

Practical-1

Aim: A Train going to Busan have 2 containers which contains Zombies. Container A has 6 zombies, and Container B has 4 zombies. Passengers have to reach in engine container by passing through them. Help them by transferring zombies from both that containers to a new Container C. Build a C++ program for it.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int i,j,n1,n2,n3,k;

cout << "~ Enter Size for First Array : ";
cin >> n1;
cout << endl << "~ Elements for First Array: " << endl;

int a[n1],b[n2],c[n3];

for(i=0;i<n1;i++)
{
cout << "- a[" << i << "] : ";
cin >> a[i];
c[i]=a[i];
}

k=i;
cout << endl << "~ Enter Size for Second Array : ";
cin >> n2;
cout << endl << "~ Elements for Second Array : " << endl;
```

```

for(i=0;i<n2;i++)
{
cout << "- b["<<i<<"] : ";
cin >> b[i];
c[k]=b[i];
k++;
}

cout <<endl <<"~ The New Array (Merged Array) : ";
for(i=0;i<k;i++)
{
cout << c[i] << " ";
}
cout <<endl;

return 0;

}

```

Output:

```

M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\1.exe
~ Enter Size for First Array : 5
~ Elements for First Array:
- a[0] : 1
- a[1] : 0
- a[2] : 1
- a[3] : 7
- a[4] : 2
~ Enter Size for Second Array : 5
~ Elements for Second Array :
- b[0] : 1
- b[1] : 9
- b[2] : 2
- b[3] : 7
- b[4] : 7
~ The New Array (Merged Array) : 1 0 1 7 1 9 2 7 7 7
-----
Process exited after 13.38 seconds with return value 0
Press any key to continue . . .

```

Practical-2

Aim: Hitler ordered a 10 soldiers paired to align in a row. He wants to know that which soldier have the highest killing score. Help him by design a C++ Program.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int i,larg;

cout << "~ Size of soliders paired : 10 "<<endl;
cout <<endl<< "~ Elements of array : "<<endl<<endl;

int a[10];

for(i=0;i<10;i++)
{
cout << "a["<<i<<"]": ";
cin >> a[i];
}

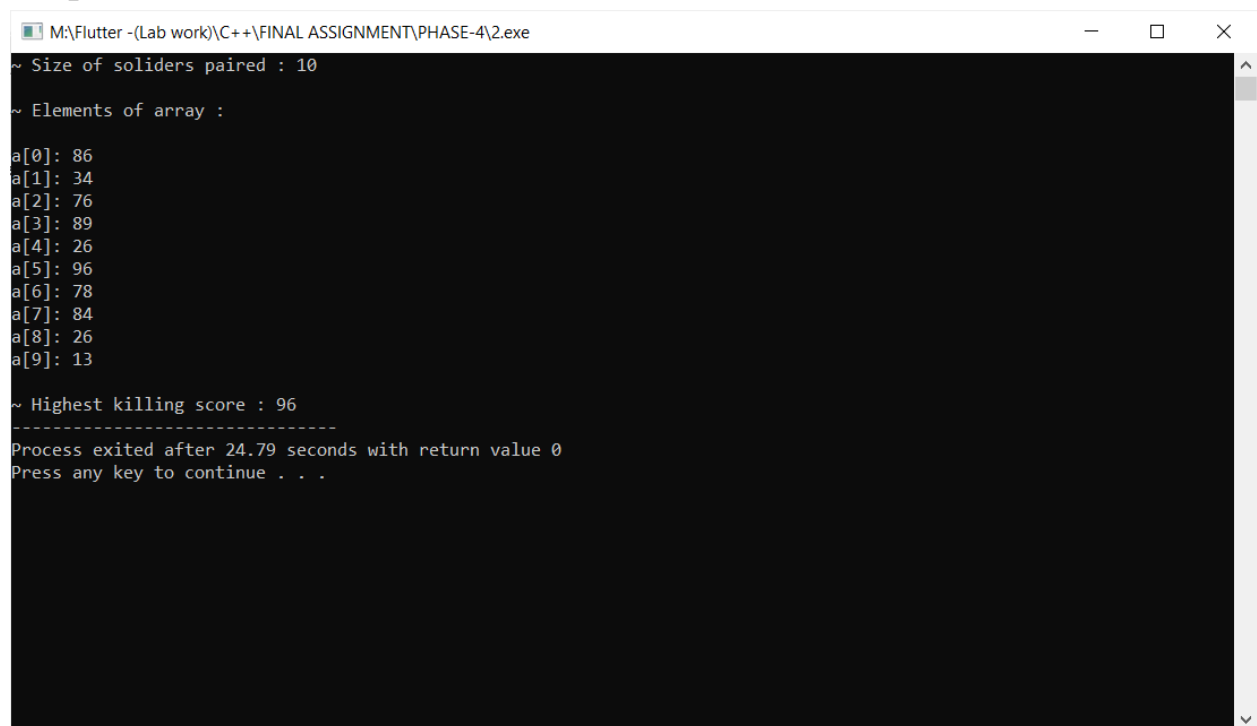
larg = a[0];

cout <<endl<< "~ Highest killing score : ";
for(i=1;i<10;i++)
{
if(larg<a[i])
larg = a[i];
}
cout <<larg;
```

```
return 0;
```

```
}
```

Output:



```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\2.exe
~ Size of soliders paired : 10
~ Elements of array :
a[0]: 86
a[1]: 34
a[2]: 76
a[3]: 89
a[4]: 26
a[5]: 96
a[6]: 78
a[7]: 84
a[8]: 26
a[9]: 13
~ Highest killing score : 96
-----
Process exited after 24.79 seconds with return value 0
Press any key to continue . . .
```

Practical-3

Aim: Design a C++ system which automatically identifies if a given word contains any letter or symbol between both SHIFT keys from our regular PC Keyboard.

