

NAME-HIMA SAMEERA.N

ROLL-21071A6790

1) Given a row wise sorted matrix of size **R*C** where R and C are always **odd**, find the median of the matrix.

```
#include <iostream>

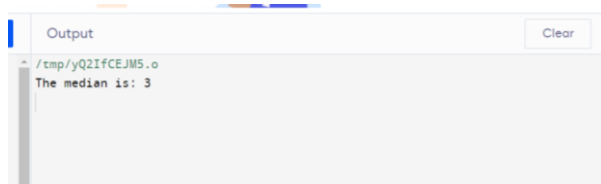
using namespace std;

int Findmedian(int arr[3][3], int row, int col)
{
    int median[row * col];
    int index = 0;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            median[index] = arr[i][j];
            index++;
        }
    }
    return median[(row * col) / 2];
}

int main()
{
    int row = 3, col = 3;
    int arr[3][3] = {{1, 3, 8},
                     {2, 3, 4},
                     {1, 2, 5}};

    cout << "The median is: " << Findmedian
    (arr, row, col) << endl;

    return 0;
}
```

A screenshot of a code editor's output window. The window has a title bar with 'Output' and a 'Clear' button. The output text shows the command prompt path '/tmp/yQ2IfCEJM5.o' followed by the program's output 'The median is: 3'.

2) Given the arrival and departure times of all trains that reach a railway station, the task is to find the minimum number of platforms required for the railway station so that no train waits. We are given two arrays that represent the arrival and departure times of trains that stop.

```
def findPlatform(arr, dep, n):  
    plat_needed = 1  
    result = 1  
    for i in range(n)  
    plat_needed = 1  
    for j in range(n)  
    if i != j:  
    if (arr[i] >= arr[j] and dep[j] >= arr[i]):  
    plat_needed += 1  
    result = max(result, plat_needed)  
    return result  
  
# Driver code  
def main():  
    arr = [100, 300, 500]  
    dep = [900, 400, 600]  
    n = len(arr)  
    print("{}".format(findPlatform(arr, dep, n)))  
  
if __name__ == '__main__':  
    main()  
  
output:
```

Shell

2

> |