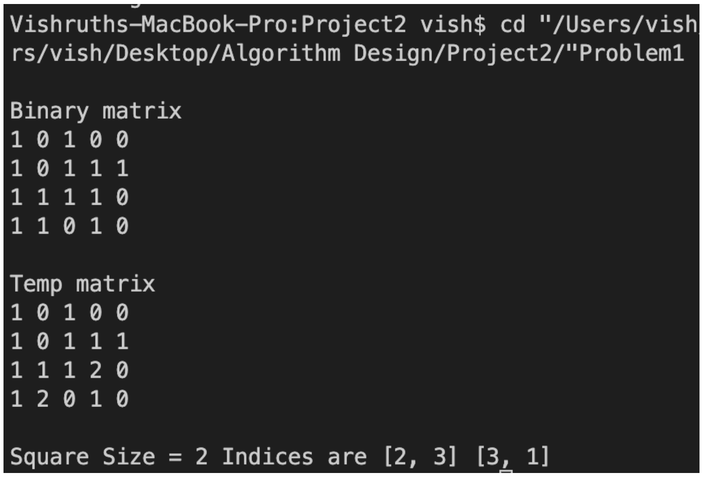
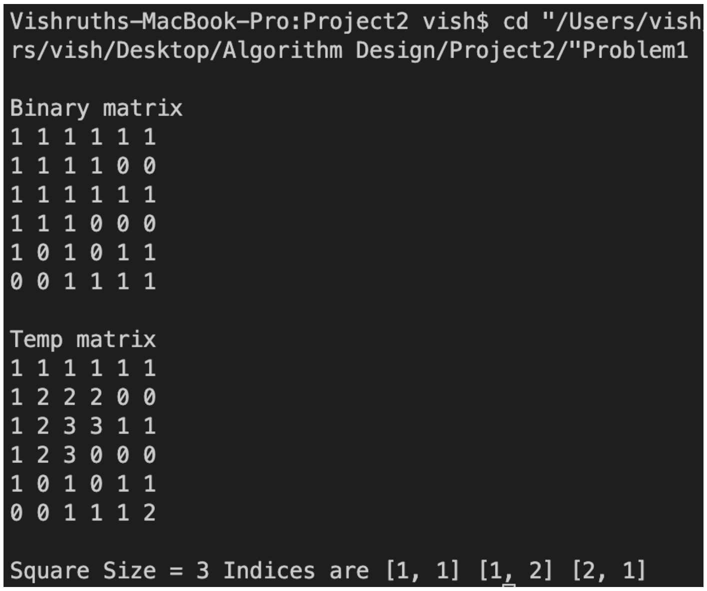
Problem 1

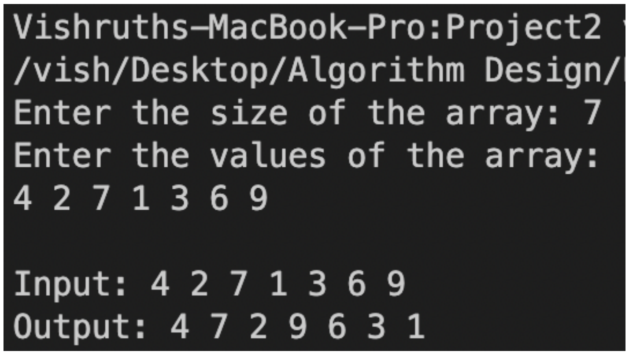


```
Vishruths-MacBook-Pro:Project2 vish$ cd "/Users/vish
rs/vish/Desktop/Algorithm Design/Project2/"Problem1

Binary matrix
1 0 1 0 0
1 0 1 1 1
1 1 1 1 0
1 1 0 1 0

Temp matrix
1 0 1 0 0
1 0 1 1 1
1 1 1 2 0
1 2 0 1 0

Square Size = 2 Indices are [2, 3] [3, 1]
```