Program:

```
#include<iostream>
#include<string.h>
using namespace std;

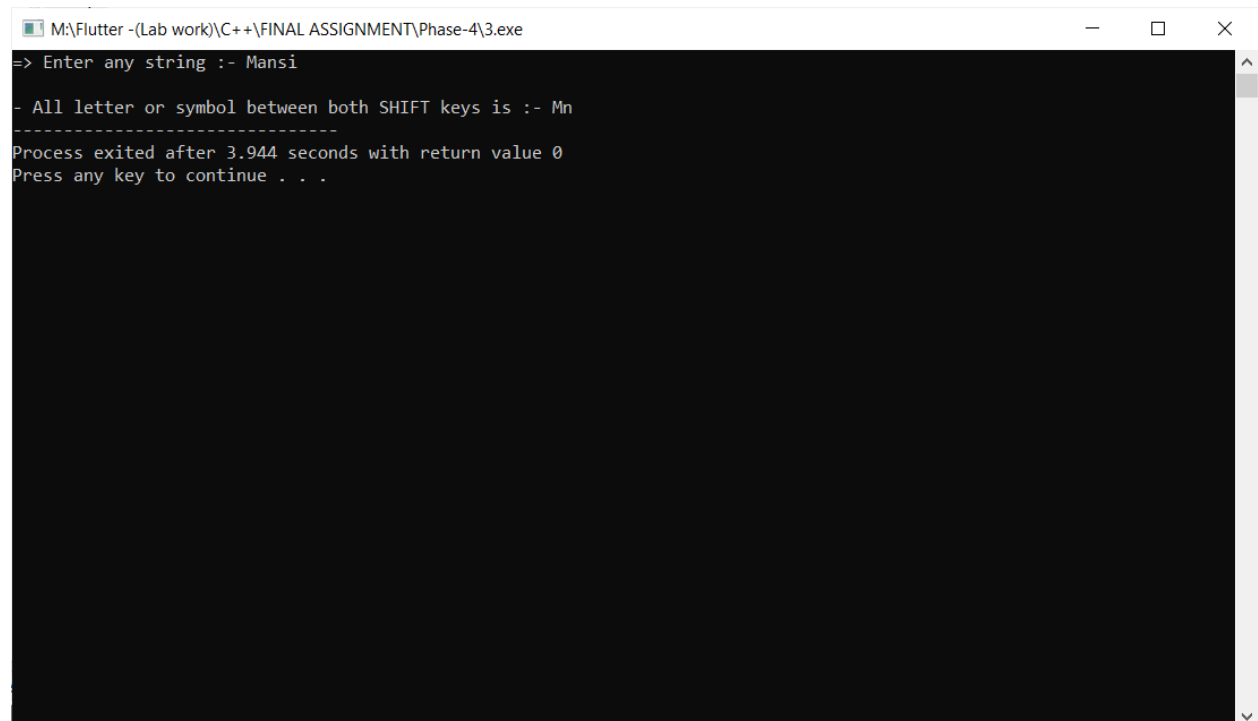
class Keyboard
{
    private:
        char a[100];
        int i;

    public:
        Keyboard()
        {
            cout<<"=> Enter any string :- "; cin>>this->a;
        }

        void Shift()
        {
            cout<<endl<<"- All letter or symbol between both SHIFT keys is :- ";
            for(i=0;a[i]!='\0';i++)
            {
                if(a[i]=='z'||a[i]=='x'||a[i]=='c'||a[i]=='v'||a[i]=='b'||a[i]=='n'||a[i]=='m'||
                a[i]=='Z'||a[i]=='X'||a[i]=='C'||a[i]=='V'||a[i]=='B'||a[i]=='N'||a[i]=='M'||
                a[i]==' ','||a[i]=='.'||a[i]=='/'||a[i]=='<'||a[i]=='>'||a[i]=='?')
                {
                    cout<<a[i];
                }
            }
        }
    }
```

```
        }  
    }  
};  
int main()  
{  
    Keyboard k1;  
  
    k1.Shift();  
  
    return 0;  
}
```

Output:



```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\3.exe  
=> Enter any string :- Mansi  
  
- All letter or symbol between both SHIFT keys is :- Mn  
-----  
Process exited after 3.944 seconds with return value 0  
Press any key to continue . . .
```

Practical-4

Aim: Ronak gives an examination in which he gains internal viva marks in all subjects like this: 23, 19, 22, 28 & 23(all marks are out of 30). And gains final written examination marks like this: 65, 58, 49, 52 & 64 (all marks are out of 70). Evaluate final total examination marks by adding both exam marks and reveal marks out of 100 in each subject by using C++. Also, final total average value for that.

