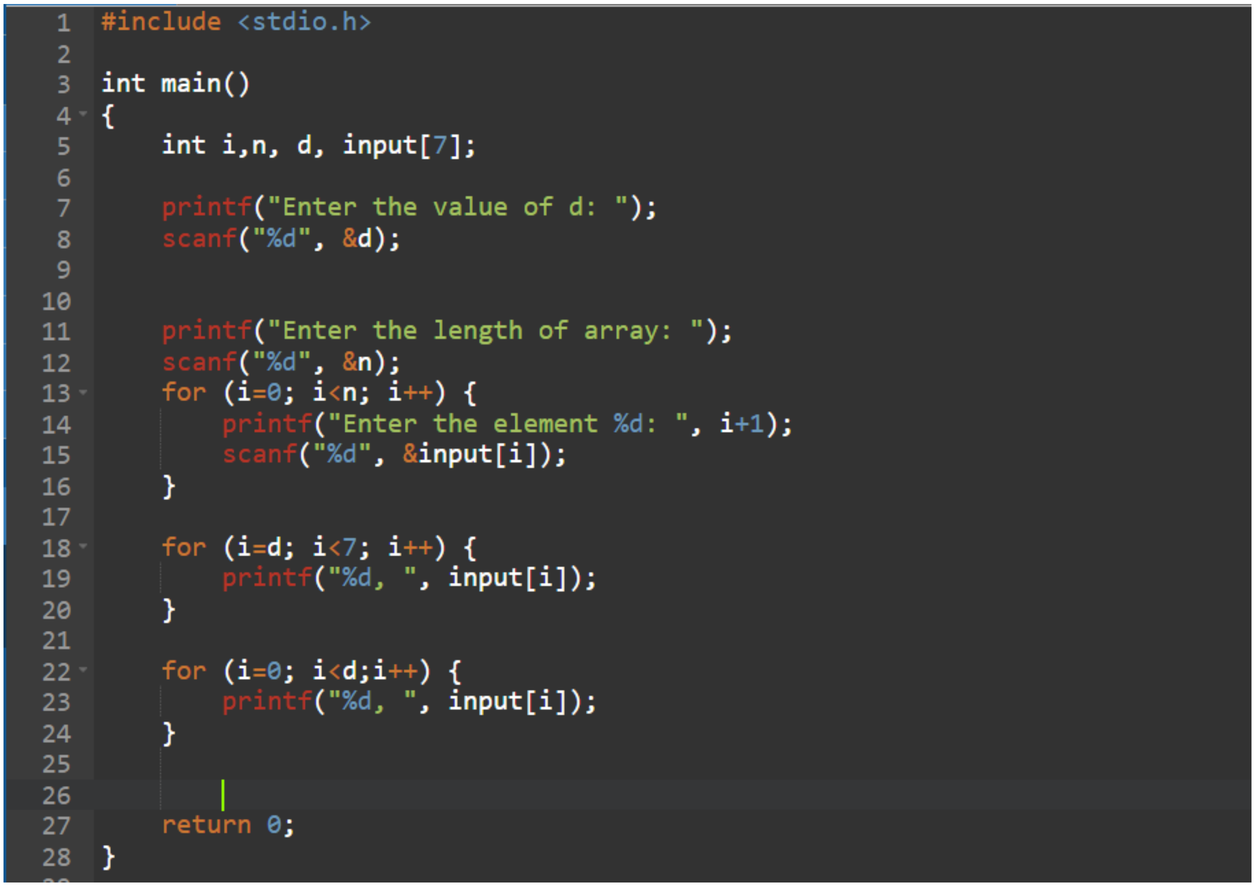
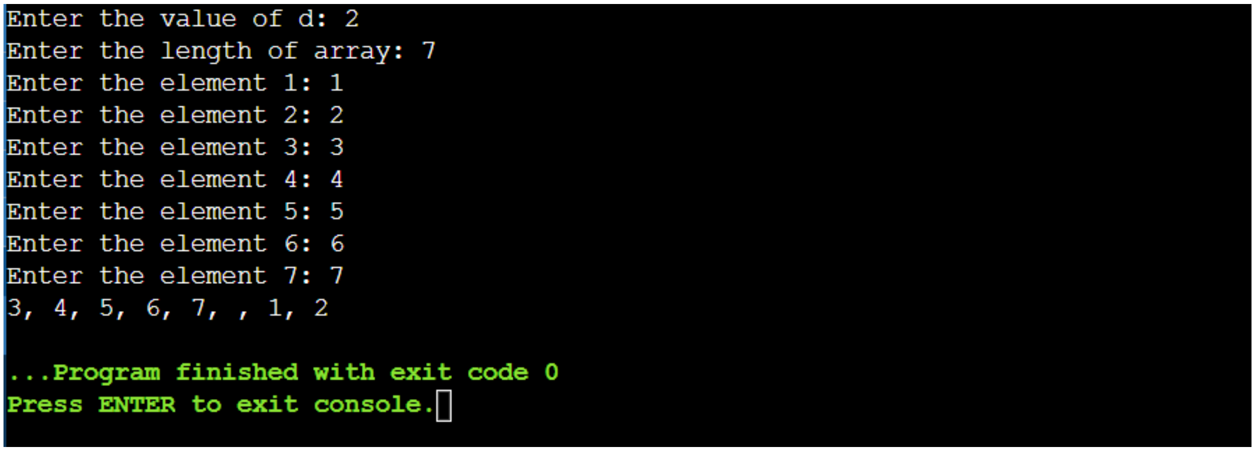
PF LAB 07

TASK 02:

Note: (C File is not available because I forgot to copy them at the library and they have been deleted now.)



