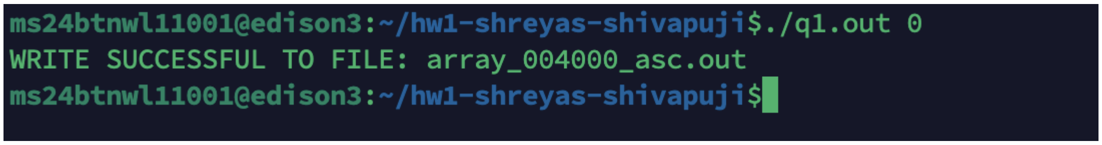
ME5470: Introduction to Parallel Scientific Computing

Homework 1

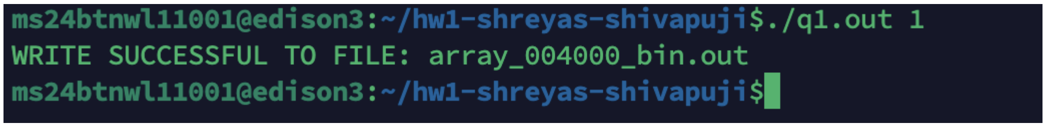
Shreyas A Shivapuji
MS24BTNWL11001

Question 1(a)

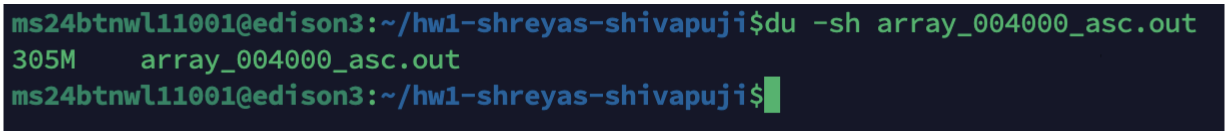
The ASCII file is of **305 MB** in size and the Binary file is of **123 MB**. The screenshots showing the size of the files are attached below:



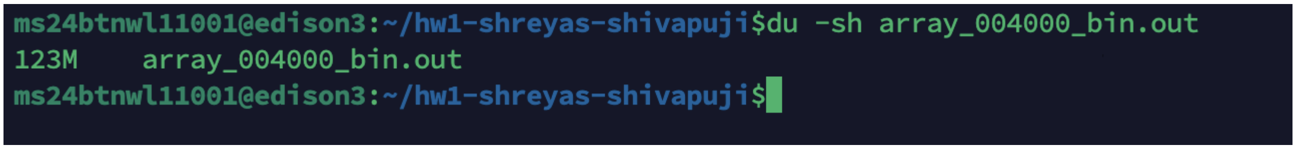
```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q1.out 0
WRITE SUCCESSFUL TO FILE: array_004000_asc.out
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```



```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q1.out 1
WRITE SUCCESSFUL TO FILE: array_004000_bin.out
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```



```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$du -sh array_004000_asc.out
305M    array_004000_asc.out
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```



```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$du -sh array_004000_bin.out
123M    array_004000_bin.out
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```

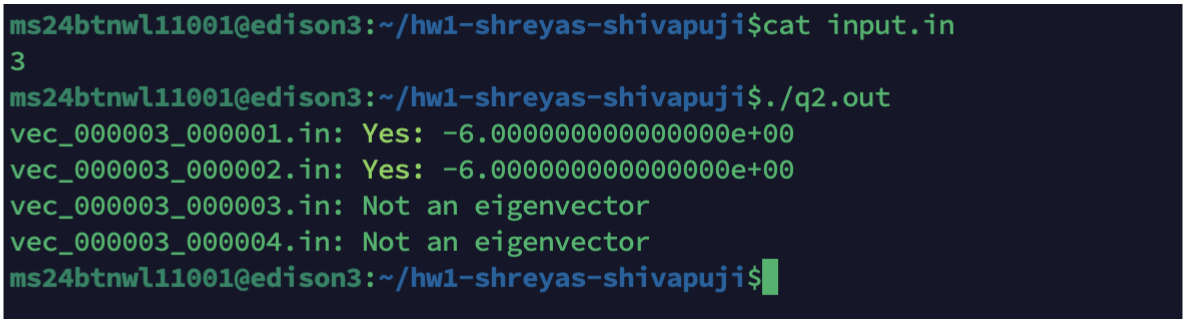
Question 1(b)

For $n = 4000$, the number of double-valued elements in the array are **16000000**. Therefore, the total size of the array $= 8 * 16000000 =$ **128 MB**, which is very close to the size of the binary output file "array_004000_bin.out" **(123 MB)**. The ASCII file is larger than the binary file because the full representation of a double in ASCII representation is 20 bytes, while it is of 8 bytes in binary format. Therefore, binary format is more effective to store files with larger data.