Program:

```
#include<iostream>
#include<string.h>
using namespace std;

class Ronak
{
    private:
        int Ronak1[5] = {23, 19, 22, 28, 23};
        int Ronak2[5] = {65, 58, 49, 52, 64};
        int sum=0, i;

    public:

        void Mark()
        {
            cout<<endl<<"=> Ronak viva marks :- 23, 19, 22, 28, 23"<<endl;
            cout<<"=> Ronak examination marks :- 65, 58, 49, 52, 64 "<<endl;
            cout<<endl<<"=> Final Examination Mark of :- ";
            for(i=0;i<5;i++)
            {
                cout<<" , "<<Ronak1[i]+Ronak2[i];

                sum = sum +(Ronak1[i]+Ronak2[i]);
            }
        }
    }
```

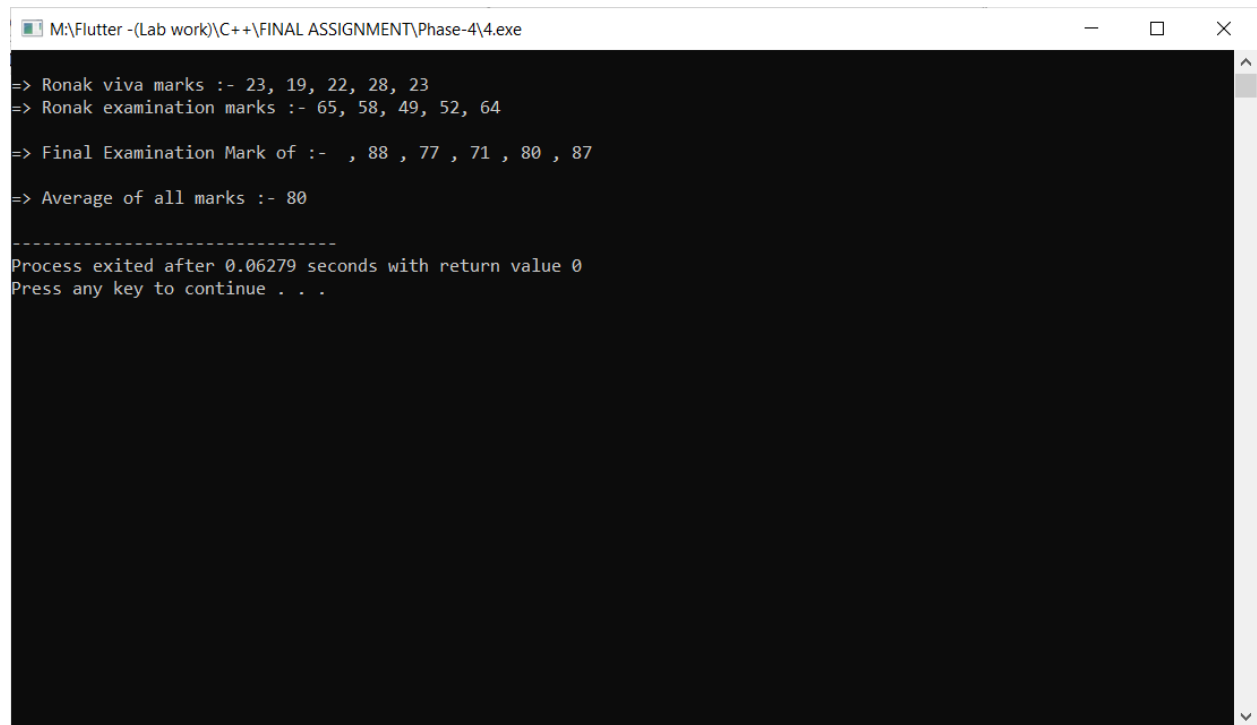
```
        cout<<endl;
        cout<<endl<<"=> Average of all marks :- "<<sum/5<<endl;
    }
};

int main()
{
    Ronak r1;

    r1.Mark();

    return 0;
}
```


Output:



```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\4.exe

=> Ronak viva marks :- 23, 19, 22, 28, 23
=> Ronak examination marks :- 65, 58, 49, 52, 64

=> Final Examination Mark of :-  , 88 , 77 , 71 , 80 , 87

=> Average of all marks :- 80

-----
Process exited after 0.06279 seconds with return value 0
Press any key to continue . . .
```

Practical-5

Aim: Devansh gives a list of random numbers to his colleague Rohan to distinguish all odd and even numbers, and store them in different lists. Help Rohan by building such a solution with help of C++.