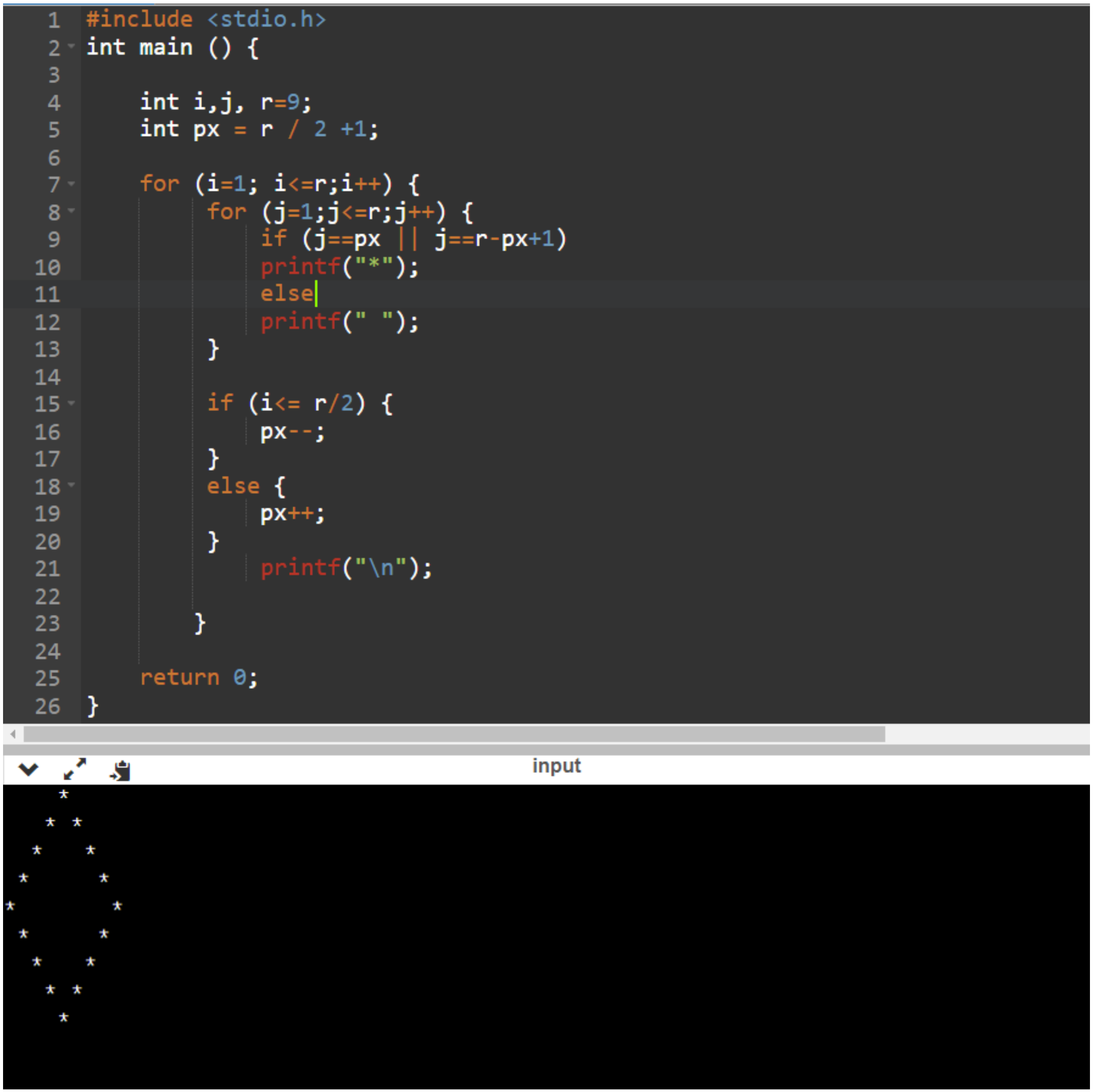
```c
#include <stdio.h>

int main()
{
    int i,n, d, input[7];

    printf("Enter the value of d: ");
    scanf("%d", &d);


    printf("Enter the length of array: ");
    scanf("%d", &n);
    for (i=0; i<n; i++) {
        printf("Enter the element %d: ", i+1);
        scanf("%d", &input[i]);
    }

    for (i=d; i<7; i++) {
        printf("%d, ", input[i]);
    }

    for (i=0; i<d;i++) {
        printf("%d, ", input[i]);
    }


    return 0;
}
```



```
Enter the value of d: 2
Enter the length of array: 7
Enter the element 1: 1
Enter the element 2: 2
Enter the element 3: 3
Enter the element 4: 4
Enter the element 5: 5
Enter the element 6: 6
Enter the element 7: 7
3, 4, 5, 6, 7, , 1, 2

...Program finished with exit code 0
Press ENTER to exit console.
