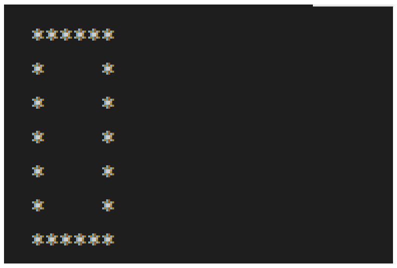
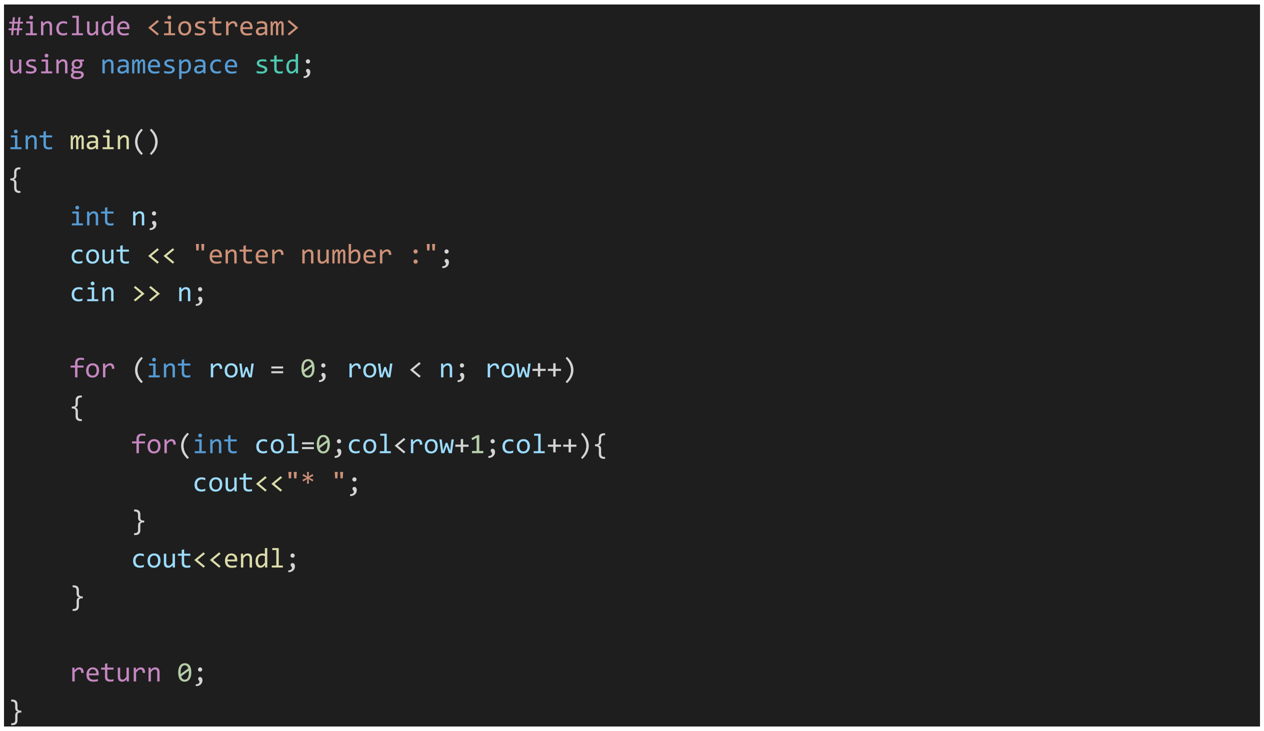
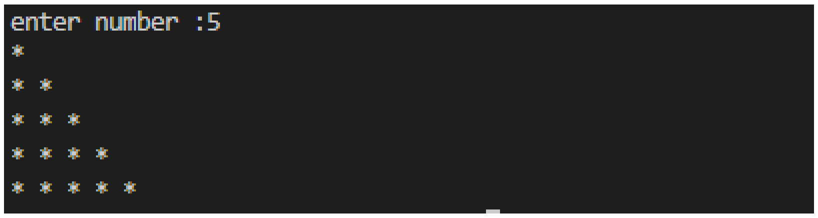
```cpp
#include <iostream>
using namespace std;

int main(){
    int rowCount;
    cout<<"enter rowCount :";
    cin>>rowCount;

    int colCount;
    cout<<"enter colCount :";
    cin>>colCount;
    for(int row=0;row<rowCount;row++){
        if(row==0 || row==rowCount-1){
            for(int col=0;col<colCount;col++){
                cout<<"*";
            }
        }
        else{
            cout<<"*";
            for(int i=0;i<colCount-2;i++){
                cout<<" ";
            }
            cout<<"*";
        }
         cout<<endl;
    }

    return 0;
}
```



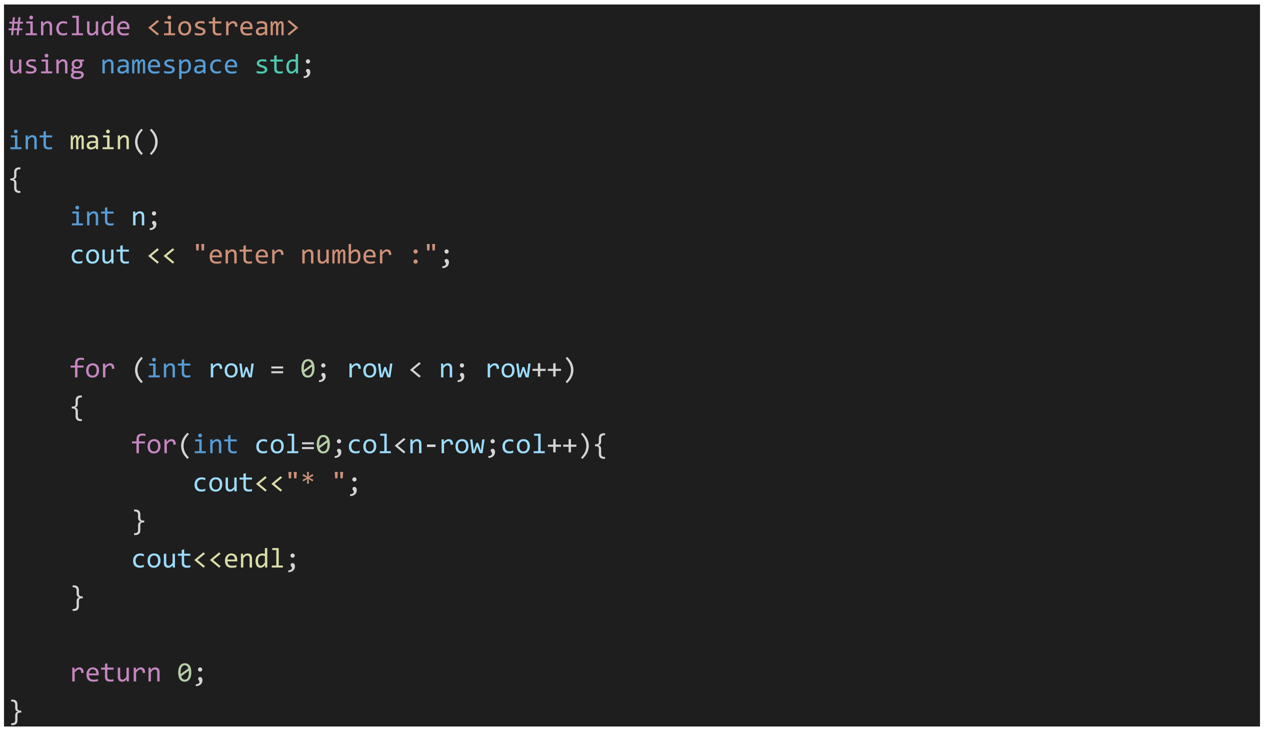
```
******
*    *
*    *
*    *
*    *
*    *
******
```



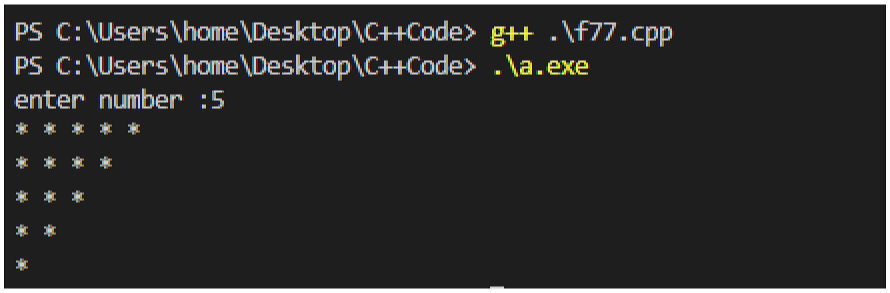
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<row+1;col++){
            cout<<"* ";
        }
        cout<<endl;
    }

    return 0;
}
```



```
enter number :5
*
* *
* * *
* * * *
* * * * *
```

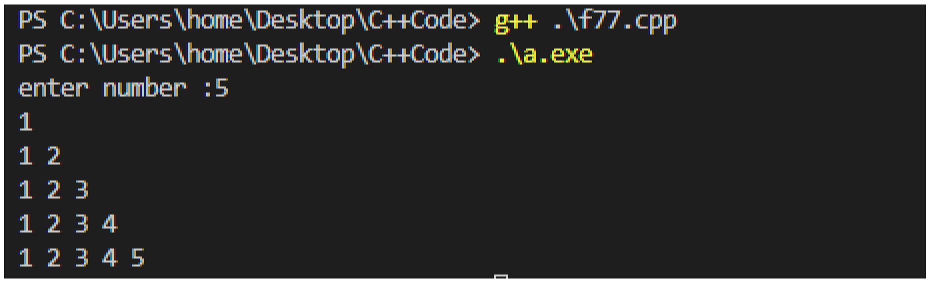


```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";