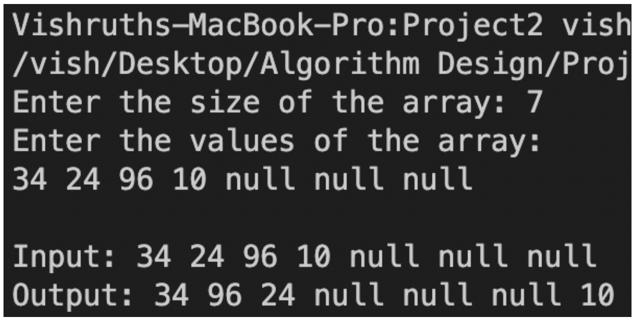


```
Vishruths-MacBook-Pro:Project2 vish$ cd "/Users/vish
rs/vish/Desktop/Algorithm Design/Project2/"Problem1

Binary matrix
1 1 1 1 1 1
1 1 1 1 0 0
1 1 1 1 1 1
1 1 1 0 0 0
1 0 1 0 1 1
0 0 1 1 1 1

Temp matrix
1 1 1 1 1 1
1 2 2 2 0 0
1 2 3 3 1 1
1 2 3 0 0 0
1 0 1 0 1 1
0 0 1 1 1 2

Square Size = 3 Indices are [1, 1] [1, 2] [2, 1]
```

Problem 2A

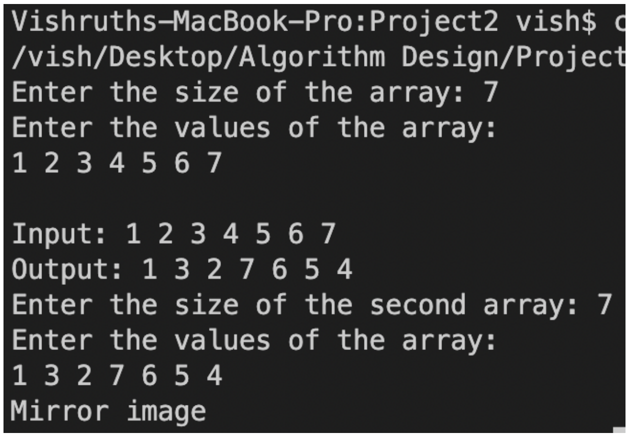


```
Vishruths-MacBook-Pro:Project2
/vish/Desktop/Algorithm Design/
Enter the size of the array: 7
Enter the values of the array:
4 2 7 1 3 6 9

Input: 4 2 7 1 3 6 9
Output: 4 7 2 9 6 3 1
```

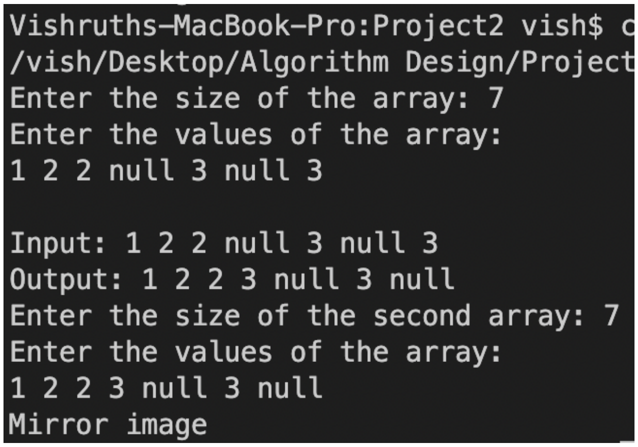


```
Vishruths-MacBook-Pro:Project2 vish
/vish/Desktop/Algorithm Design/Proj
Enter the size of the array: 7
Enter the values of the array:
34 24 96 10 null null null

Input: 34 24 96 10 null null null
Output: 34 96 24 null null null 10
```

Problem 2B

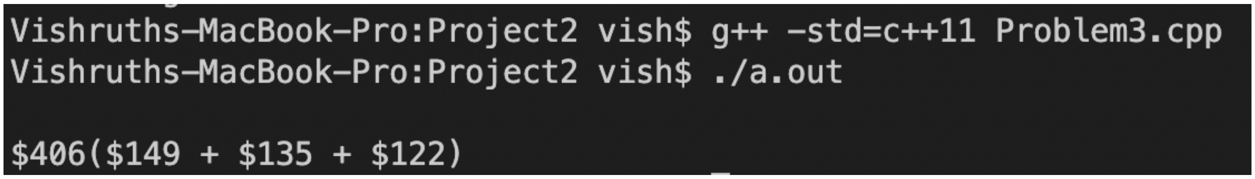


```
Vishruths-MacBook-Pro:Project2 vish$ c
/vish/Desktop/Algorithm Design/Project
Enter the size of the array: 7
Enter the values of the array:
1 2 3 4 5 6 7

Input: 1 2 3 4 5 6 7
Output: 1 3 2 7 6 5 4
Enter the size of the second array: 7
Enter the values of the array:
1 3 2 7 6 5 4
Mirror image
```