```

TASK 05:

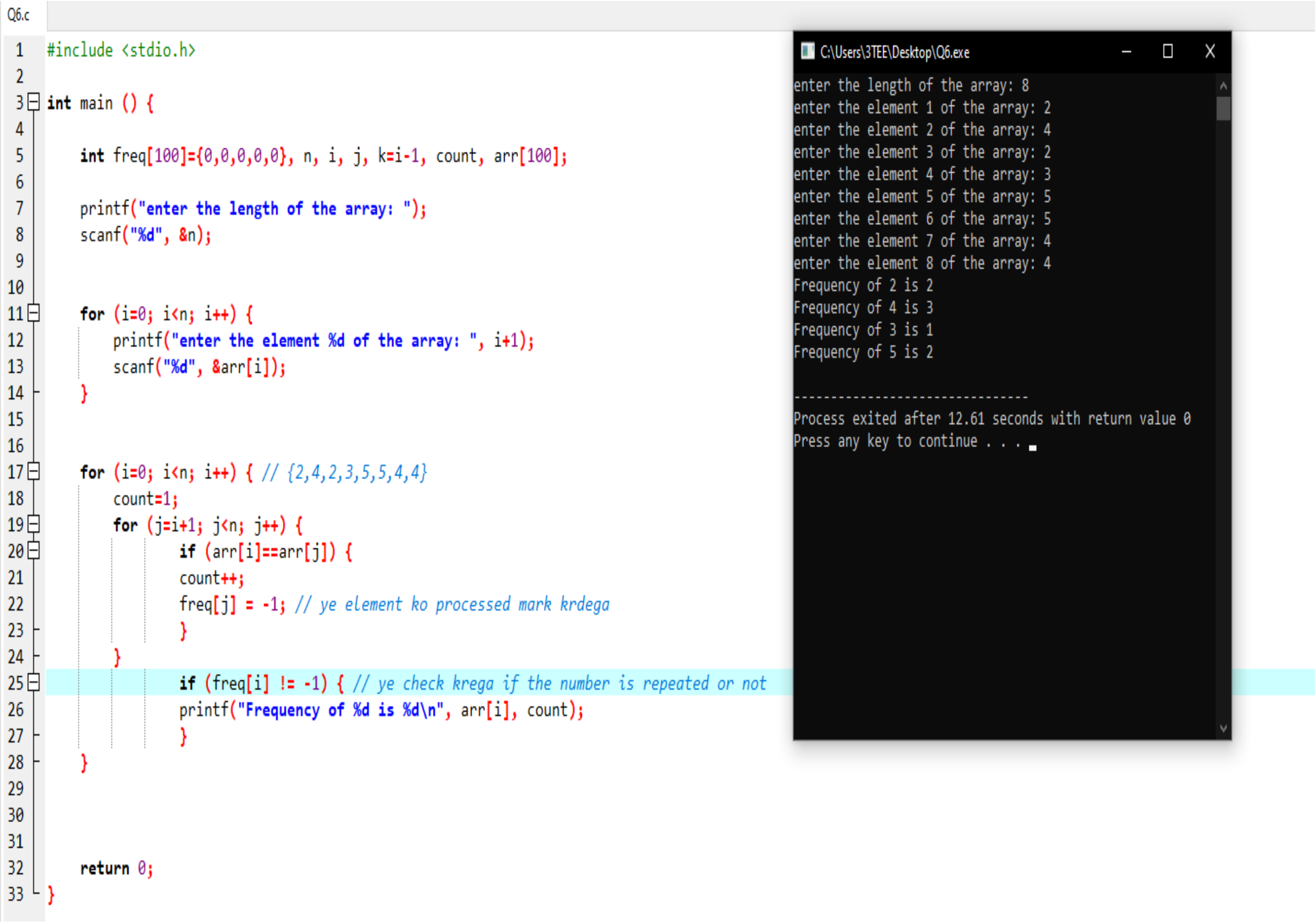
Note: (C File is not available because I forgot to copy them at the library and they have been deleted now.)