    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<n-row;col++){
            cout<<"* ";
        }
        cout<<endl;
    }

    return 0;
}
```

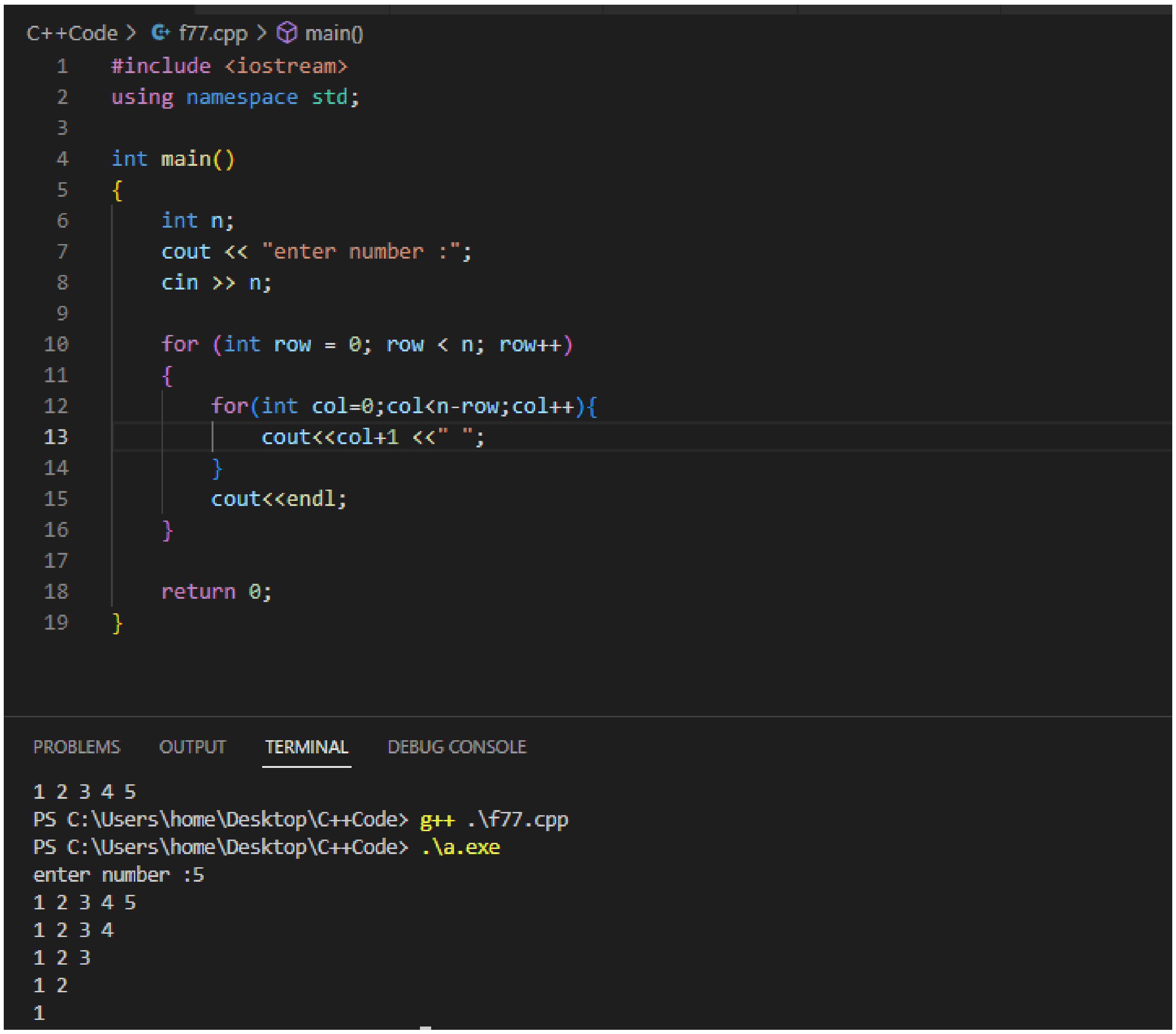


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
* * * * *
* * * *
* * *
* *
*
```

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

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<row+1;col++){
            cout<<col+1 <<" ";
        }
        cout<<endl;
    }

    return 0;
}
```



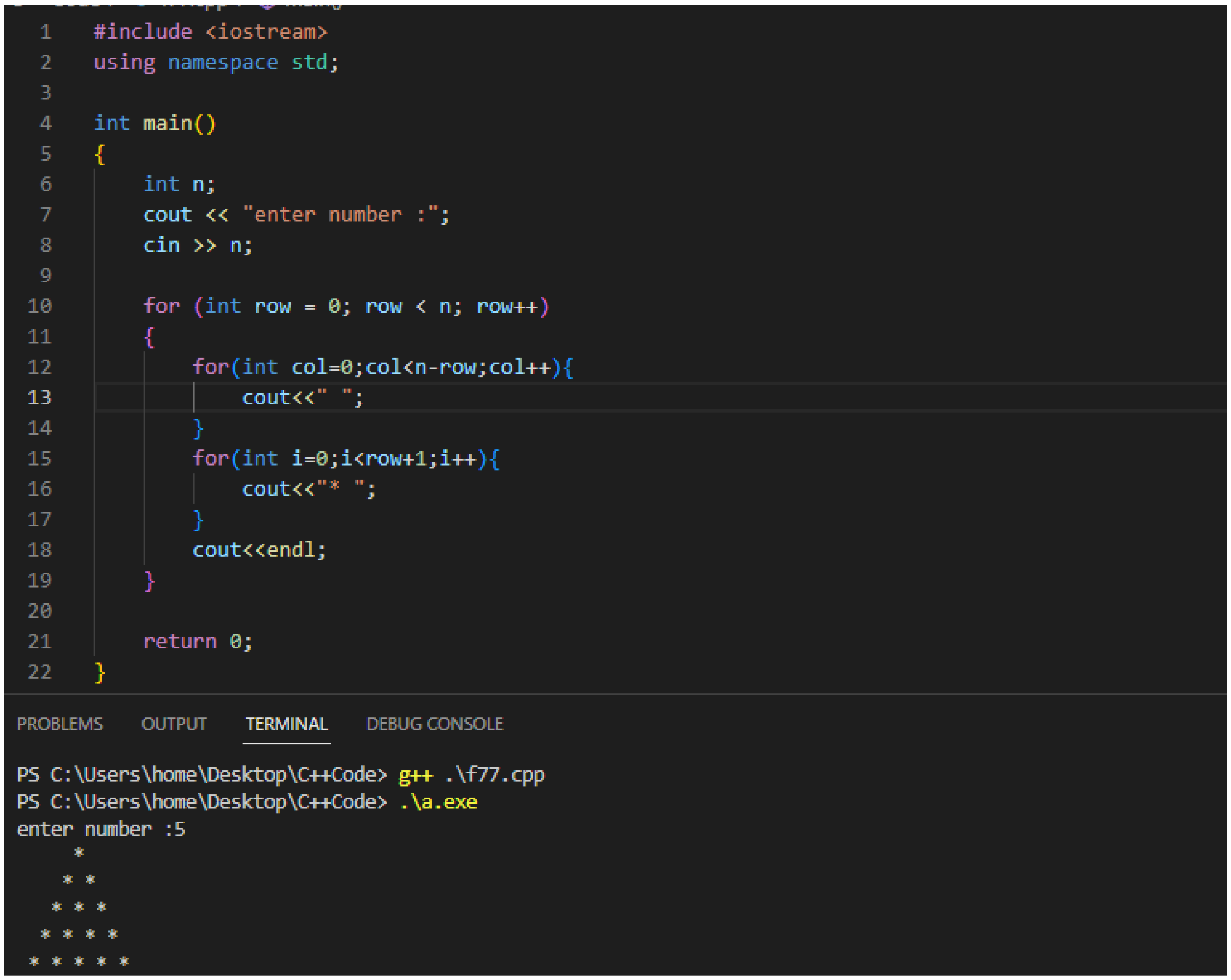
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```



C++Code > f77.cpp > main()

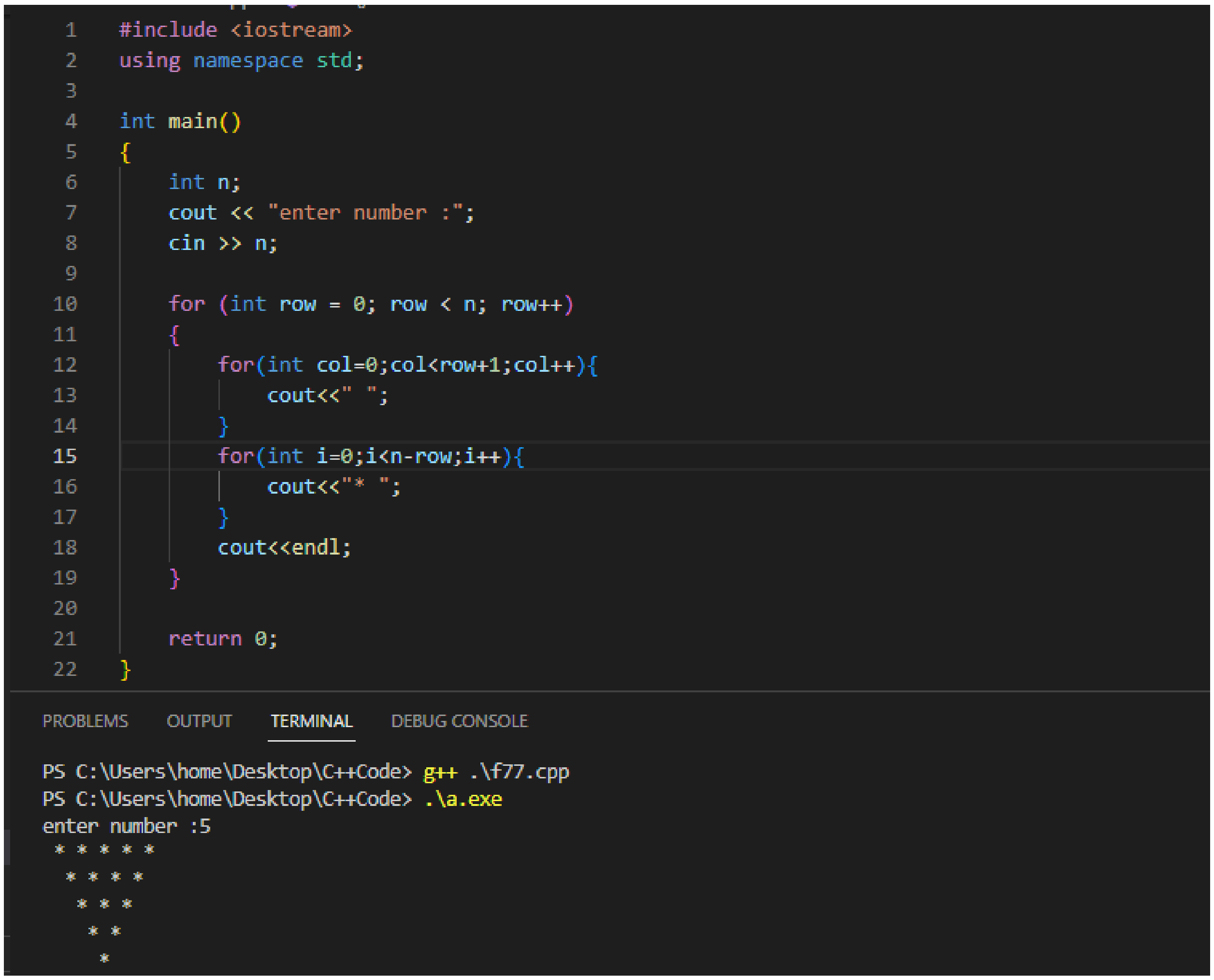
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<n-row;col++){
            cout<<col+1 <<" ";
        }
        cout<<endl;
    }

    return 0;
}
```

```
1 2 3 4 5
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```



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

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<n-row;col++){
            cout<<" ";
        }
        for(int i=0;i<row+1;i++){
            cout<<"* ";
        }
        cout<<endl;
    }

    return 0;
}
```

```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
     *
    * *
   * * *
  * * * *
 * * * * *
```

