Q1)

```c
int main()
{
    int Seats[10][4], i, j, row, col, flag = 0;
    printf("Enter if seat is reserved or available (1 = reserved / 0 = available) \n");
    for (i = 0; i < 10; i++){
        for (j = 0; j < 4; j++){
            printf("Row %d Seat %d \n" , i+1, j+1);
            scanf("%d", &Seats[i][j]);
        }
    }
    printf("Please enter which seat would you like to take? \n");
    printf("\nSeating Plan:\n");
    for (i = 0; i <10; i++) {
        for (j = 0; j < 4; j++) {
            printf("%d ", Seats[i][j]);
        }
        printf("\n");
    }
    printf("Enter Row \n");
    scanf("%d", &row);
    printf("Enter Column \n");
    scanf("%d", &col);
    row--;
    col--;
    while (flag == 0){
        if (Seats[row][col] == 1){
            printf("Seat Already Reserved \n");
            printf("Please Enter Another Seat \n");
            printf("Enter Row \n");
            scanf("%d", &row);
            printf("Enter Column \n");
            scanf("%d", &col);
            row--;
            col--;

        }else{
            Seats[row][col] = 1;
            flag = 1;
            printf("Seat Reserved Sucessfully!")
        }
    }
    return 0;
}
```

```
Row 8 Seat 4
1
Row 9 Seat 1
0
Row 9 Seat 2
0
Row 9 Seat 3
1
Row 9 Seat 4
1
Row 10 Seat 1
1
Row 10 Seat 2
1
Row 10 Seat 3
1
Row 10 Seat 4
0
Please enter which seat would you like to take?

Seating Plan:
1 1 1 1
1 1 1 0
0 0 0 0
0 0 0 1
0 0 1 1
1 0 0 0
1 1 0 0
0 1 1 1
0 0 1 1
1 1 1 0
Enter Row
1
Enter Column
3
Seat Already Reserved
Please Enter Another Seat
Enter Row
2
Enter Column
4
Seat Reserved Sucessfully!
--------------------------------
Process exited after 41.9 seconds with return value 0
Press any key to continue . . .
```

Q2)

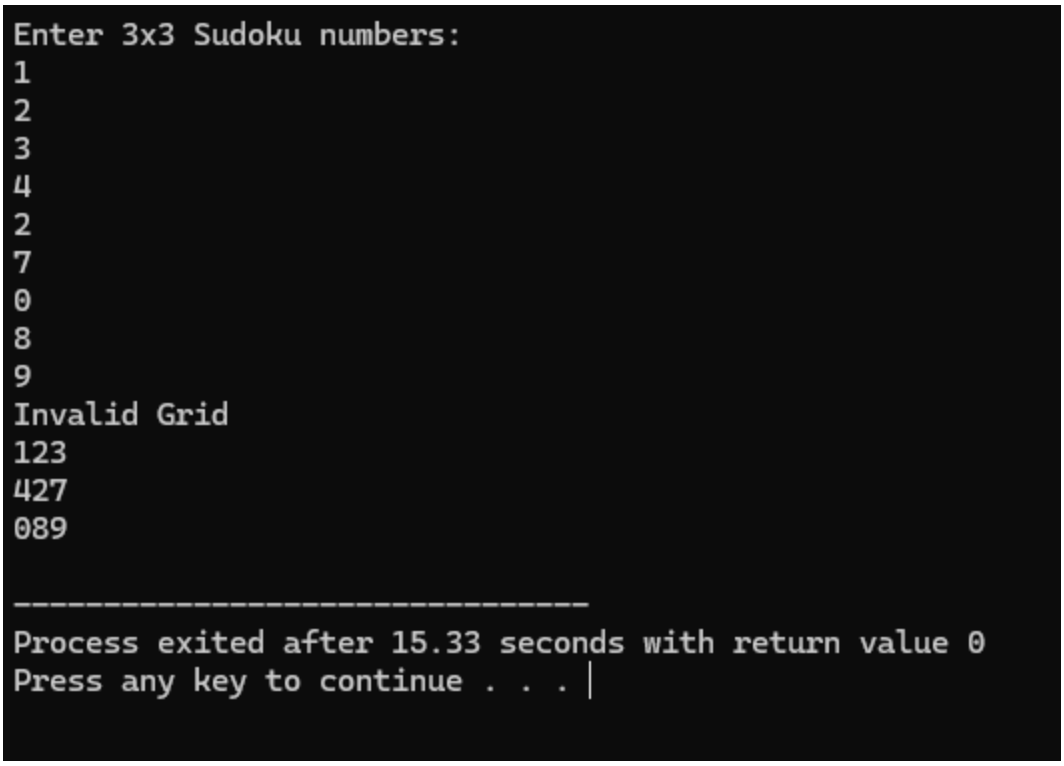
```c
#include <stdio.h>

int main() {
    int sudoku[3][3], i, j, x, y;
    int valid = 1;

    printf("Enter 3x3 Sudoku numbers:\n");
    for(i=0;i<3;i++){
        for(j=0;j<3;j++){
            scanf("%d",&sudoku[i][j]);
            if(sudoku[i][j]<1 || sudoku[i][j]>9)
                valid = 0;
        }
    }

    for(i=0;i<3;i++){
        for(j=0;j<3;j++){
            for(x=0;x<3;x++){
                for(y=0;y<3;y++){
                    if((i!=x || j!=y) && sudoku[i][j]==sudoku[x][y])
                        valid = 0;
                }
            }
        }
    }

    if(valid){
        printf("Valid Grid\n");
    }else{
        printf("Invalid Grid\n");
    }
    for (i=0;i<3;i++){
        for (j=0;j<3;j++){
            printf("%d", sudoku[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```


Enter 3x3 Sudoku numbers:
1
2
3
4
2
7
0
8
9
Invalid Grid
123
427
089
--------------------------------
Process exited after 15.33 seconds with return value 0
Press any key to continue . . .

Q3)