```c
#include <stdio.h>
int main () {

    int i,j, r=9;
    int px = r / 2 +1;

    for (i=1; i<=r;i++) {
        for (j=1;j<=r;j++) {
            if (j==px || j==r-px+1)
            printf("*");
            else
            printf(" ");
        }

        if (i<= r/2) {
            px--;
        }
        else {
            px++;
        }
            printf("\n");

       }

    return 0;
}
```

```
    *
   * *
  *   *
 *     *
*       *
 *     *
  *   *
   * *
    *
```

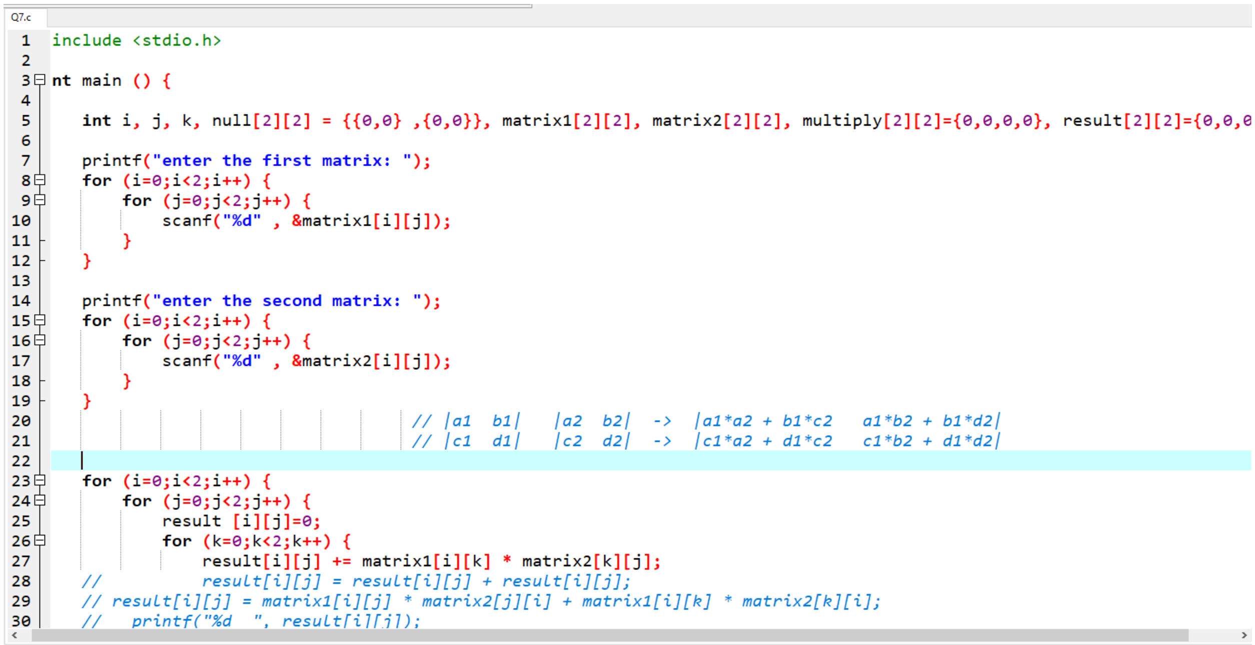
TASK 06:



```c
#include <stdio.h>

int main () {

    int freq[100]={0,0,0,0,0}, n, i, j, k=i-1, count, arr[100];

    printf("enter the length of the array: ");
    scanf("%d", &n);


    for (i=0; i<n; i++) {
        printf("enter the element %d of the array: ", i+1);
        scanf("%d", &arr[i]);
    }


    for (i=0; i<n; i++) { // {2,4,2,3,5,5,4,4}
        count=1;
        for (j=i+1; j<n; j++) {
            if (arr[i]==arr[j]) {
            count++;
            freq[j] = -1; // ye element ko processed mark krdega
            }
        }
            if (freq[i] != -1) { // ye check krega if the number is repeated or not
            printf("Frequency of %d is %d\n", arr[i], count);
            }
    }



    return 0;
}
```

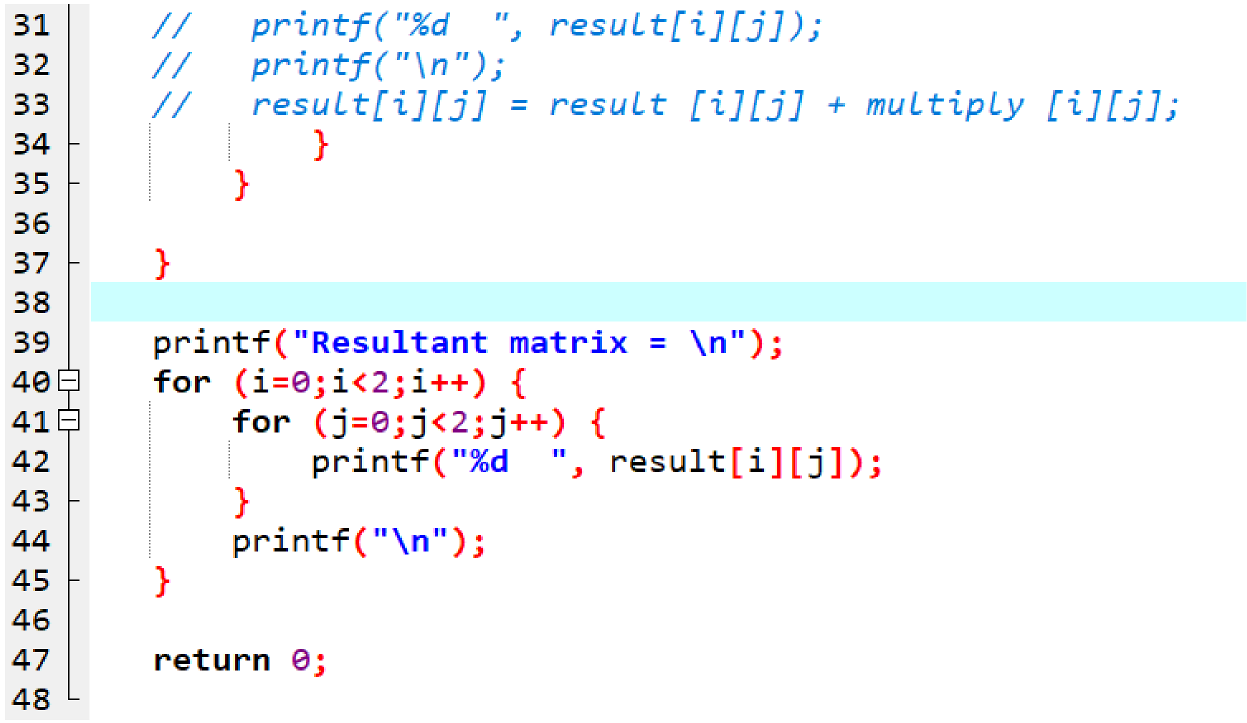
```
enter the length of the array: 8
enter the element 1 of the array: 2
enter the element 2 of the array: 4
enter the element 3 of the array: 2
enter the element 4 of the array: 3
enter the element 5 of the array: 5
enter the element 6 of the array: 5
enter the element 7 of the array: 4
enter the element 8 of the array: 4
Frequency of 2 is 2
Frequency of 4 is 3
Frequency of 3 is 1
Frequency of 5 is 2

--------------------------------
Process exited after 12.61 seconds with return value 0
Press any key to continue . . .