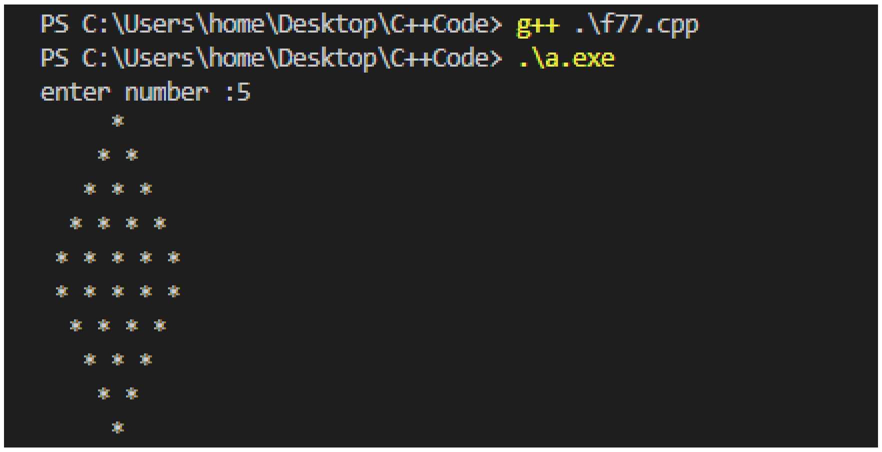


```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for (int row = 0; row < n; row++)
    {
        for(int col=0;col<row+1;col++){
            cout<<" ";
        }
        for(int i=0;i<n-row;i++){
            cout<<"* ";
        }
        cout<<endl;
    }

    return 0;
}
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

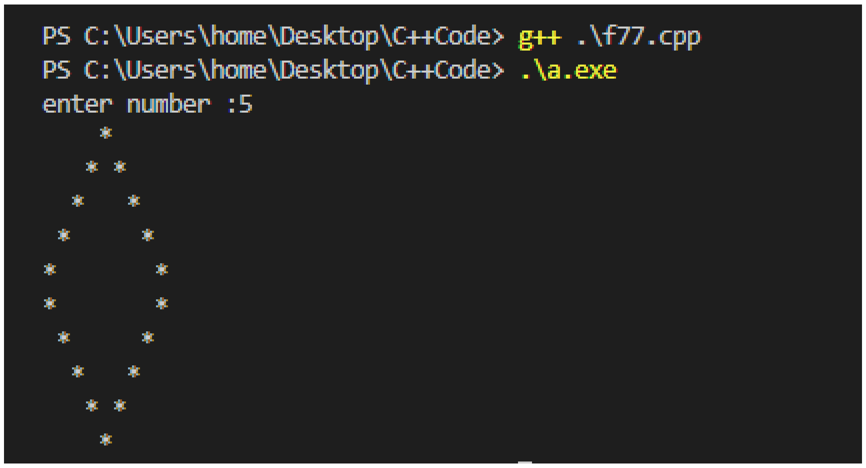
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
 * * * * *
  * * * *
   * * *
    * *
     *
```

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

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

   for(int row=0;row<n;row++){
    for(int col=0;col<n-row;col++){
        cout<<" ";
    }
    for(int i=0;i<row+1;i++){
        cout<<"* ";
    }
    cout<<endl;

   }
    for(int row=0;row<n;row++){
        for(int col=0;col<row+1;col++){
            cout<<" ";
        }
        for(int k=0;k<n-row;k++){
            cout<<"* ";
        }
        cout<<endl;
    }
    return 0;
}
```

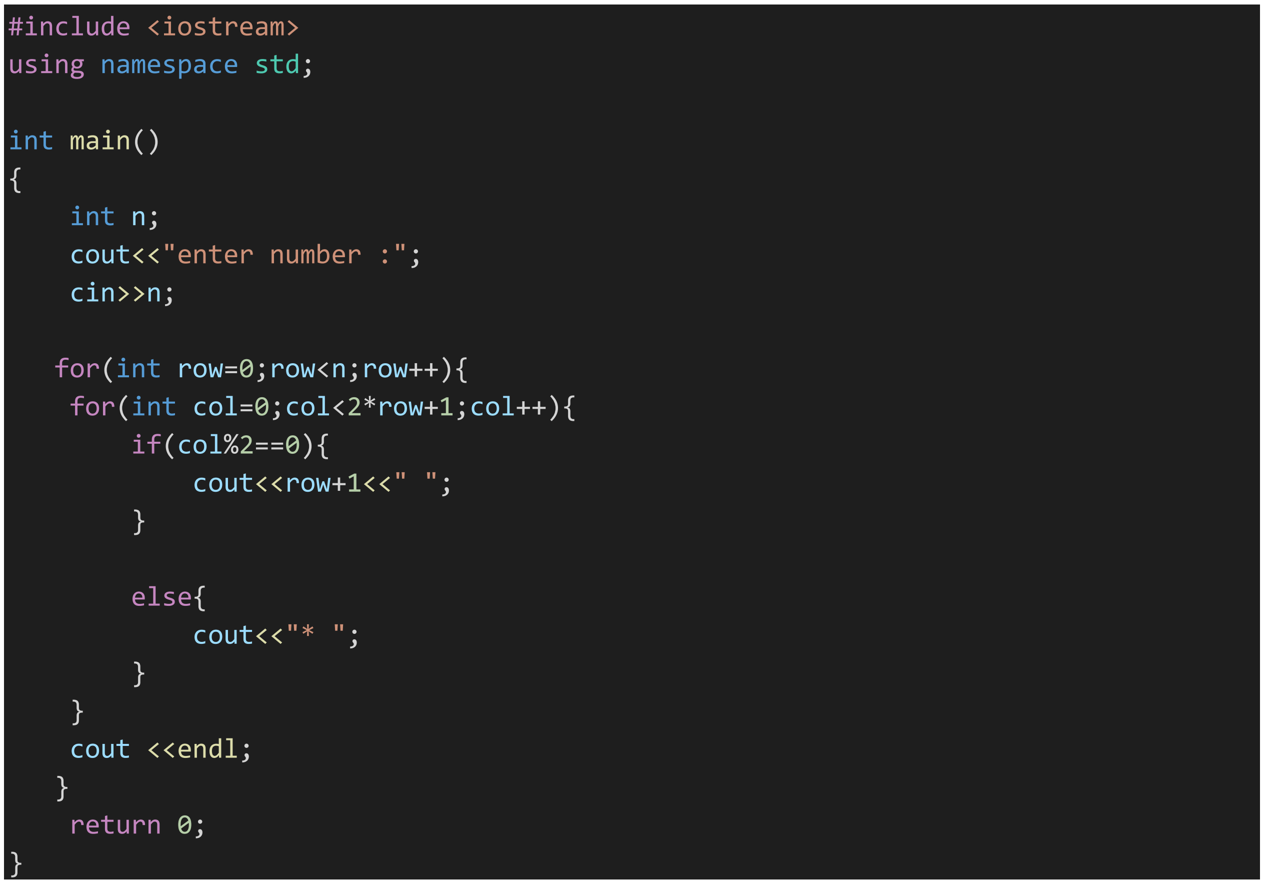


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
     *
    * *
   * * *
  * * * *
 * * * * *
 * * * * *
  * * * *
   * * *
    * *
     *
```

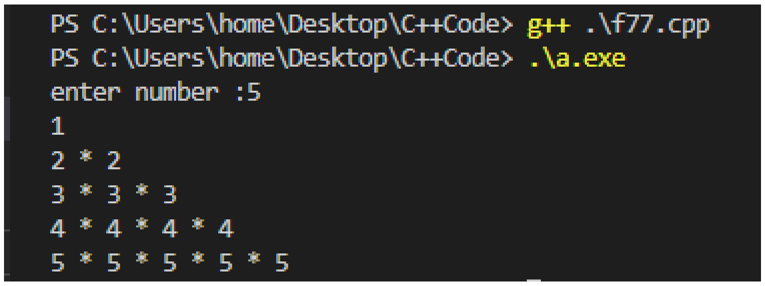
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;

    for(int row=0;row<n;row++){
        for(int col=0;col<n-row-1;col++){
            cout<<" ";
        }
        for(int i=0;i<2*row+1;i++){
            if(i==0){
                cout<<"*";
            }
            else if(i==2*row){
                cout<<"*";
            }
            else{
                cout<<" ";
            }
        }
        cout<<endl;
    }
     //2nd pyramid
        for(int row=0;row<n;row++){
            for(int col=0;col<row;col++){
                cout<<" ";
            }

            for(int i=0;i<2*n-2*row-1;i++){
                if(i==0 || i==2*n-2*row-2 ){
                    cout<<"*";
                }
                else{
                    cout<<" ";
                }
            }
             cout<<endl;
        }