```c
#include <stdio.h>

int main() {
    float sales[4][5];
    int totalSales[4]={0}, maxProduct=0;
    int maxSaleMan=0, i, j;

    printf("Enter sales for 4 salesmen and 5 products:\n");
    for(i=0;i<4;i++){
        printf("Enter for salesman %d \n", i+1);
        for(j=0;j<5;j++){
            scanf("%f",&sales[i][j]);
            totalSales[i]+=sales[i][j];
            if(sales[i][j]>maxProduct){
                maxProduct=sales[i][j];
                maxSaleMan=i;
            }
        }
    }

    for(i=0;i<4;i++){
        if(totalSales[i]>10000){
            printf("Salesman %d gets a 10 Percent Bonus \n");
        }
    }

    printf("\nSalesman totals:\n");
    for(i=0;i<4;i++)
        printf("Salesman %d = %d\n",i+1,totalSales[i]);

    printf("Salesman %d sold most expensive product, Amount = %d \n",maxSaleMan+1, maxProduct);
    return 0;
}
```

```
Enter sales for 4 salesmen and 5 products:
Enter for salesman 1
1110
2200
3300
4400
10
Enter for salesman 2
20000
30000
10
10
10
Enter for salesman 3
2400
2500
2600
7000
15
Enter for salesman 4
300
400
500
600
700
Salesman 3 gets a 10 Percent Bonus
Salesman 65536 gets a 10 Percent Bonus
Salesman 65536 gets a 10 Percent Bonus

Salesman totals:
Salesman 1 = 11020
Salesman 2 = 50030
Salesman 3 = 14515
Salesman 4 = 2500
Salesman 2 sold most expensive product, Amount = 30000

--------------------------------
Process exited after 46.78 seconds with return value 0
Press any key to continue . . .
```

Q4)

```c
#include <stdio.h>

int main() {
    int temp[3][3][3];
    float layerAvg[3];
    int i,j,k, sum;
    int max=temp[0][0][0];
    int x=0,y=0,z=0;

    printf("Enter temperatures for 3x3x3 cube:\n");
    for(i=0;i<3;i++){
        sum=0;
        for(j=0;j<3;j++){
            for(k=0;k<3;k++){
                scanf("%d",&temp[i][j][k]);
                sum+=temp[i][j][k];
                if(temp[i][j][k]>max){
                    max=temp[i][j][k];
                    x=i; y=j; z=k;
                }
            }
        }
        layerAvg[i]=sum/9;
    }

    for(i=0;i<3;i++)
        printf("Average temp of layer %d = %.2f\n",i+1,layerAvg[i]);

    printf("Hottest point = %d at (%d,%d,%d)\n",max,x,y,z);
    return 0;
}
```

```
Enter temperatures for 3x3x3 cube:
12
14
16
18
10
22
35
46
20
37
37
37
23
24
25
12
11
10
22
13
32
29
20
11
10
38
34
Average temp of layer 1 = 21.44
Average temp of layer 2 = 24.00
Average temp of layer 3 = 23.22
Hottest point = 46 at (0,2,1)

