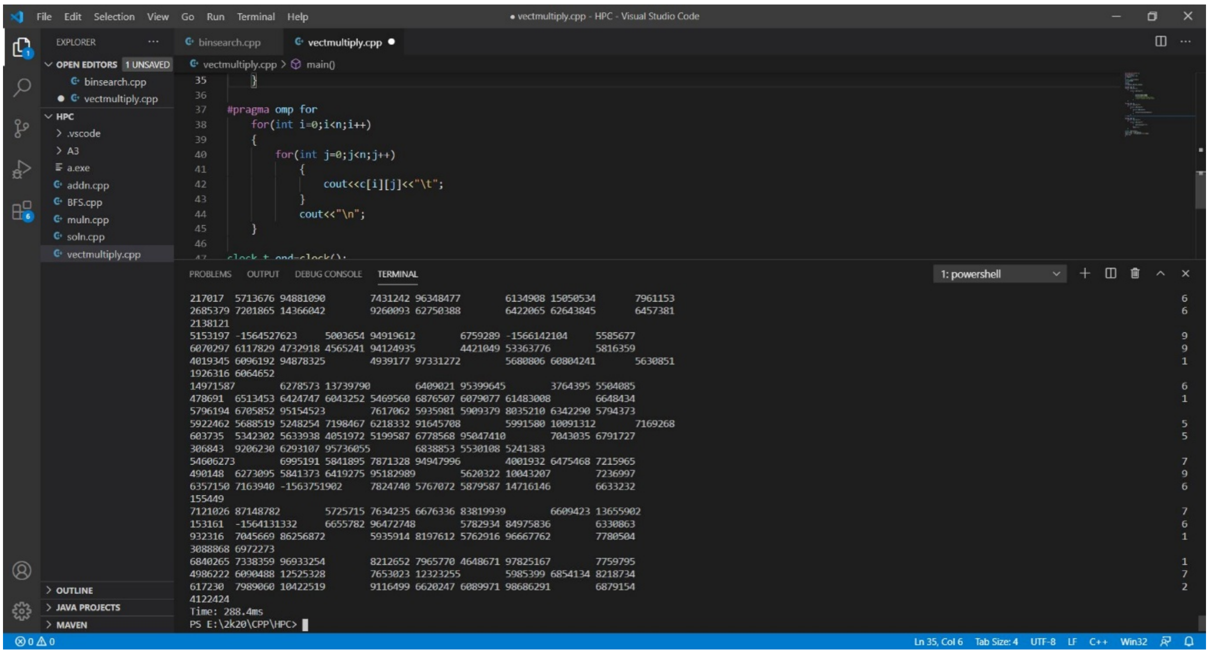
```
vec[87]:1810

vec[88]:2071

vec[89]:2115

vec[90]:1843

vec[91]:2259

vec[92]:2163

vec[93]:1788

vec[94]:2004

vec[95]:1779

vec[96]:1989

vec[97]:2088

vec[98]:2041

vec[99]:2119
Time Required: 0.2seconds
PS E:\2k20\CPP\HPC>
```

Code (Multiplication of vector & matrix):

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

#include<omp.h>

using namespace std;

int main()

{

clock_t start=clock();

int i,thread;

int n;

cin>>n;

int a[n][n],b[n][n],c[n][n];


#pragma omp for
```

```
for(int i=0;i<n;i++)
        {
                for(int j=0;j<n;j++)
                        {

                                a[i][j]=rand()%1000;
                                b[i][j]=rand()%1000;
                                //thread=omp_get_thread_num();
                                //cout<<"thread"<<thread<<endl;
                        }
        }

#pragma omp for
        for(int i=0;i<n;i++)
        {
                for(int j=0;j<n;j++)
                {
                        for(int k=0;k<n;k++)
                        {
                                c[i][j]+=(a[i][k]*b[k][j]);
                        }
                }
        }

#pragma omp for
        for (int i=0; i<n;i++)
        {
                for (int j=0; j<n;j++)
                        {
                                cout<<c[i][j] <<"\t";
                        }
                        cout<<"\n";
        }
clock_t end=clock();
double time =(end-start);
cout<<"Time: "<<(time/10)<<"ms"<<endl;
return 0;
}
```

Output:



```
    }

#pragma omp for
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
            {
                cout<<c[i][j]<<"\t";
            }
            cout<<"\n";
    }

clock_t end=clock();
```

```
217017  5713676 94881090        7431242 96348477        6134908 15050534        7961153          6
2685379 7201865 14366042        9260093 62750388        6422065 62643845        6457381          6
2138121
5153197 -1564527623     5003654 94919612        6759289 -1566142104     5585677                  9
6070297 6117829 4732918 4565241 94124935        4421049 53363776        5816359                  9
4019345 6096192 94878325        4939177 97331272        5680806 60804241        5630851          1
1926316 6064652
14971587        6278573 13739790        6409021 95399645        3764395 5504085                  6
478691  6513453 6424747 6043252 5469560 6876507 6079077 61483008        6648434                  1
5796194 6705852 95154523        7617062 5935981 5909379 8035210 6342290 5794373
5922462 5688519 5248254 7198467 6218332 91645708        5991580 10091312        7169268          5
603735  5342302 5633938 4051972 5199587 6778568 95047410        7043035 6791727                  5
306843  9206230 6293107 95736055        6838853 5530108 5241383
54606273        6995191 5841895 7871328 94947996        4001932 6475468 7215965                  7
490148  6273095 5841373 6419275 95182989        5620322 10043207        7236997                  9
6357150 7163940 -1563751902     7824740 5767072 5879587 14716146        6633232                  6
155449
7121026 87148782        5725715 7634235 6676336 83819939        6609423 13655902                 7
153161  -1564131332     6655782 96472748        5782934 84975836        6330863                  6
932316  7045669 86256872        5935914 8197612 5762916 96667762        7780504                  1
3088868 6972273
6840265 7338359 96933254        8212652 7965770 4648671 97825167        7759795                  1
4986222 6090488 12525328        7653023 12323255        5985399 6854134 8218734                  7
617230  7989060 10422519        9116499 6620247 6089971 98686291        6879154                  2
4122424
Time: 288.4ms
PS E:\2k20\CPP\HPC>
```