Program:

```
#include<iostream>
using namespace std;

int main()
{
    int n,i,j;

    cout << "-> Enter Size of Array : ";
    cin >> n;
    cout << endl << "-> Elements of Array : " << endl;

    int a[n];
    for(i=0 ;i<n ;i++)
    {
        cout << "- a[" << i << "]" : ";
        cin >> a[i];
    }

    cout << endl << "-> Even number : " << endl;

    for(i=0;i<n;i++)
    {
        if(a[i]%2==0)
        {
            cout << " " << a[i] ;
        }
    }
```

```
}  
cout <<endl;  
  
cout <<endl<< "-> Odd number : "<<endl;  
  
for(i=0;i<n;i++)  
{  
if(a[i]%2!=0)  
{  
cout << " "<< a[i] ;  
}  
}  
  
return 0;  
  
}
```

Output:

```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\5.exe
-> Enter Size of Array : 10

-> Elements of Array :
- a[0] : 54
- a[1] : 21
- a[2] : 34
- a[3] : 65
- a[4] : 43
- a[5] : 87
- a[6] : 56
- a[7] : 11
- a[8] : 39
- a[9] : 98

-> Even number :
54 34 56 98

-> Odd number :
21 65 43 87 11 39
-----
Process exited after 25.56 seconds with return value 0
Press any key to continue . . .
```

Practical-6

Aim: A Teacher gives a list to students in which a list contains many years in numeric format i.e 1994, 1947, 2002, 1996, etc. All students supposed to find all duplicate years which occurs more than once and store them into another list. Design such type of system with help of C++.

Program:

```
#include<iostream>
using namespace std;

int main()
{
    int i,j,n;

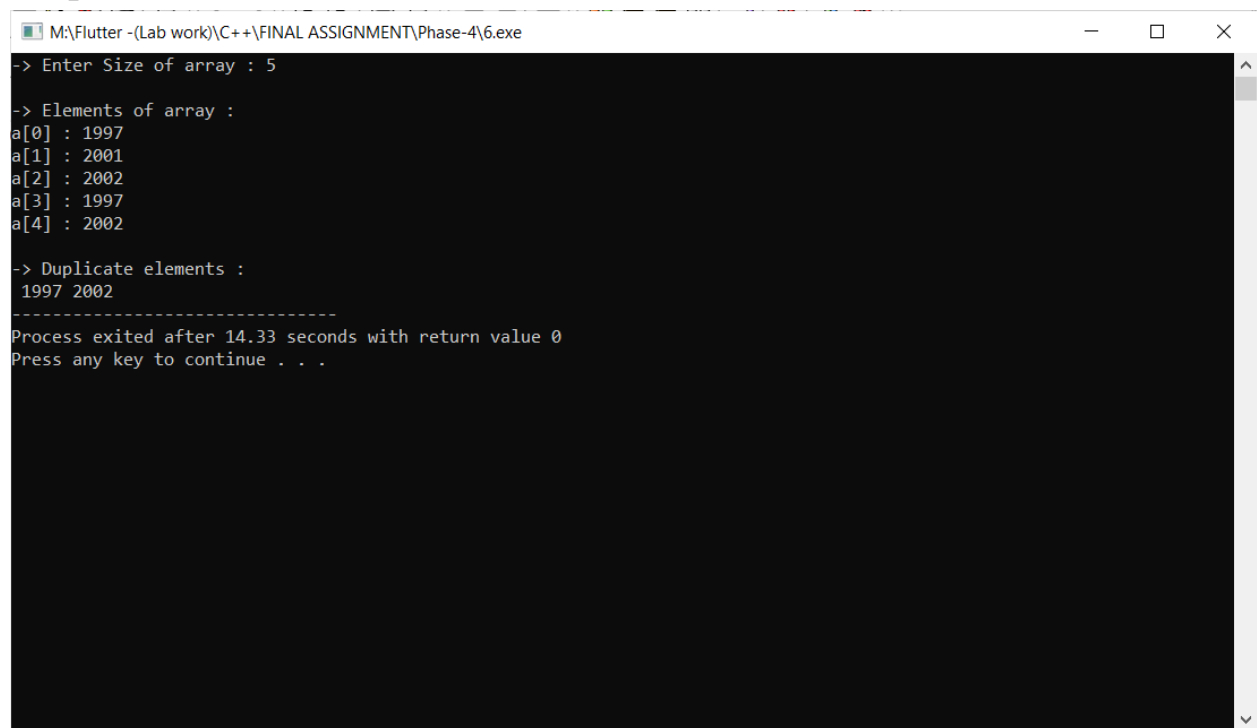
    cout << "-> Enter Size of array : ";
    cin >> n;
    cout << endl << "-> Elements of array : " << endl;

    int a[n];
    for(i=0;i<n;i++)
    {
        cout << "a[" << i << "] : ";
        cin >> a[i];
    }

    cout << endl << "-> Duplicate elements : " << endl;
    for(i=0;i<n;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(a[i]==a[j])
            {
                cout << " " << a[i];
            }
        }
    }
}
```

```
        }  
    }  
    return 0;  
}
```

Output:



```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\6.exe  
-> Enter Size of array : 5  
-> Elements of array :  
a[0] : 1997  
a[1] : 2001  
a[2] : 2002  
a[3] : 1997  
a[4] : 2002  
-> Duplicate elements :  
1997 2002  
-----  
Process exited after 14.33 seconds with return value 0  
Press any key to continue . . .
```