--------------------------------
Process exited after 64.33 seconds with return value 0
Press any key to continue . . .
```

Q5)

```c
#include <stdio.h>

int main() {
    int marks[2][3][3];
    int total[2][3]={0};
    int classTop[2]={0}, classTopMarks[2]={0};
    int overallTop=0, overallMarks=0, i, j, k;

    printf("Enter marks: \n");
    for(i=0;i<2;i++){
        printf("Class %d \n", i+1);
        for(j=0;j<3;j++){
            printf("Student %d \n", j+1);
            for(k=0;k<3;k++){
                printf("Subject %d \n", k+1);
                scanf("%d",&marks[i][j][k]);
                total[i][j]+=marks[i][j][k];
            }
            if(total[i][j]>classTopMarks[i]){
                classTopMarks[i]=total[i][j];
                classTop[i]=j;
            }
            if(total[i][j]>overallMarks){
                overallMarks=total[i][j];
                overallTop=j;
            }
        }
    }

    for(i=0;i<2;i++)
        printf("Top Performer of Class %d: Student %d with %d marks\n",i+1,classTop[i]+1,classTopMarks[i]);
    printf("Overall Topper: Student %d with %d marks\n",overallTop+1,overallMarks);

    return 0;
}
```

```
Enter marks:
Class 1
Student 1
Subject 1
80
Subject 2
90
Subject 3
78
Student 2
Subject 1
88
Subject 2
99
Subject 3
76
Student 3
Subject 1
58
Subject 2
69
Subject 3
70
Class 2
Student 1
Subject 1
98
Subject 2
80
Subject 3
90
Student 2
Subject 1
70
Subject 2
77
Subject 3
80
Student 3
Subject 1
65
Subject 2
40
Subject 3
77
Top Performer of Class 1: Student 2 with 263 marks
Top Performer of Class 2: Student 1 with 268 marks
Overall Topper: Student 1 with 268 marks

--------------------------------
Process exited after 46.91 seconds with return value 0
```

Q6)

```c
#include <stdio.h>

int main() {
    int i,j;

    for(i=1;i<=5;i++){
        for(j=1;j<=i;j++)
            printf("%d", j*i);
        printf("\n");
    }
    for(i=5;i>=1;i--){
        for(j=1;j<=i;j++)
            printf("%d", j*i);
        printf("\n");
    }

    return 0;
}
```

```
1
24
369
481216
510152025
510152025
481216
369
24
1

--------------------------------
Process exited after 0.2779 seconds with return value 0
Press any key to continue . . .
```

Q7)

```c
#include <stdio.h>

int main() {
    int data[3][3], key[3][3], enc[3][3];
    int max=0, x=0, y=0, i, j;

    printf("Enter Data matrix:\n");
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            scanf("%d",&data[i][j]);

    printf("Enter Key matrix:\n");
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            scanf("%d",&key[i][j]);

    printf("Encrypted Matrix:\n");
    for(i=0;i<3;i++){
        for(j=0;j<3;j++){
            enc[i][j] = (data[i][j]*key[i][j]) + (i+j);
            printf("%d ",enc[i][j]);
            if(enc[i][j]>max){
                max=enc[i][j];
                x=i; y=j;
            }
        }
        printf("\n");
    }

    printf("Max encrypted value = %d at cell (%d,%d)\n",max,x,y);
    return 0;
}
```

```
Enter Data matrix:
1
2
3
4
5
6
7
8
9
Enter Key matrix:
5
3
2
1
5
6
7
8
9
Encrypted Matrix:
5 7 8
5 27 39
51 67 85
Max encrypted value = 85 at cell (2,2)

--------------------------------
Process exited after 20.5 seconds with return value 0
Press any key to continue . . .
```