    return 0;
}
```



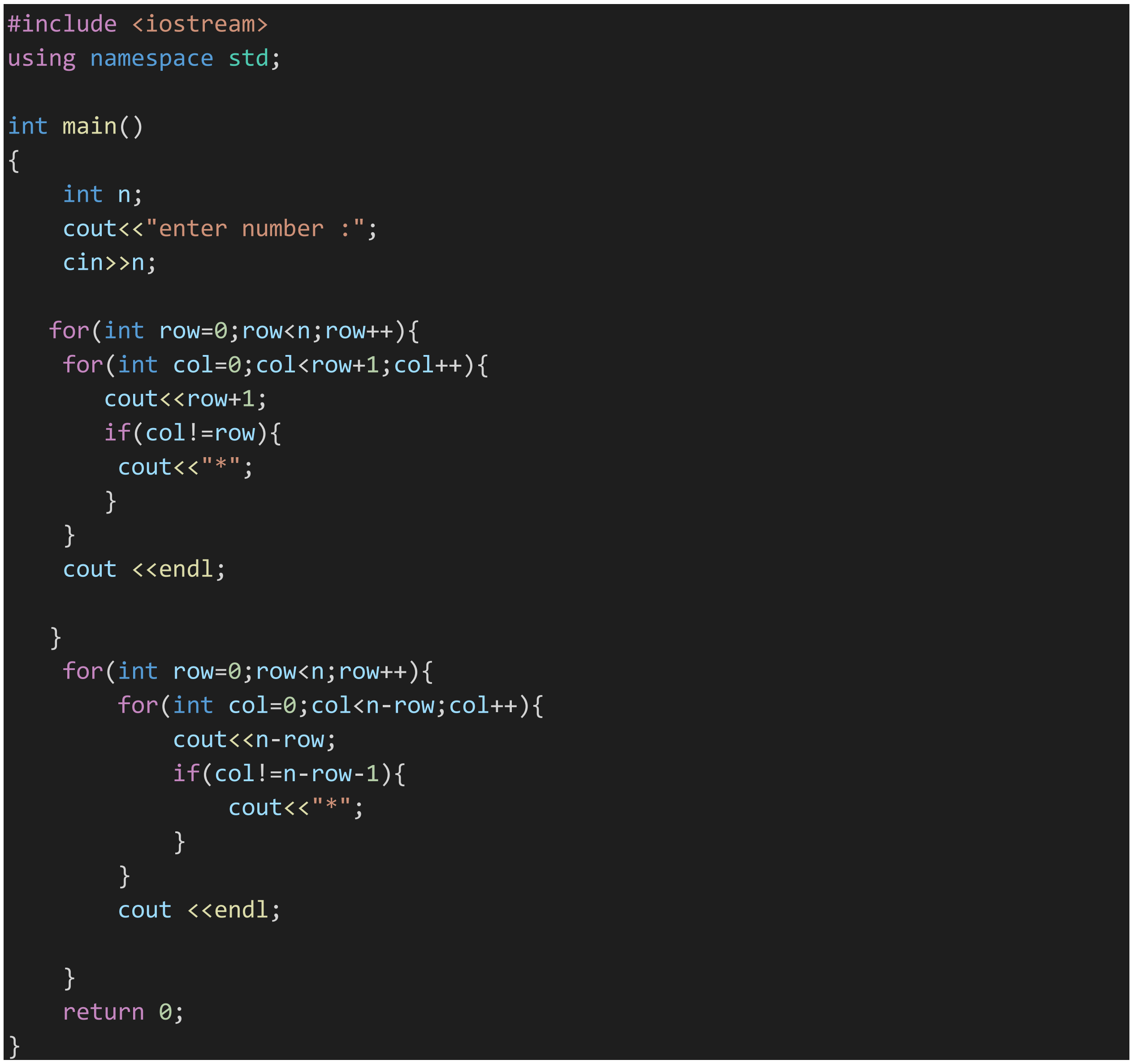
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
    *
   * *
  *   *
 *     *
*       *
*       *
 *     *
  *   *
   * *
    *
```



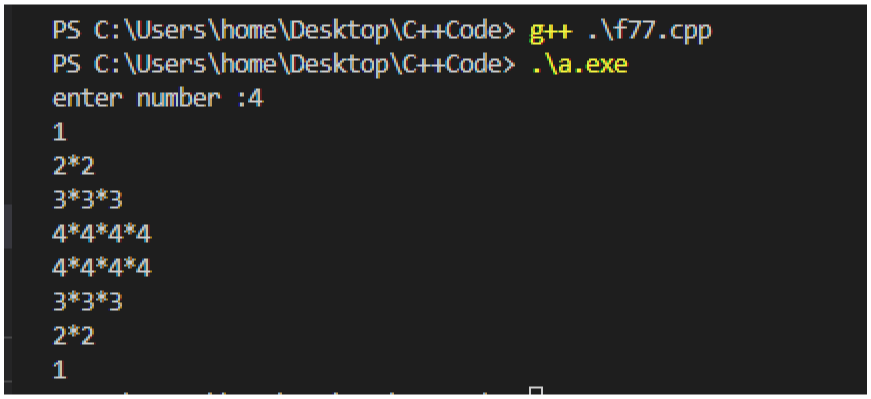
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout<<"enter number :";
    cin>>n;

   for(int row=0;row<n;row++){
    for(int col=0;col<2*row+1;col++){
        if(col%2==0){
            cout<<row+1<<" ";
        }

        else{
            cout<<"* ";
        }
    }
    cout <<endl;
   }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
1
2 * 2
3 * 3 * 3
4 * 4 * 4 * 4
5 * 5 * 5 * 5 * 5
```

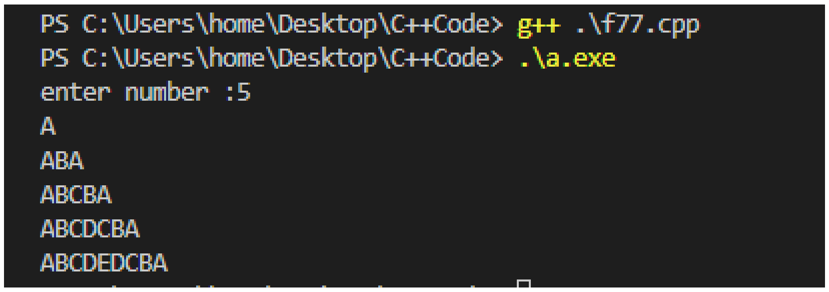


```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout<<"enter number :";
    cin>>n;

   for(int row=0;row<n;row++){
    for(int col=0;col<row+1;col++){
       cout<<row+1;
       if(col!=row){
        cout<<"*";
       }
    }
    cout <<endl;

   }
    for(int row=0;row<n;row++){
        for(int col=0;col<n-row;col++){
            cout<<n-row;
            if(col!=n-row-1){
                cout<<"*";
            }
        }
        cout <<endl;

    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :4
1
2*2
3*3*3
4*4*4*4
4*4*4*4
3*3*3
2*2
1
```

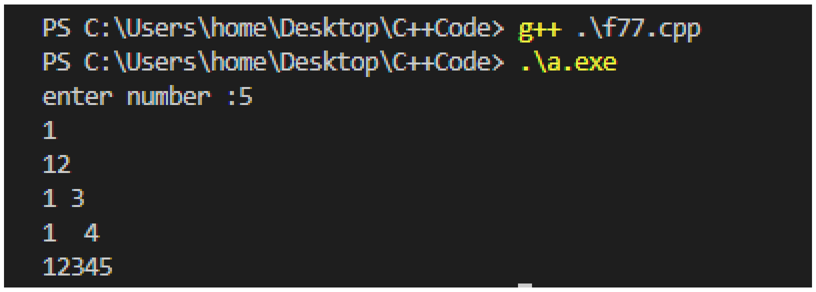
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;
    for (int row = 0; row < n; row++)
    {
        int col;
        for(col=0;col<row+1;col++){
            int ans=col+1;
            char ch= ans+'A'-1;
            cout<<ch;
        }
        for(int col=row;col>=1;col=col-1){
             int ans=col;
            char ch= ans+'A'-1;
            cout<<ch;
        }
        cout<<endl;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
A
ABA
ABCBA
ABCDCBA
ABCDEDCBA
```

Q) Numeric Hollow Half Pyramid

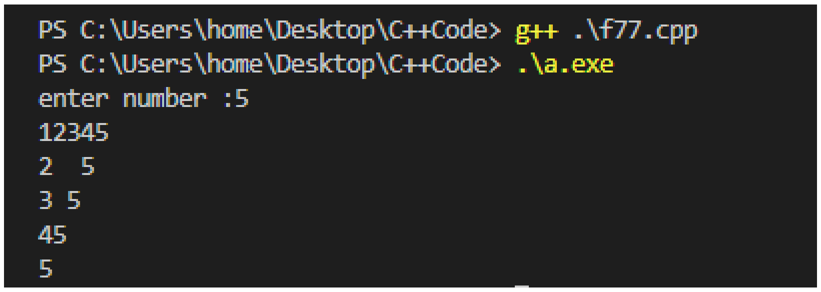
```cpp
#include <iostream>
using namespace std;

int main()
{
   int n;
   cout<<"enter number :";
   cin>>n;
   for(int r=0;r<n;r++){
    for(int c=0;c<r+1;c++){
        if(c==0|| c==r || r==n-1){
            cout<<c+1;
        }
        else{
            cout<<" ";
        }
    }
    cout<<endl;
   }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
1
12
1 3
1  4
12345
```

Q) Numeric Hollow Inverted Half Pyramid

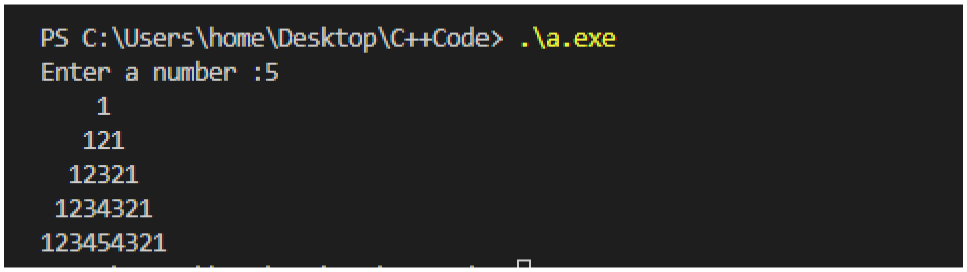
```cpp
#include <iostream>
using namespace std;

int main()
{
   int n;
   cout<<"enter number :";
   cin>>n;
   for(int r=0;r<n;r++){
    for(int c=r;c<n;c++){
        if(r==0|| c==r || c==n-1){
            cout<<c+1;
        }
        else{
            cout<<" ";
        }
    }
    cout<<endl;
   }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
12345
2  5
3 5
45
5
```