Practical-7

Aim: A Frontman can randomly assign two 3x3 matrices to all participants in Squid Games. All participants have to add that matrices and store final answer as a separate matrix to win this type of round in the game. Build a C++ system to help them all.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int i,j;

cout <<"-> Enter 3*3 Matrix : "<<endl;
cout <<endl<<"-> Enter Size of Array 1 : "<<endl<<endl;

int a[3][3];
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
cout << "a["<<i<<"]["<<j<<"]: ";
cin >> a[i][j];
}
}

cout <<endl<<"-> Elements of array 1 : "<<endl<<endl;

for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
```

```
cout << " " << a[i][j];
}
cout << endl;
}
cout << endl << "-----" << endl << endl;
```

```
cout << "-> Enter Size of Array 2 : " << endl << endl;
```

```
int b[3][3];
for(i=0;i<3;i++)
{
    for(j=0;j<3;j++)
    {
        cout << "b[" << i << "][" << j << "]: ";
        cin >> b[i][j];
    }
}
cout << endl << "-> Elements of array 2 : " << endl << endl;
for(i=0;i<3;i++)
{
    for(j=0;j<3;j++)
    {
        cout << " " << b[i][j];
    }
    cout << endl;
}
```

```
cout << endl << "-----" << endl << endl;
```

```
int c[3][3];
for(i=0;i<3;i++)
{
    for(j=0;j<3;j++)
    {
        c[i][j]=a[i][j]+b[i][j];
    }
}
```

```
cout << "-> Sum of two matrix : " << endl << endl;
for(i=0;i<3;i++)
```



```

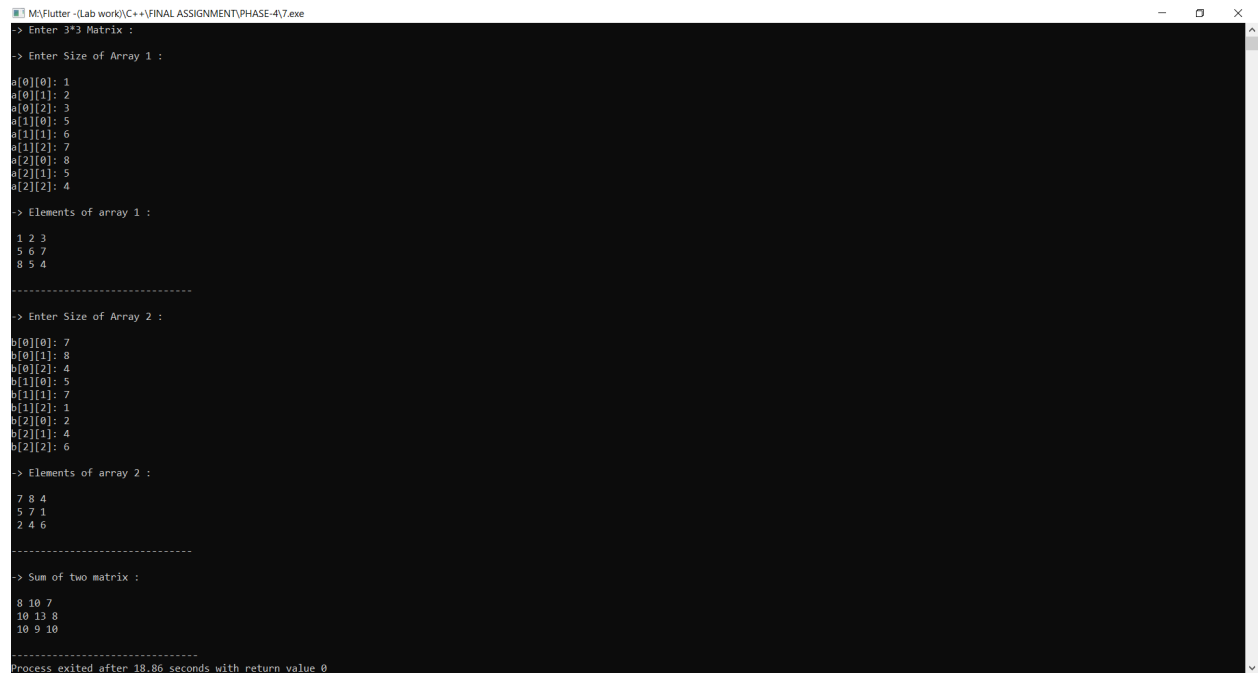
{
for(j=0;j<3;j++)
{
cout << " " <<c[i][j];
}
cout << endl;
}

return 0;

}

```

Output:



```

M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\7.exe
-> Enter 3*3 Matrix :

-> Enter Size of Array 1 :

a[0][0]: 1
a[0][1]: 2
a[0][2]: 3
a[1][0]: 5
a[1][1]: 6
a[1][2]: 7
a[2][0]: 8
a[2][1]: 5
a[2][2]: 4

-> Elements of array 1 :

1 2 3
5 6 7
8 5 4

-----
-> Enter Size of Array 2 :

b[0][0]: 7
b[0][1]: 8
b[0][2]: 4
b[1][0]: 5
b[1][1]: 7
b[1][2]: 1
b[2][0]: 2
b[2][1]: 4
b[2][2]: 6