```

TASK 07:

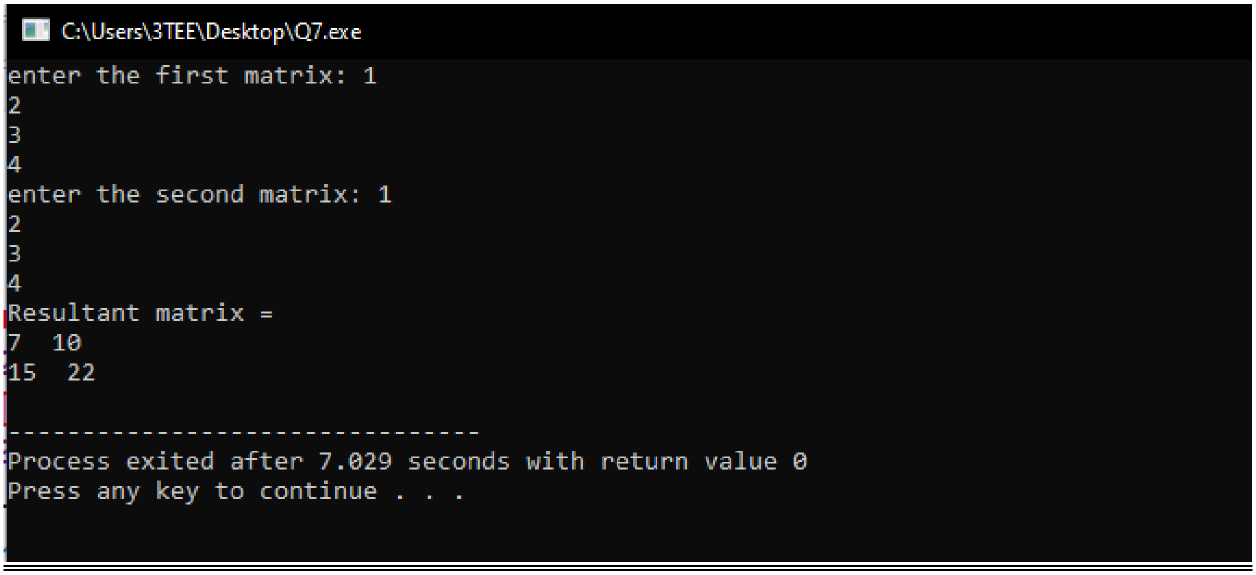


Q7.c

```
include <stdio.h>

nt main () {

    int i, j, k, null[2][2] = {{0,0} ,{0,0}}, matrix1[2][2], matrix2[2][2], multiply[2][2]={0,0,0,0}, result[2][2]={0,0,0

    printf("enter the first matrix: ");
    for (i=0;i<2;i++) {
        for (j=0;j<2;j++) {
            scanf("%d" , &matrix1[i][j]);
        }
    }

    printf("enter the second matrix: ");
    for (i=0;i<2;i++) {
        for (j=0;j<2;j++) {
            scanf("%d" , &matrix2[i][j]);
        }
    }
                                    // |a1  b1|    |a2  b2|  ->  |a1*a2 + b1*c2    a1*b2 + b1*d2|
                                    // |c1  d1|    |c2  d2|  ->  |c1*a2 + d1*c2    c1*b2 + d1*d2|

    for (i=0;i<2;i++) {
        for (j=0;j<2;j++) {
            result [i][j]=0;
            for (k=0;k<2;k++) {
                result[i][j] += matrix1[i][k] * matrix2[k][j];
    //          result[i][j] = result[i][j] + result[i][j];