Q) Numeric Palindrome Equilateral Pyramid

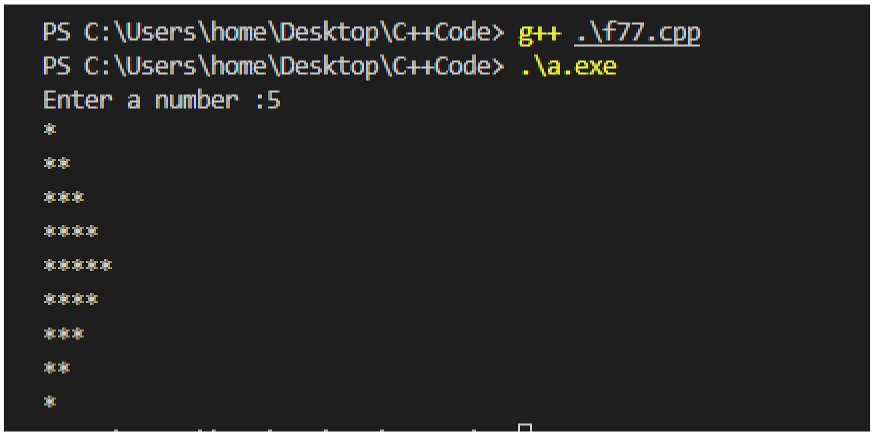
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "Enter a number :";
    cin >> n;
    int k = n;
    for (int r = 0; r < n; r++)
    {
        int count = 1;
        for (int c = 0; c < k; c++)
        {
            if (c < n - r - 1)
            {
                cout << " ";
            }
            else if (c <= n - 1)
            {
                cout << count;
                count++;
            }
            else if (c == n)
            {
                count = count - 2;
                cout << count;
                count--;
            }
            else
            {
                cout << count;
                count--;
            }
        }
        cout << endl;
        k++;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
Enter a number :5
    1
   121
  12321
 1234321
123454321
```

Q) solid half pyramid

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

int main()
{
    int n;
    cout << "Enter a number :";
    cin >> n;
    for(int r=0;r<n;r++){
        for(int c=0;c<r+1;c++){
            cout<<"*";
        }
        cout<<endl;
    }
     for(int r=0;r<n-1;r++){
        for(int c=0;c<n-r-1;c++){
            cout<<"*";
        }
         cout<<endl;
    }
    return 0;
}
```

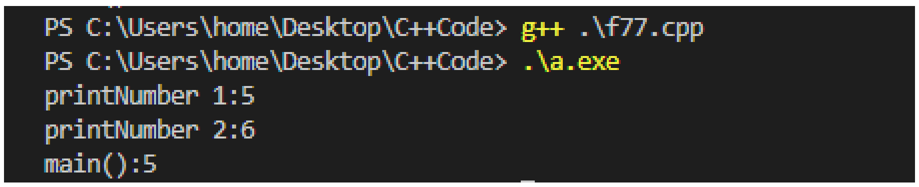


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
Enter a number :5
*
**
***
****
*****
****
***
**
*
```

Function:

Example of pass by value

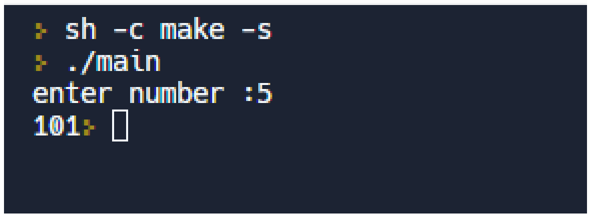
```cpp
#include <iostream>
using namespace std;

void printNumber(int a){
    cout<<"printNumber 1:"<<a<<endl;;
    a++;
    cout<<"printNumber 2:"<<a<<endl;
}
int main()
{
    int a=5;
    printNumber(a);
    cout<<"main():"<<a<<endl;;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
printNumber 1:5
printNumber 2:6
main():5
```

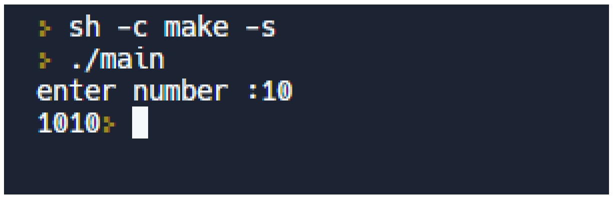
Q)Decimal to Binary

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

int decmailToBinary(int n) {
  int ans = 0;
  int i = 0;
  while (n != 0) {
    int digit = n & 1;
    ans = (digit * pow(10, i)) + ans;
    n = n >> 1;
    i++;
  }
  return ans;
}

int main() {
  int n;
  cout << "enter number :";
  cin >> n;
  int ans = decmailToBinary(n);
   cout << ans;
}
```



```
 sh -c make -s
 ./main
enter number :5
101
```

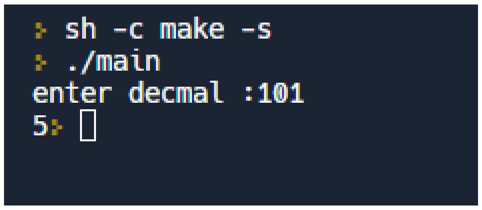
```cpp
#include <cmath>
#include <iostream>
using namespace std;

int decmalToBinary(int n) {
  int ans = 0;
  int i = 0;
  while (n != 0) {
    int digit = n % 2;
    ans = (digit * pow(10, i)) + ans;
    n = n / 2;
    i++;
  }
  return ans;
}

int main() {
  int n;
  cout << "enter number :";
  cin >> n;
  int ans = decmalToBinary(n);
  cout << ans;
}
```



```
› sh -c make -s
› ./main
enter number :10
1010›
```

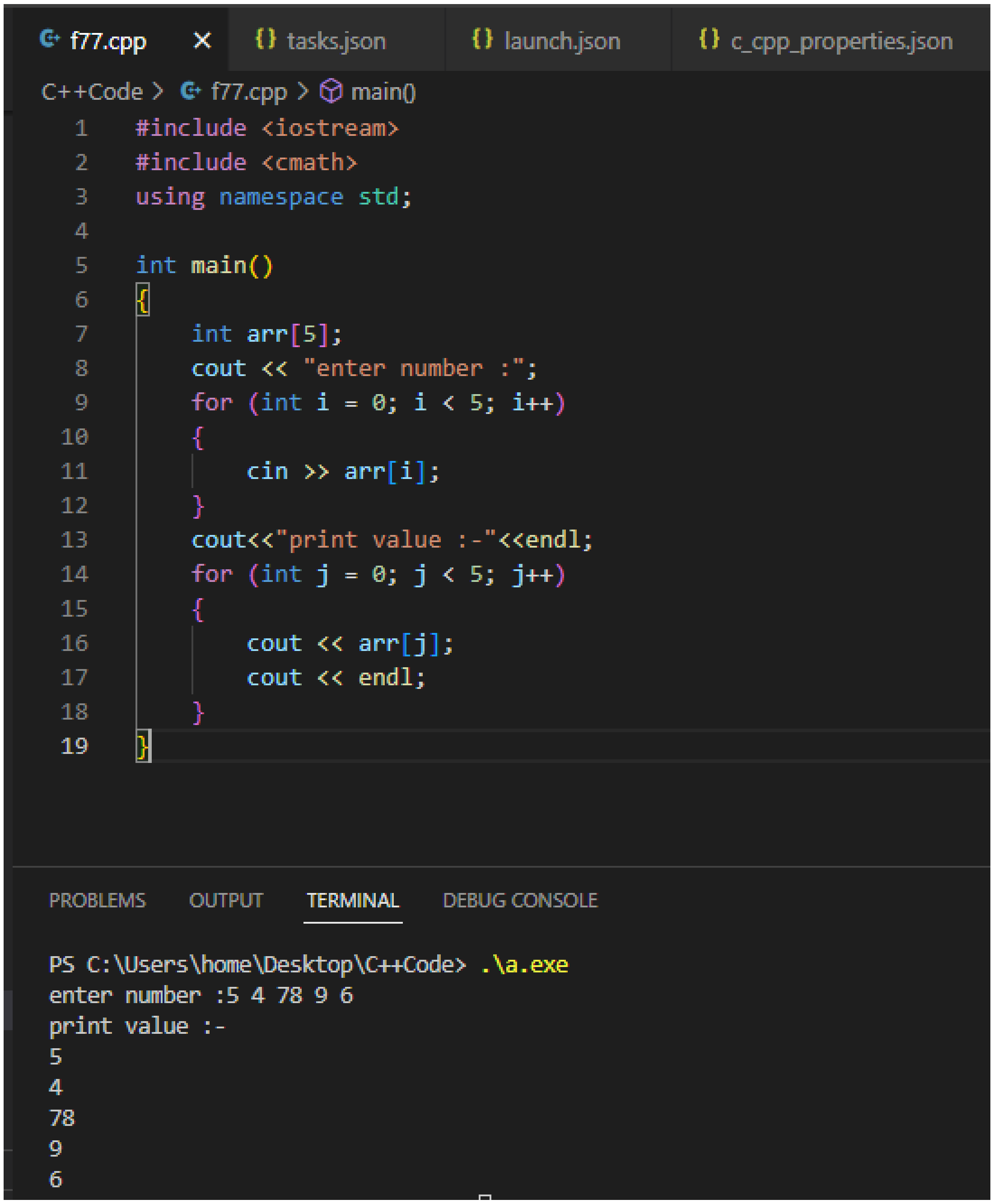
Q)Binary to Decimal

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

int binaryToDecimal(int n){
   int ans = 0;
  int i = 0;
  while (n != 0) {
    int digit = n %10;
    ans = (digit * pow(2, i)) + ans;
   n=n/10;
    i++;
  }
  return ans;
}

int main() {
  int n;
  cout << "enter decimal :";
  cin >> n;
  int ans = binaryToDecimal(n);
  cout << ans;
}
```