-> Elements of array 2 :

7 8 4
5 7 1
2 4 6

-----
-> Sum of two matrix :

8 10 7
10 13 8
10 9 10

-----
Process exited after 18.86 seconds with return value 0

```

Practical-8

Aim: Design a system in C++ which automatically transpose any given Matrix of RxC dimension. Where R is number of Rows and C is number of Columns. Help three musketeers for passing an interview round by solving this last question.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int r,c,i,j;

cout << "-> How many rows : ";
cin >> r;
cout << "-> How many cols : ";
cin >> c;

cout << endl << "-> Enter matrix elements : " << endl << endl;

int a[r][c];
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout << "- a[" << i << "][" << j << "]: ";
cin >> a[i][j];
}
}
cout << endl << "-> Entered Matrix : " << endl << endl;
for(i=0;i<r;i++)
{
```

```
for(j=0;j<c;j++)
{
cout << " "<< a[i][j];
}
cout << endl;
}
```

```
int tra[r][c];
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
tra[i][j] = a[i][j];
}
}
```

```
cout <<endl<< "-> Transpose Matrix : "<<endl<<endl;
for(i=0;i<c;i++)
{
for(j=0;j<r;j++)
{
cout <<" " << tra[j][i];
}
cout << endl;
}
```

```
return 0;
```

```
}
```

Output:

```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\8.exe
-> How many rows : 3
-> How many cols : 3

-> Enter matrix elements :

- a[0][0] : 1
- a[0][1] : 5
- a[0][2] : 6
- a[1][0] : 8
- a[1][1] : 3
- a[1][2] : 6
- a[2][0] : 1
- a[2][1] : 9
- a[2][2] : 5

-> Entered Matrix :

1 5 6
8 3 6
1 9 5

-> Transpose Matrix :

1 8 1
5 3 9
6 6 5

-----
Process exited after 12.37 seconds with return value 0
Press any key to continue . . .
```

Practical-9

Aim: Colombian army arranged all 9 forbidden refugees in a 3x3 matrix formation. An army cadets have to find that which one of the refugees stats the highest weight and which one weighs the lowest weight. Help army cadets by preparing C++ solution for their undercover mission.

Program:

```
#include<iostream>
using namespace std;

class Colombian_army
{
    private:
        int a[3][3];
        int i,j;

    public:

        void setdata()
        {
            cout<<endl<<"=> Enter Refugees stats weight : "<<endl;

            for(i=0;i<3;i++)
            {
                for(j=0;j<3;j++)
                {
                    cout<<" a["<<i<<"]["<<j<<"]: ";
                    cin>>a[i][j];
                }
            }
        }
}
```

```

void solution()
{
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            if(a[i][j]>a[0][0])
            {
                a[0][0]=a[i][j];
            }
        }
    }
    cout<<endl<<"=> One of the refugees stats the highest weight is:
"<<a[0][0]<<endl;

    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            if(a[i][j]<a[0][0])
            {
                a[0][0]=a[i][j];
            }
        }
    }
    cout<<endl<<"=> One of the refugees stats the Lowest weight is:
"<<a[0][0]<<endl;
}

};

int main()
{
    Columbian_army c1;

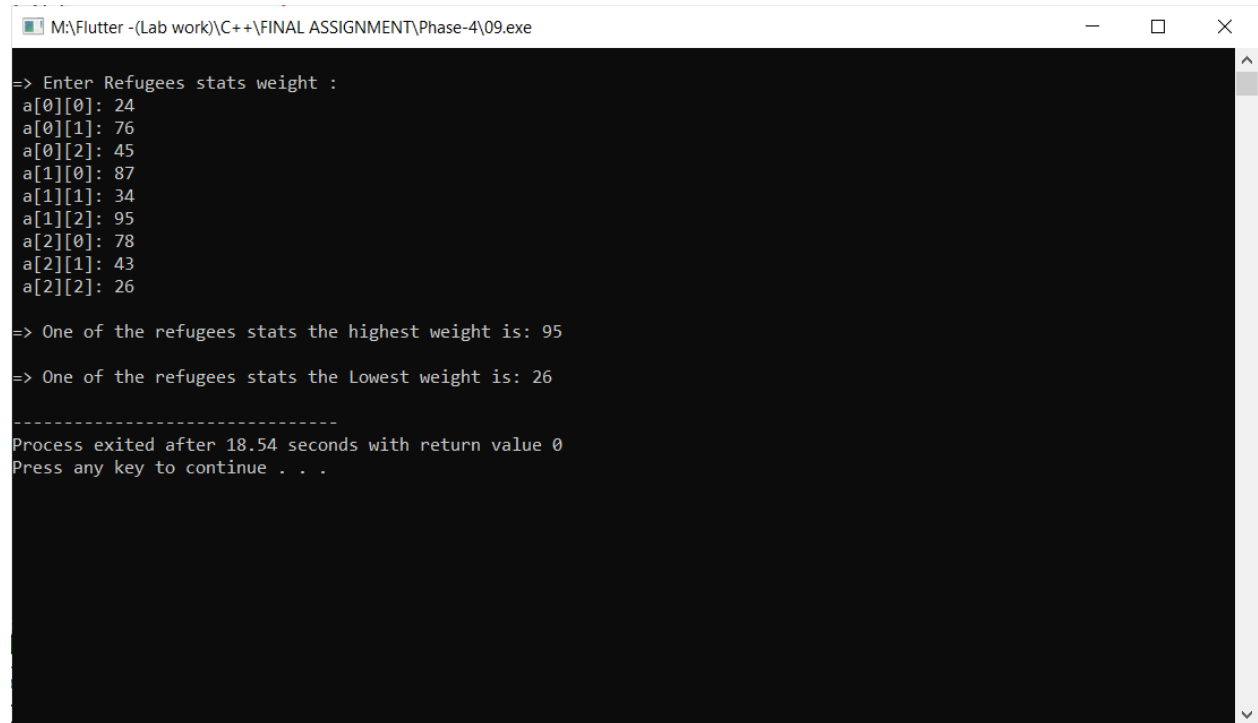
    c1.setdata();

    c1.solution();

    return 0;
}