    // result[i][j] = matrix1[i][j] * matrix2[j][i] + matrix1[i][k] * matrix2[k][i];
    //   printf("%d  ", result[i][j]);
```

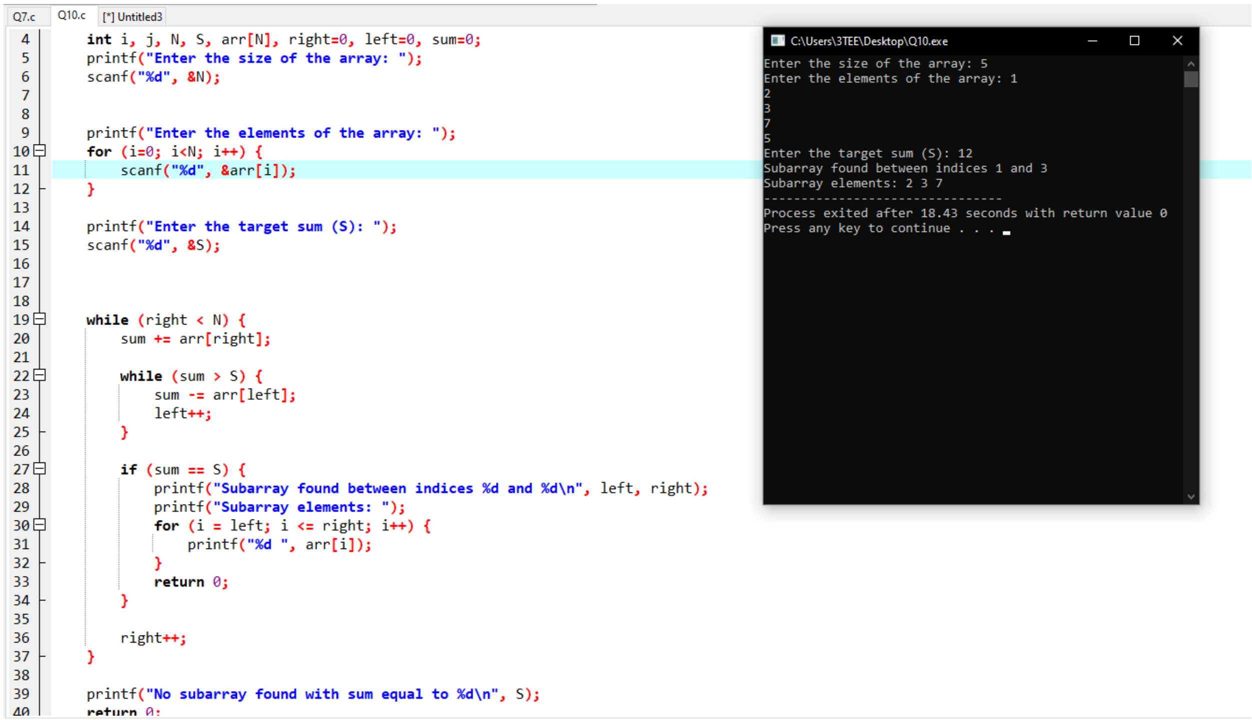


```
    //   printf("%d  ", result[i][j]);
    //   printf("\n");
    //   result[i][j] = result [i][j] + multiply [i][j];
                }
            }

    }

    printf("Resultant matrix = \n");
    for (i=0;i<2;i++) {
        for (j=0;j<2;j++) {
            printf("%d  ", result[i][j]);
        }
        printf("\n");
    }

    return 0;
```



```
enter the first matrix: 1
2
3
4
enter the second matrix: 1
2
3
4
Resultant matrix =
7  10
15  22

--------------------------------
Process exited after 7.029 seconds with return value 0
Press any key to continue . . .
```

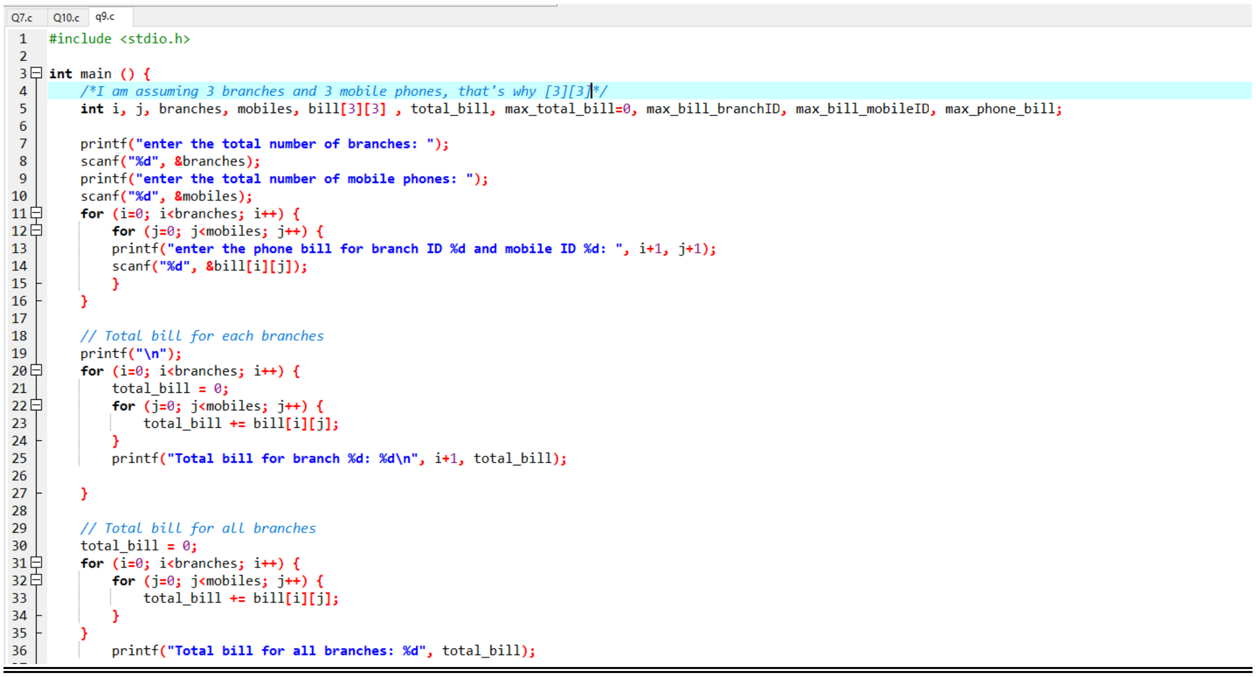
TASK 10:



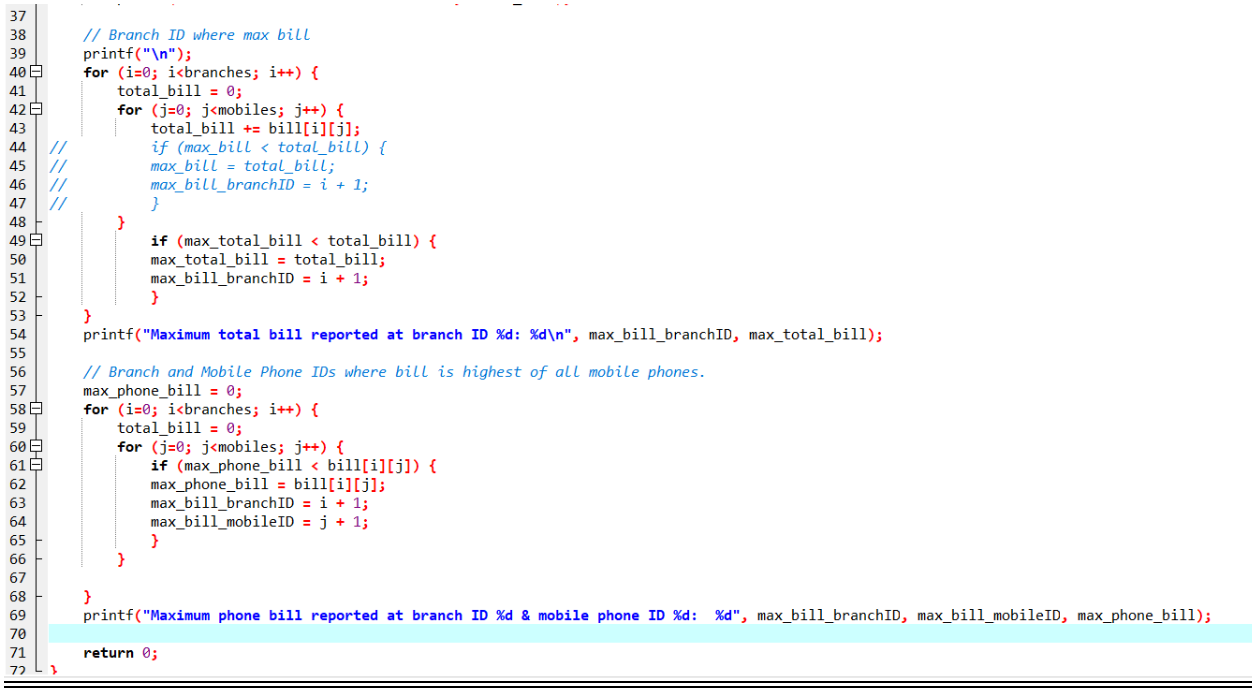
```c
    int i, j, N, S, arr[N], right=0, left=0, sum=0;
    printf("Enter the size of the array: ");
    scanf("%d", &N);


    printf("Enter the elements of the array: ");
    for (i=0; i<N; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Enter the target sum (S): ");
    scanf("%d", &S);



    while (right < N) {
        sum += arr[right];

        while (sum > S) {
            sum -= arr[left];
            left++;
        }

        if (sum == S) {
            printf("Subarray found between indices %d and %d\n", left, right);
            printf("Subarray elements: ");
            for (i = left; i <= right; i++) {
                printf("%d ", arr[i]);
            }
            return 0;
        }

        right++;
    }

    printf("No subarray found with sum equal to %d\n", S);
    return 0;
```

```
Enter the size of the array: 5
Enter the elements of the array: 1
2
3
7
5
Enter the target sum (S): 12
Subarray found between indices 1 and 3
Subarray elements: 2 3 7
--------------------------------
Process exited after 18.43 seconds with return value 0
Press any key to continue . . .