```
 > sh -c make -s
 > ./main
enter decmal :101
5> 
```

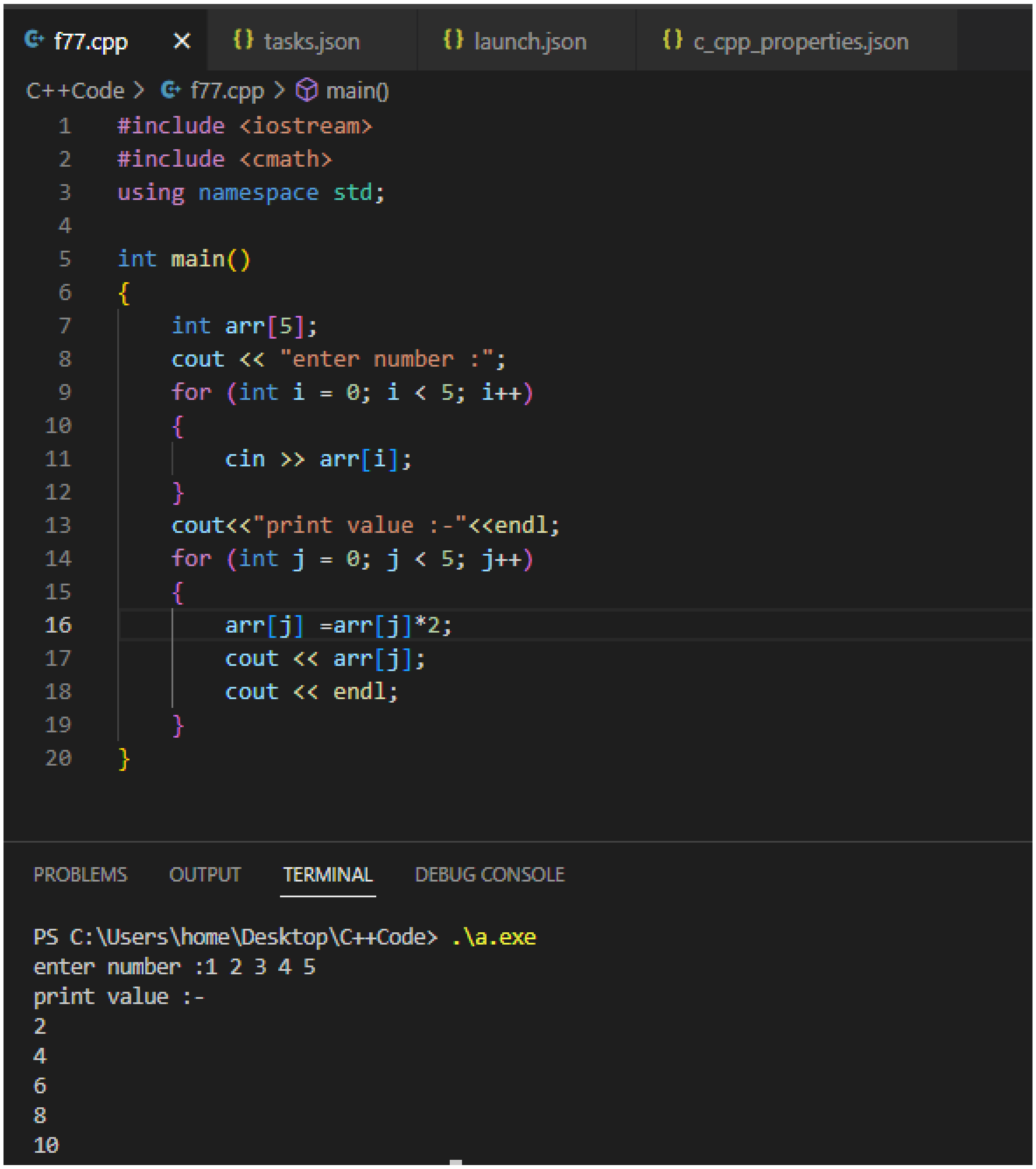
Q)Take 5 element in array and print value



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

int main()
{
    int arr[5];
    cout << "enter number :";
    for (int i = 0; i < 5; i++)
    {
        cin >> arr[i];
    }
    cout<<"print value :-"<<endl;
    for (int j = 0; j < 5; j++)
    {
        cout << arr[j];
        cout << endl;
    }
}
```

```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5 4 78 9 6
print value :-
5
4
78
9
6
```

Q)Take 5 element in array and print double value

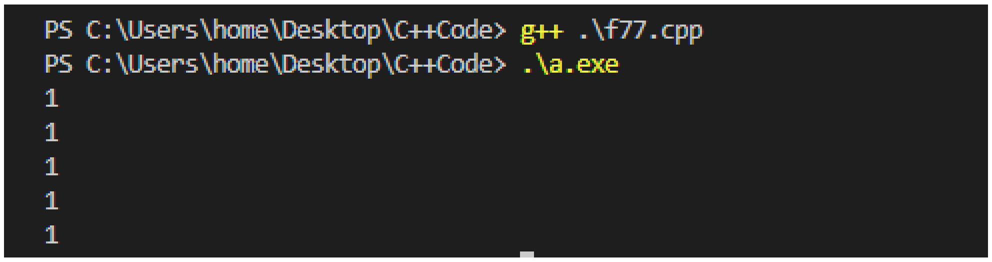


```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    int arr[5];
    cout << "enter number :";
    for (int i = 0; i < 5; i++)
    {
        cin >> arr[i];
    }
    cout<<"print value :-"<<endl;
    for (int j = 0; j < 5; j++)
    {
        arr[j] =arr[j]*2;
        cout << arr[j];
        cout << endl;
    }
}
```

```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :1 2 3 4 5
print value :-
2
4
6
8
10
```

Q)Assign 1 value to all array value

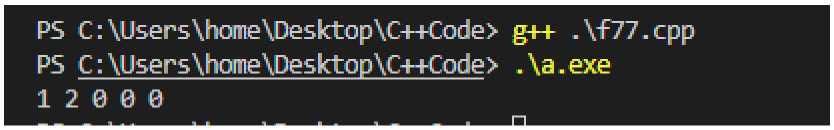
```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    int arr[] = {1, 2, 3, 4, 5};
    for (int i = 0; i < 5; i++)
    {
        arr[i] = 1;
        cout<<arr[i]<<endl;
    }

}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1
1
1
1
1
```

Q)

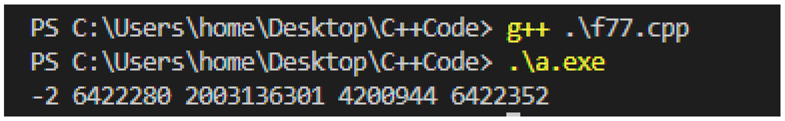
```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    int arr[5] = {1, 2};
    for (int i = 0; i < 5; i++)
    {

        cout<<arr[i]<<" ";
    }

}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 0 0 0
```

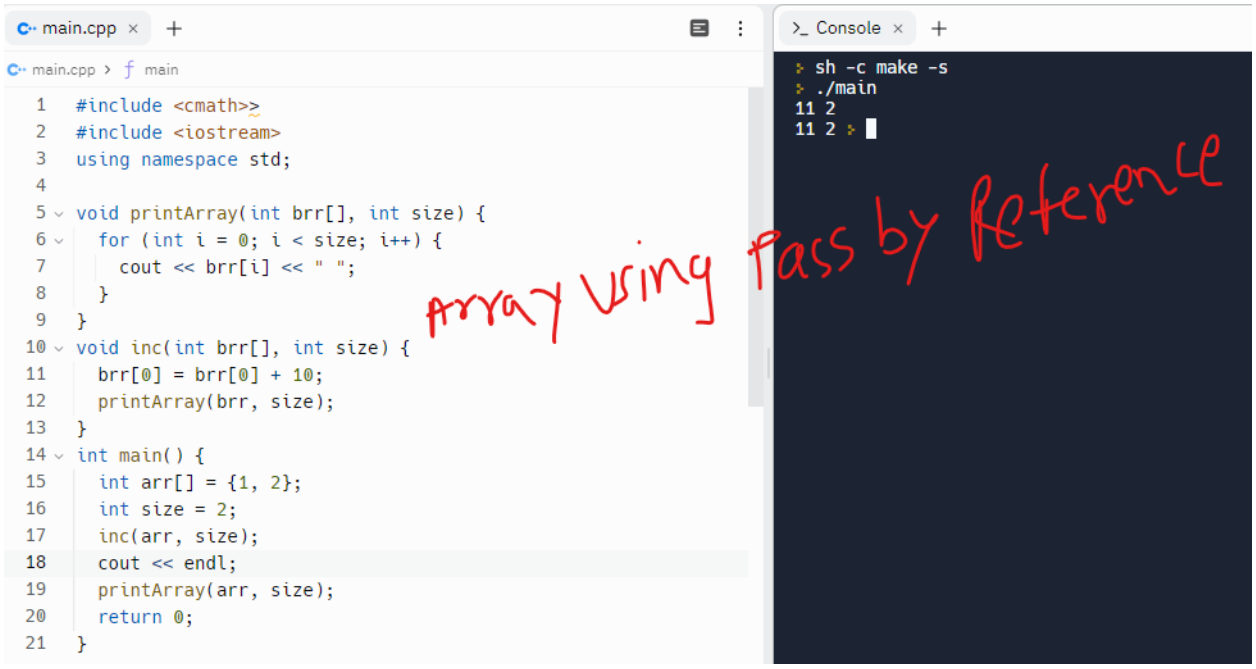
Q) Garbage value

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

int main()
{
    int arr[5];
    for (int i = 0; i < 5; i++)
    {
        cout << arr[i] << " ";
    }
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
-2 6422280 2003136301 4200944 6422352
```

Q) Array using Pass by reference



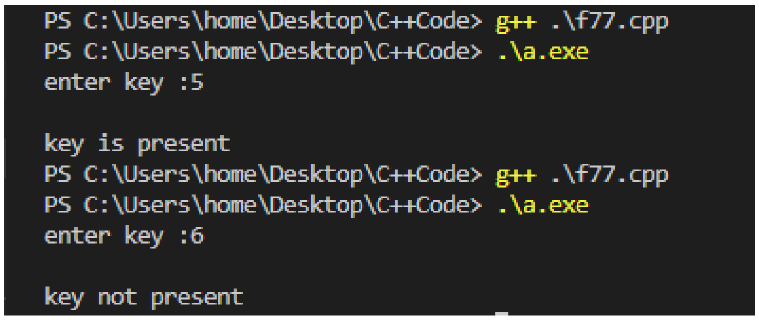
```cpp
#include <cmath>>
#include <iostream>
using namespace std;

void printArray(int brr[], int size) {
  for (int i = 0; i < size; i++) {
    cout << brr[i] << " ";
  }
}
void inc(int brr[], int size) {
  brr[0] = brr[0] + 10;
  printArray(brr, size);
}
int main() {
  int arr[] = {1, 2};
  int size = 2;
  inc(arr, size);
  cout << endl;
  printArray(arr, size);
  return 0;
}
```