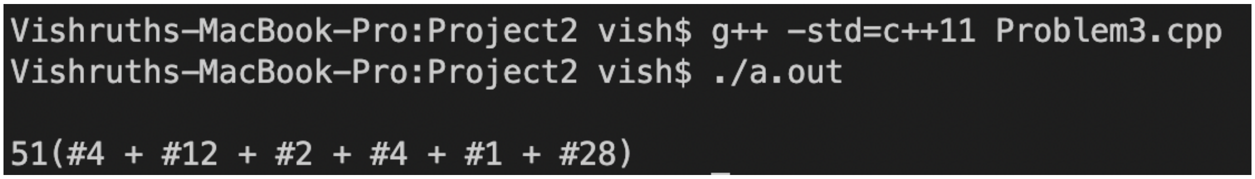


```
Vishruths-MacBook-Pro:Project2 vish$ c
/vish/Desktop/Algorithm Design/Project
Enter the size of the array: 7
Enter the values of the array:
1 2 2 null 3 null 3

Input: 1 2 2 null 3 null 3
Output: 1 2 2 3 null 3 null
Enter the size of the second array: 7
Enter the values of the array:
1 2 2 3 null 3 null
Mirror image
```

Problem 3



```
Vishruths-MacBook-Pro:Project2 vish$ g++ -std=c++11 Problem3.cpp
Vishruths-MacBook-Pro:Project2 vish$ ./a.out

$406($149 + $135 + $122)
```



```
Vishruths-MacBook-Pro:Project2 vish$ g++ -std=c++11 Problem3.cpp
Vishruths-MacBook-Pro:Project2 vish$ ./a.out

51(#4 + #12 + #2 + #4 + #1 + #28)
```

Theoretical Time Complexity Analysis:

Consider the following function in Problem 3:

```
void HungarianAlgorithm::step3(int *assignment, double *distMatrix, bool *starMatrix, bool *newStarMatrix, bool *primeMatrix, bool *coveredColumns, bool *coveredRows, int nOfRows, int nOfColumns, int minDim)
{
   bool zerosFound;
   int row, col, starCol;

   zerosFound = true;
   while (zerosFound)
   {
       zerosFound = false;
       for (col = 0; col<nOfColumns; col++)
           if (!coveredColumns[col])
               for (row = 0; row<nOfRows; row++)
                   if ((!coveredRows[row]) && (fabs(distMatrix[row + nOfRows*col]) < DBL_EPSILON))
                   {
                       primeMatrix[row + nOfRows*col] = true;

                       for (starCol = 0; starCol<nOfColumns; starCol++)
                           if (starMatrix[row + nOfRows*starCol])
                               break;

                       if (starCol == nOfColumns)
                       {
                           step4(assignment, distMatrix, starMatrix, newStarMatrix, primeMatrix, coveredColumns, coveredRows, nOfRows, nOfColumns, minDim, row, col);
                           return;
                       }
                       else
                       {
                           coveredRows[row] = true;
                           coveredColumns[starCol] = false;
                           zerosFound = true;
                           break;
                       }
                   }
   }

   step5(assignment, distMatrix, starMatrix, newStarMatrix, primeMatrix,
coveredColumns, coveredRows, nOfRows, nOfColumns, minDim);
}
```

In this function, we can see that the while loop will run until zerosFound becomes false.
So, the worst case time complexity for this is O(n).
This while loop has a for loop nested in it which will run from 0 to n = noOfColumns. So, the worst case time complexity until this segment is O(n*n).
This for loop has another for loop nested in it which will run from 0 to n = noOfRows. So, the worst case time complexity until this segment is O(n*n*n).
Further, this for loop has another for loop nested in it which will run from 0 to n = noOfColumns. So, the worst case time complexity until this segment is O(n*n*n*n).

Therefore, the theoretical time complexity for this program is Big O(n^4).