```

Output:



```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\09.exe

=> Enter Refugees stats weight :
a[0][0]: 24
a[0][1]: 76
a[0][2]: 45
a[1][0]: 87
a[1][1]: 34
a[1][2]: 95
a[2][0]: 78
a[2][1]: 43
a[2][2]: 26

=> One of the refugees stats the highest weight is: 95
=> One of the refugees stats the Lowest weight is: 26

-----
Process exited after 18.54 seconds with return value 0
Press any key to continue . . .
```

Practical-10

Aim: Help Martin to solve a special kind of puzzle by designing a C++ system. Total 25 random numbers arranged in a form of Square Matrix. To solve that puzzle, he has to find addition of all diagonally aligned numbers on puzzle cardboard.

Program:

```
#include<iostream>
using namespace std;

class Martin
{
    private:
        int a[5][5];
        int i,j,sum=0;

    public:

        Martin()
        {
            cout<<endl<<"=> Enter 25 Random Number:-"<<endl;

            for(i=0;i<5;i++)
            {
                for(j=0;j<5;j++)
                {
                    cout<<"- a["<<i<<"]["<<j<<"]: ";
                    cin>>a[i][j];
                }
            }
        }

        void solution()
        {
```



```

        cout<<endl<<"=> Matrix is:-"<<endl;

        for(i=0;i<5;i++)
        {
            for(j=0;j<5;j++)
            {
                cout<<a[i][j]<<" ";
            }
            cout<<endl;
        }
        for(i=0;i<5;i++)
        {
            for(j=0;j<5;j++)
            {
                if(i==j)
                {
                    sum+=a[i][j];
                }
            }
        }
        cout<<endl<<"=> Addition of Digonal Elements is: "<<sum<<endl;
    }
};

int main()
{
    Martin m1;

    m1.solution();

    return 0;
}

```

Output:

M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\10.exe

```
=> Enter 25 Random Number:-
- a[0][0]: 1
- a[0][1]: 7
- a[0][2]: 9
- a[0][3]: 2
- a[0][4]: 3
- a[1][0]: 5
- a[1][1]: 6
- a[1][2]: 3
- a[1][3]: 8
- a[1][4]: 9
- a[2][0]: 5
- a[2][1]: 4
- a[2][2]: 8
- a[2][3]: 2
- a[2][4]: 1
- a[3][0]: 7
- a[3][1]: 4
- a[3][2]: 5
- a[3][3]: 9
- a[3][4]: 6
- a[4][0]: 4
- a[4][1]: 1
- a[4][2]: 2
- a[4][3]: 5
- a[4][4]: 4

=> Matrix is:-
1 7 9 2 3
5 6 3 8 9
9 4 8 2 1
7 4 5 9 6
4 1 2 5 4

=> Addition of Digonal Elements is: 28

-----
Process exited after 20.05 seconds with return value 0
Press any key to continue . . .
```

Practical-11

Aim: A one-sided open Tennis ball jar has capacity of storing total 5 different balls. Each ball has unique number attached as a label itself. Arrange all that balls in a jar in such a way that their order is stats as a reverse by referring attached numbers as a label. Create a C++ system for doing such type of task.

Program:

```
#include<iostream>
using namespace std;

class Jar
{
    private:
        int j[5]={110, 220, 330, 440, 550};
        int i;

    public:

        void solution()
        {
            cout<<endl<<"=> Every Ball with unique Number: "<<endl<<endl;

            for(i=0;i<5;i++)
            {
                cout<<"=> Ball Number is: "<<i <<" "<<" -> Ball is:
" <<j[i]<<endl;;

            }

            cout<<endl<<"=> Ball order is stats as a reverse: "<<endl<<endl;
```

```

        for(i=4;i>=0;i--)
        {
            cout<<"=> Ball Number is: "<<i<<" "<<" -> Ball is:
" <<j[i]<<endl;;

        }

    };

int main()
{
    Jar j1;

    j1.solution();

    return 0;
}