```
sh -c make -s
./main
11 2
11 2
```

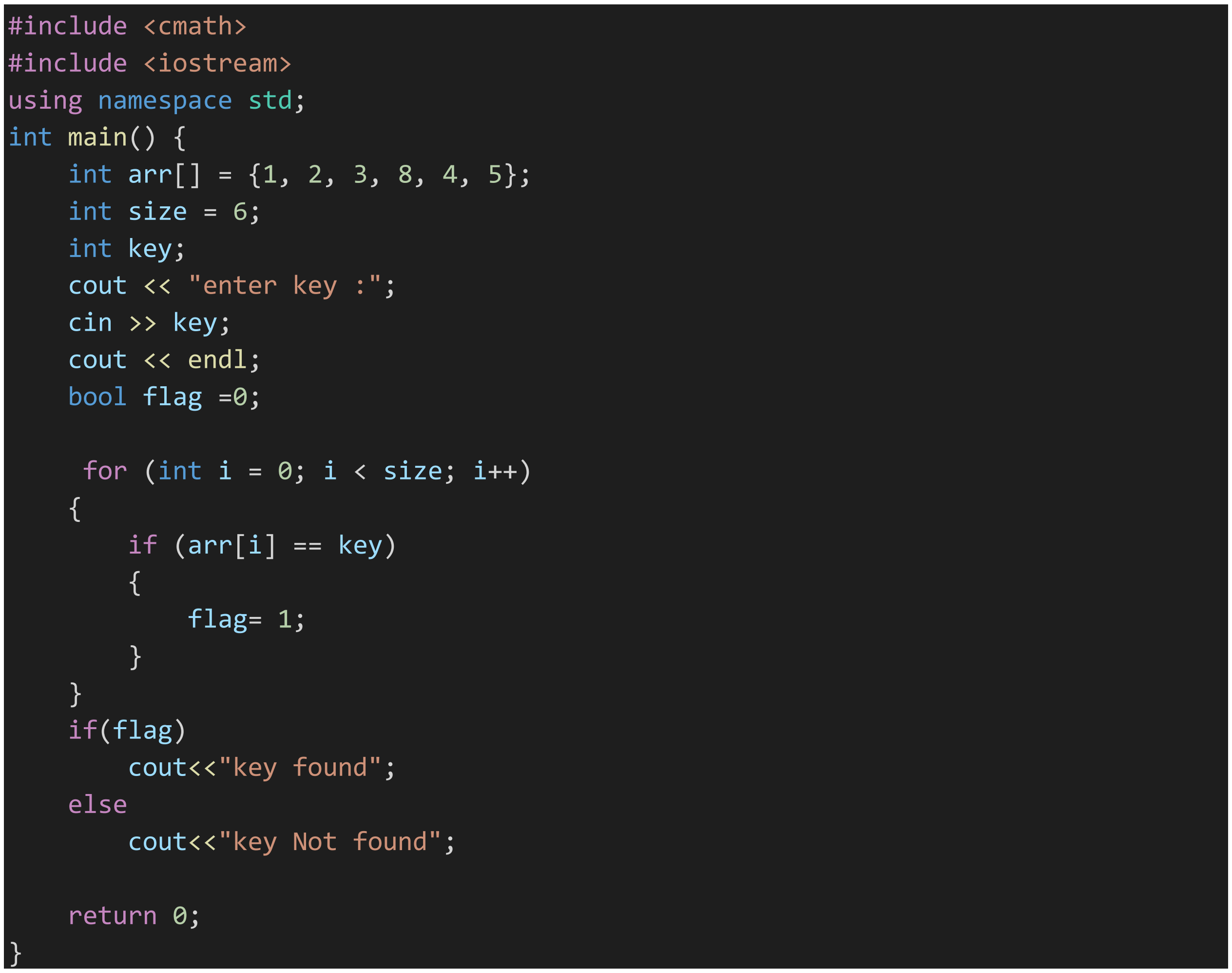
Array using Pass by Reference

Q) Linear search for find element in array

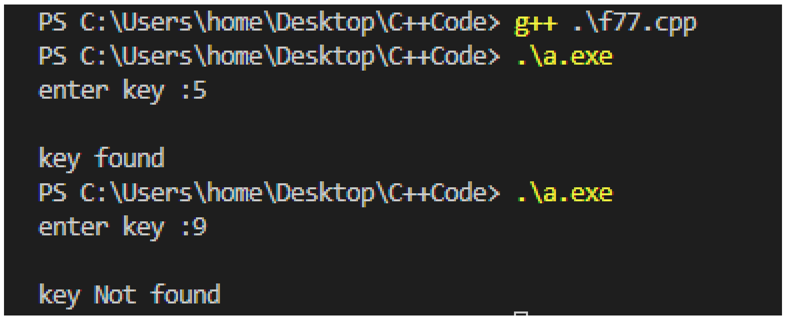
```cpp
#include <cmath>
#include <iostream>
using namespace std;

bool findElement(int arr[], int size, int key)
{
    for (int i = 0; i < size; i++)
    {
        if (arr[i] == key)
        {
            return true;
        }
    }
    return false;
}

int main()
{
    int arr[] = {1, 2, 3, 8, 4, 5};
    int size = 6;
    int key;
    cout << "enter key :";
    cin >> key;
    cout << endl;
    if (findElement(arr, size, key))
    {
        cout << "key is present";
    }
    else
    {
        cout << "key not present";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter key :5

key is present
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter key :6

key not present
```

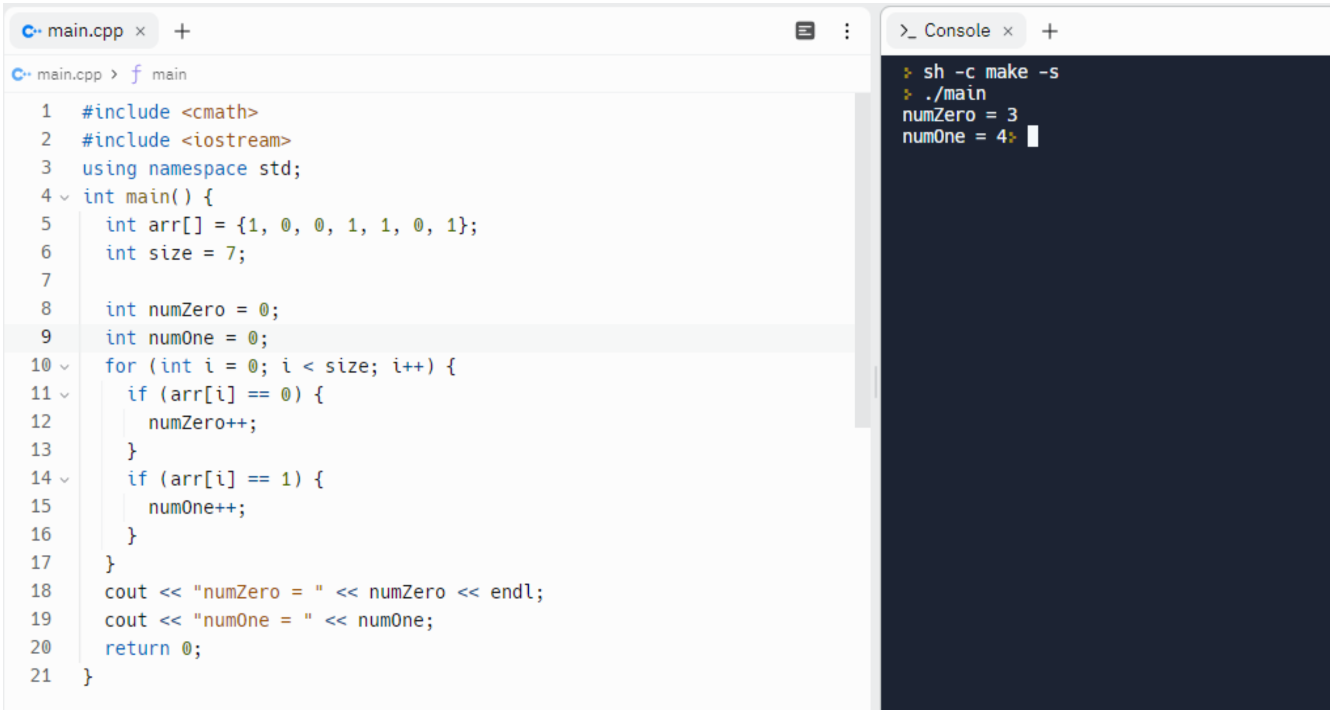


```cpp
#include <cmath>
#include <iostream>
using namespace std;
int main() {
    int arr[] = {1, 2, 3, 8, 4, 5};
    int size = 6;
    int key;
    cout << "enter key :";
    cin >> key;
    cout << endl;
    bool flag =0;

     for (int i = 0; i < size; i++)
    {
        if (arr[i] == key)
        {
            flag= 1;
        }
    }
    if(flag)
        cout<<"key found";
    else
        cout<<"key Not found";

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter key :5

key found
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter key :9

key Not found
```

Q) find 0's and 1's in array



main.cpp

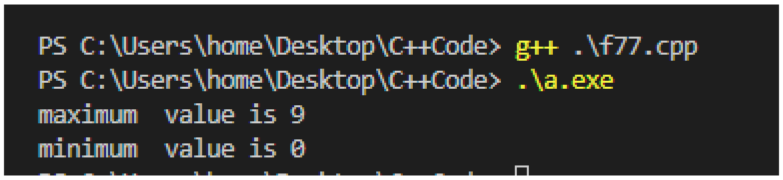
```cpp
#include <cmath>
#include <iostream>
using namespace std;
int main() {
  int arr[] = {1, 0, 0, 1, 1, 0, 1};
  int size = 7;