```

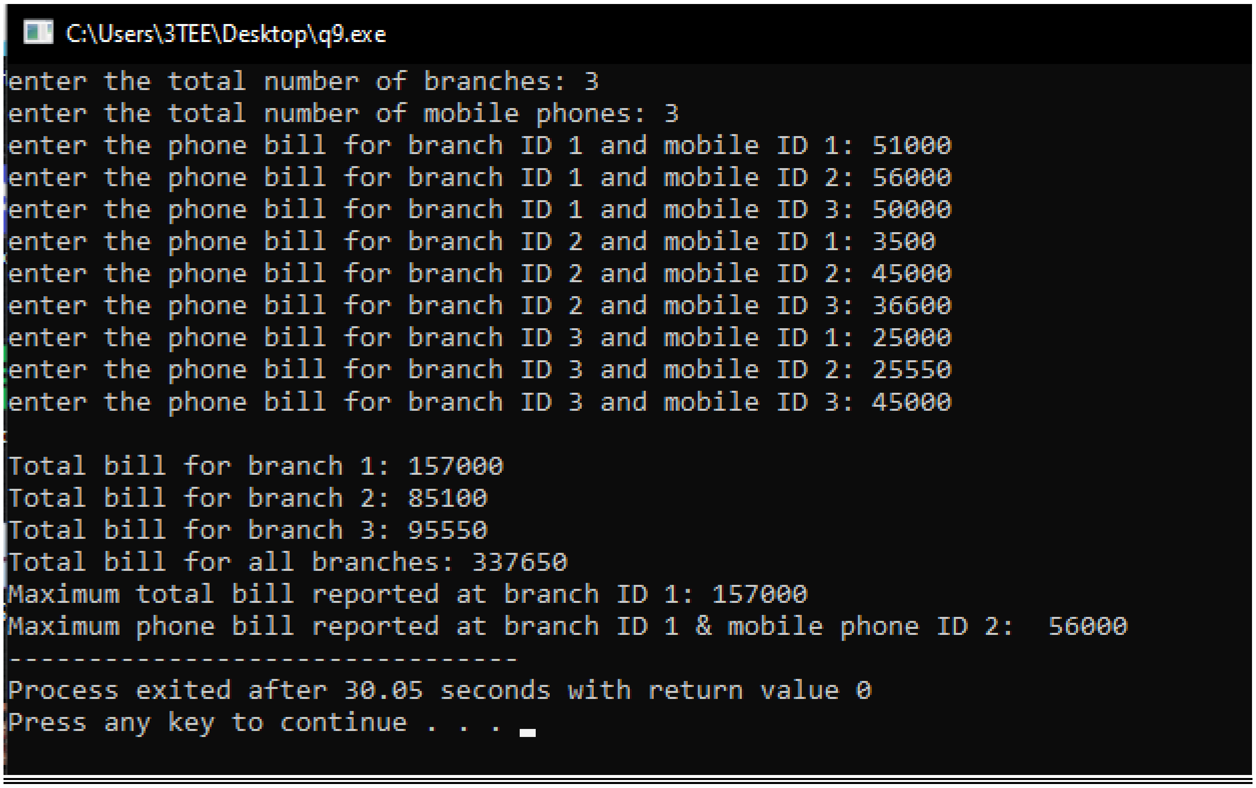
TASK 09:



```c
#include <stdio.h>

int main () {
    /*I am assuming 3 branches and 3 mobile phones, that's why [3][3]*/
    int i, j, branches, mobiles, bill[3][3] , total_bill, max_total_bill=0, max_bill_branchID, max_bill_mobileID, max_phone_bill;

    printf("enter the total number of branches: ");
    scanf("%d", &branches);
    printf("enter the total number of mobile phones: ");
    scanf("%d", &mobiles);
    for (i=0; i<branches; i++) {
        for (j=0; j<mobiles; j++) {
        printf("enter the phone bill for branch ID %d and mobile ID %d: ", i+1, j+1);
        scanf("%d", &bill[i][j]);
        }
    }

    // Total bill for each branches
    printf("\n");
    for (i=0; i<branches; i++) {
        total_bill = 0;
        for (j=0; j<mobiles; j++) {
            total_bill += bill[i][j];
        }
        printf("Total bill for branch %d: %d\n", i+1, total_bill);

    }

    // Total bill for all branches
    total_bill = 0;
    for (i=0; i<branches; i++) {
        for (j=0; j<mobiles; j++) {
            total_bill += bill[i][j];
        }
    }
        printf("Total bill for all branches: %d", total_bill);
```



```c

    // Branch ID where max bill
    printf("\n");
    for (i=0; i<branches; i++) {
        total_bill = 0;
        for (j=0; j<mobiles; j++) {
            total_bill += bill[i][j];
//          if (max_bill < total_bill) {
//          max_bill = total_bill;
//          max_bill_branchID = i + 1;
//          }
        }
            if (max_total_bill < total_bill) {
            max_total_bill = total_bill;
            max_bill_branchID = i + 1;
            }
    }
    printf("Maximum total bill reported at branch ID %d: %d\n", max_bill_branchID, max_total_bill);

    // Branch and Mobile Phone IDs where bill is highest of all mobile phones.
    max_phone_bill = 0;
    for (i=0; i<branches; i++) {
        total_bill = 0;
        for (j=0; j<mobiles; j++) {
            if (max_phone_bill < bill[i][j]) {
            max_phone_bill = bill[i][j];
            max_bill_branchID = i + 1;
            max_bill_mobileID = j + 1;
            }
        }

    }
    printf("Maximum phone bill reported at branch ID %d & mobile phone ID %d:  %d", max_bill_branchID, max_bill_mobileID, max_phone_bill);

    return 0;
}
```



```
enter the total number of branches: 3
enter the total number of mobile phones: 3
enter the phone bill for branch ID 1 and mobile ID 1: 51000
enter the phone bill for branch ID 1 and mobile ID 2: 56000
enter the phone bill for branch ID 1 and mobile ID 3: 50000
enter the phone bill for branch ID 2 and mobile ID 1: 3500
enter the phone bill for branch ID 2 and mobile ID 2: 45000
enter the phone bill for branch ID 2 and mobile ID 3: 36600
enter the phone bill for branch ID 3 and mobile ID 1: 25000
enter the phone bill for branch ID 3 and mobile ID 2: 25550
enter the phone bill for branch ID 3 and mobile ID 3: 45000

Total bill for branch 1: 157000
Total bill for branch 2: 85100
Total bill for branch 3: 95550
Total bill for all branches: 337650
Maximum total bill reported at branch ID 1: 157000
Maximum phone bill reported at branch ID 1 & mobile phone ID 2:  56000
--------------------------------
Process exited after 30.05 seconds with return value 0
Press any key to continue . . .
```