```

Output:

```
M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\Phase-4\11.exe

=> Every Ball with unique Number:

=> Ball Number is: 0 -> Ball is: 110
=> Ball Number is: 1 -> Ball is: 220
=> Ball Number is: 2 -> Ball is: 330
=> Ball Number is: 3 -> Ball is: 440
=> Ball Number is: 4 -> Ball is: 550

=> Ball order is stats as a reverse:

=> Ball Number is: 4 -> Ball is: 550
=> Ball Number is: 3 -> Ball is: 440
=> Ball Number is: 2 -> Ball is: 330
=> Ball Number is: 1 -> Ball is: 220
=> Ball Number is: 0 -> Ball is: 110

-----
Process exited after 0.06076 seconds with return value 0
Press any key to continue . . .
```

Practical-12

Aim: A College wants to celebrate all degree holder students to throwing their hats in a predefined way: First all 25 students have to arranged in a Square Matrix. First, an upper half of triangle matrix willthrowing hats and then a lower half of triangle matrix will. Help them to achieve this unique idea by using C++.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int r,c,i,j;

cout << "-> How many rows : ";
cin >> r;
cout << "-> How many cols : ";
cin >> c;
cout << "-> Enter Size of array : "<<endl<<endl;

int a[r][c];
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout << "a["<<i<<"]["<<j<<"] : ";
cin >> a[i][j];
}
}
cout <<endl<< "-> Enter Matrix : "<<endl<<endl;
for(i=0;i<r;i++)
```

```

{
for(j=0;j<c;j++)
{
cout << " " <<a[i][j];
}
cout << endl;
}
cout <<endl<<"-> Lower triangular matrix : "<<endl<<endl;
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
if(i<j)
{
cout << " 0";
}
else
{
cout <<" "<<a[i][j];
}
}
cout << endl;
}
cout <<endl<<"-> Upper triangular matrix : "<<endl<<endl;
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
if(i>j)
{
cout << " 0";
}
else
{
cout <<" "<<a[i][j];
}
}
cout << endl;
}
return 0;

```

}

Output:

```

M:\Flutter -(lab work)\C++\FINAL ASSIGNMENT\PHASE-4\12.exe
-> How many rows : 3
-> How many cols : 3
-> Enter Size of array :

a[0][0] : 1
a[0][1] : 2
a[0][2] : 3
a[1][0] : 5
a[1][1] : 6
a[1][2] : 7
a[2][0] : 5
a[2][1] : 6
a[2][2] : 8

-> Enter Matrix :

1 2 3
5 6 7
5 6 8

-> Lower triangular matrix :

1 0 0
5 6 0
5 0 8

-> Upper triangular matrix :

1 2 3
0 6 7
0 0 8

-----
Process exited after 8.33 seconds with return value 0
Press any key to continue . . .
```


Practical-13

Aim: A Math teacher wants to teach how to perform a dot product of two matrices. Design a better approach in C++ to help this math teacher.

Program:

```
#include<iostream>
using namespace std;

int main()
{

int r,c,i,j;

cout << "-> How many rows : ";
cin >>r ;
cout << "-> How many cols : ";
cin >>c ;
cout <<endl<<"-> Enter Matrix elements : "<<endl <<endl;

int a[r][c];

for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout << "a["<<i<<"]["<<j<<"] : ";
cin >> a[i][j];
}
}

cout <<endl <<"-> Entered Matrix : "<<endl;

for(i=0;i<r;i++)
```

```

{
for(j=0;j<c;j++)
{
cout << " " <<a[i][j];
}
cout << endl;
}

```

```

cout <<endl<<"-> Enter Matrix elements : "<<endl <<endl;

```

```

int b[r][c];

```

```

for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout << "b["<<i<<"]["<<j<<"] : ";
cin >> b[i][j];
}
}

```

```

cout <<endl <<"-> Entered Matrix : "<<endl;
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout << " " <<b[i][j];
}
cout << endl;
}

```

```

//Displaying the multiplication of two matrix.

```

```

cout <<endl<< "-> Product of two matrices : "<<endl<<endl;
int d[r][c];
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
d[i][j] = a[i][j] * b[i][j];
}
}

```

```

}
}

for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
cout <<" "<< d[i][j];
}
cout <<endl;
}

return 0;
}

```

Output:

```

M:\Flutter -(Lab work)\C++\FINAL ASSIGNMENT\PHASE-4\13.exe
-> How many rows : 3
-> How many cols : 3

-> Enter Matrix elements :
a[0][0] : 2
a[0][1] : 3
a[0][2] : 5
a[1][0] : 6
a[1][1] : 8
a[1][2] : 7
a[2][0] : 1
a[2][1] : 2
a[2][2] : 5

-> Entered Matrix :
2 3 5
6 8 7
1 2 5

-> Enter Matrix elements :
b[0][0] : 8
b[0][1] : 3
b[0][2] : 4
b[1][0] : 6
b[1][1] : 2
b[1][2] : 7
b[2][0] : 1
b[2][1] : 5
b[2][2] : 7

-> Entered Matrix :
8 3 4
6 2 7
1 5 7

-> Product of two matrices :
16 9 20
36 16 49
1 10 35

-----
Process exited after 18.96 seconds with return value 0
Press any key to continue . . .

```