Question 2

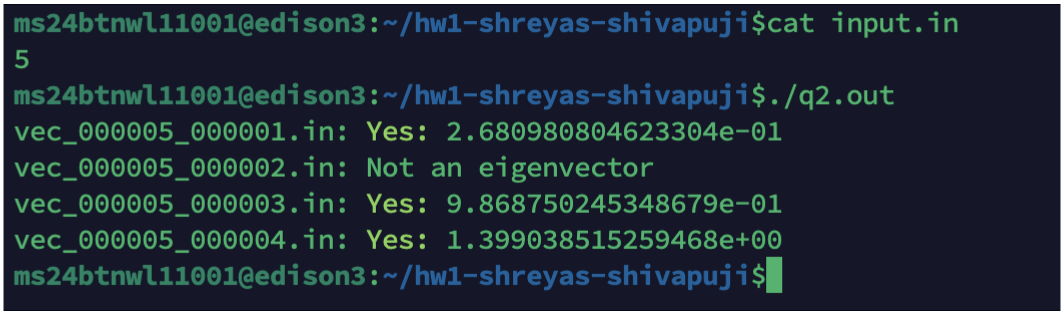
The code was implemented using 4 functions: **read_matrix()**, **read_vector()**, **is_eigen()** and **write_eigen()**, which were called from the main function to execute their respective functionalities. The screenshots of the tests ran on the code, with the eigenvalue results in each of the tests are attached below:

For **n** = **3**:



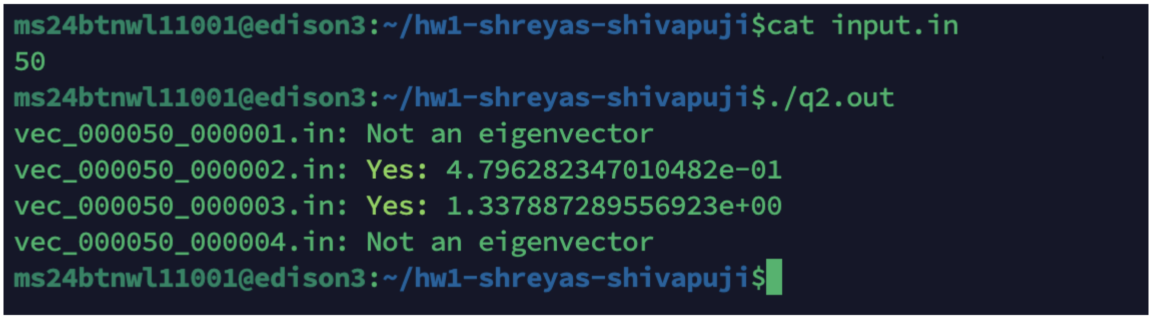
```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$cat input.in
3
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q2.out
vec_000003_000001.in: Yes: -6.000000000000000e+00
vec_000003_000002.in: Yes: -6.000000000000000e+00
vec_000003_000003.in: Not an eigenvector
vec_000003_000004.in: Not an eigenvector
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```

For **n** = **5**:



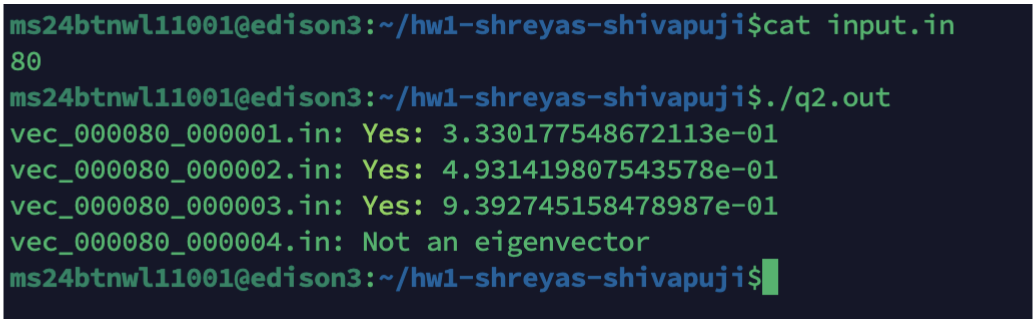
```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$cat input.in
5
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q2.out
vec_000005_000001.in: Yes: 2.680980804623304e-01
vec_000005_000002.in: Not an eigenvector
vec_000005_000003.in: Yes: 9.868750245348679e-01
vec_000005_000004.in: Yes: 1.399038515259468e+00
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```

For **n** = **50**:



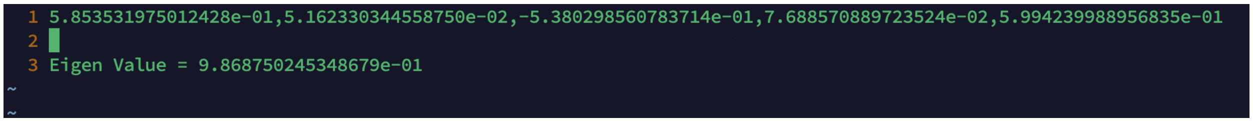
```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$cat input.in
50
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q2.out
vec_000050_000001.in: Not an eigenvector
vec_000050_000002.in: Yes: 4.796282347010482e-01
vec_000050_000003.in: Yes: 1.337887289556923e+00
vec_000050_000004.in: Not an eigenvector
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```

For **n** = **80**:



```
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$cat input.in
80
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$./q2.out
vec_000080_000001.in: Yes: 3.330177548672113e-01
vec_000080_000002.in: Yes: 4.931419807543578e-01
vec_000080_000003.in: Yes: 9.392745158478987e-01
vec_000080_000004.in: Not an eigenvector
ms24btnwl11001@edison3:~/hw1-shreyas-shivapuji$
```

The eigen values of a vector were also successfully appended to their respective files. Below is a screenshot which demonstrates this in one of the file “vec_000005_000001.in” :



```
5.853531975012428e-01,5.162330344558750e-02,-5.380298560783714e-01,7.688570889723524e-02,5.994239988956835e-01

Eigen Value = 9.868750245348679e-01
~
~
```