  int numZero = 0;
  int numOne = 0;
  for (int i = 0; i < size; i++) {
    if (arr[i] == 0) {
      numZero++;
    }
    if (arr[i] == 1) {
      numOne++;
    }
  }
  cout << "numZero = " << numZero << endl;
  cout << "numOne = " << numOne;
  return 0;
}
```

Console

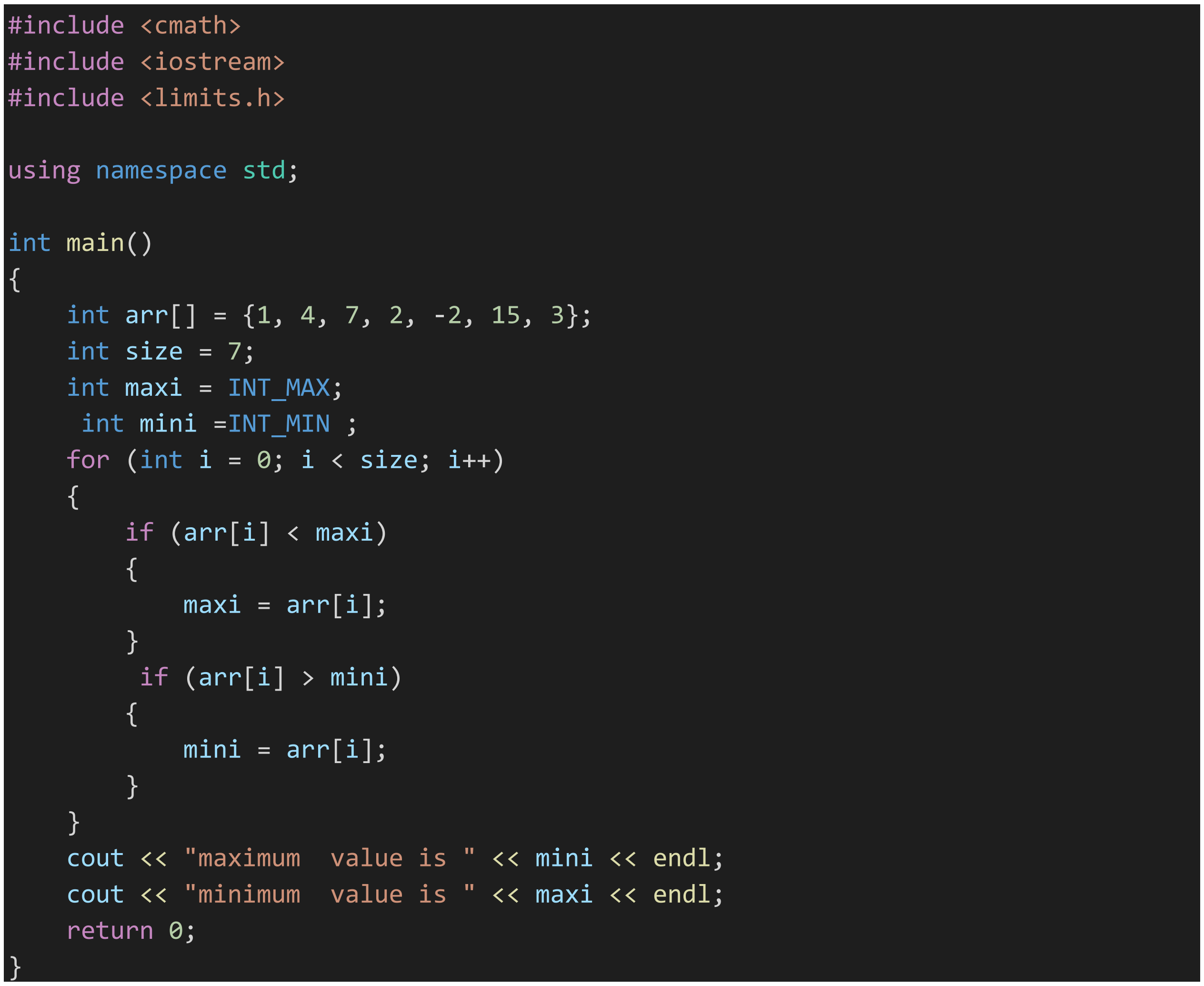
```
› sh -c make -s
› ./main
numZero = 3
numOne = 4›
```

Q) find maximum and minimum number

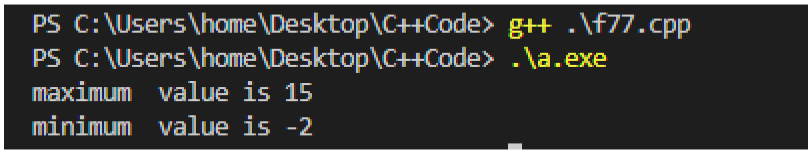
```cpp
#include <cmath>
#include <iostream>
#include <limits.h>

using namespace std;

int main()
{
    int arr[] = {1, 4, 7, 2, 0, 9, 3};
    int size = 7;
    int maxi = INT_MIN;

    for (int i = 0; i < size; i++)
    {
        if (arr[i] > maxi)
        {
            maxi = arr[i];
        }
    }
    cout << "maximum  value is " << maxi << endl;
    int mini = INT_MAX;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] < mini)
        {
            mini = arr[i];
        }
    }
    cout << "minimum  value is " << mini << endl;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
maximum  value is 9
minimum  value is 0
```



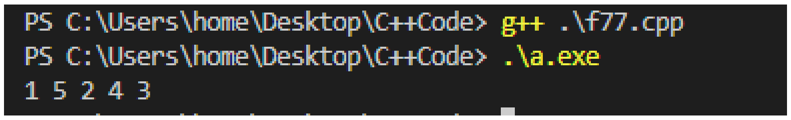
```cpp
#include <cmath>
#include <iostream>
#include <limits.h>

using namespace std;

int main()
{
    int arr[] = {1, 4, 7, 2, -2, 15, 3};
    int size = 7;
    int maxi = INT_MAX;
     int mini =INT_MIN ;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] < maxi)
        {
            maxi = arr[i];
        }
         if (arr[i] > mini)
        {
            mini = arr[i];
        }
    }
    cout << "maximum  value is " << mini << endl;
    cout << "minimum  value is " << maxi << endl;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
maximum  value is 15
minimum  value is -2
```

Q) Extreme print array

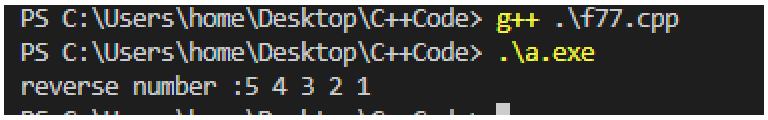
```cpp
#include <cmath>
#include <iostream>

using namespace std;

int main()
{
    int arr[] = {1, 2, 3, 4, 5};
    int size = 5;
    int start = 0;
    int end = size - 1;
    while (start <= end)
    {
        if (start == end)
        {
            cout << arr[start] << " ";
        }
        else
        {
            cout << arr[start] << " ";
            cout << arr[end] << " ";
        }
        start++;
        end--;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 5 2 4 3
```

Q) Reverse Number

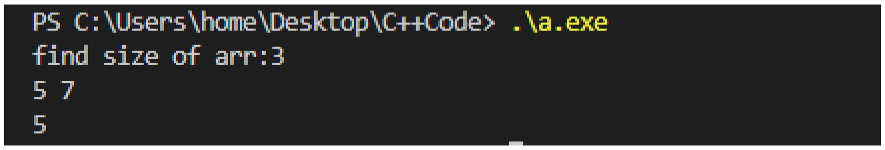
```cpp
#include <cmath>
#include <iostream>

using namespace std;

int main()
{
    int arr[] = {1, 2, 3, 4, 5};
    int size = 5;
    int start = 0;
    int end = size - 1;
    while (start <= end)
    {
        swap(arr[start], arr[end]);
        start++;
        end--;
    }
    cout << "reverse number :";
    for (int i = 0; i < size; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```



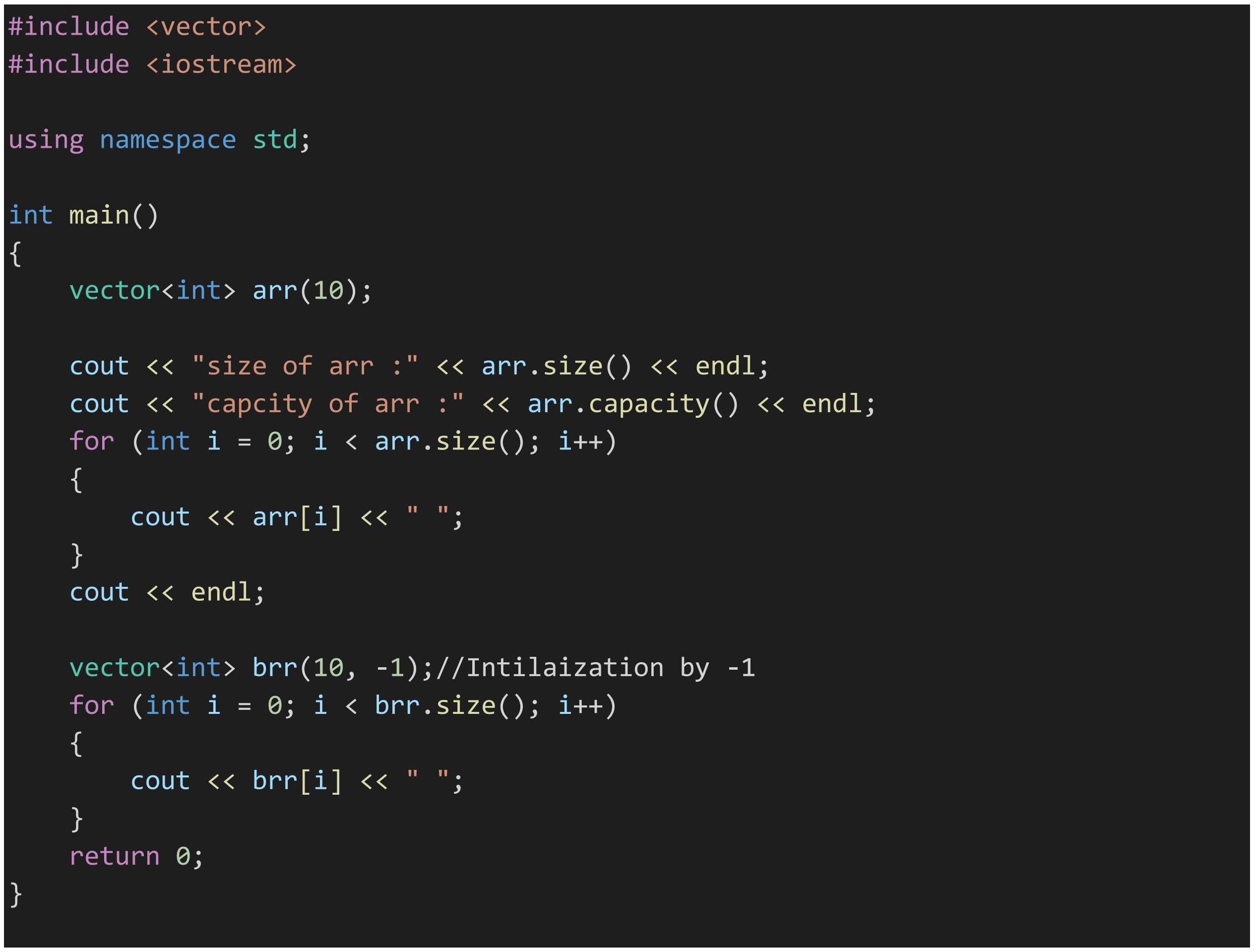
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
reverse number :5 4 3 2 1
```

Vector

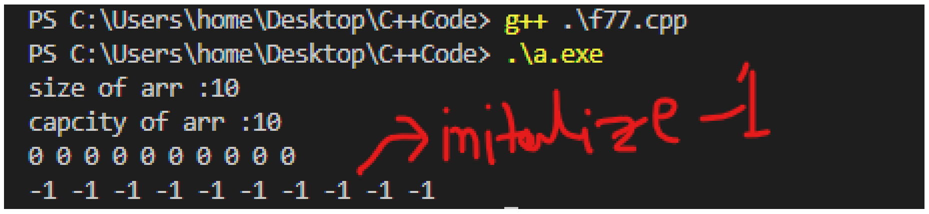
```cpp
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    vector<int> arr;
    int ans = (sizeof(arr) / sizeof(int));
    cout << "find size of arr:" << ans << endl;

    arr.push_back(5);
    arr.push_back(7);
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;

    arr.pop_back();
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
find size of arr:3
5 7
5
```

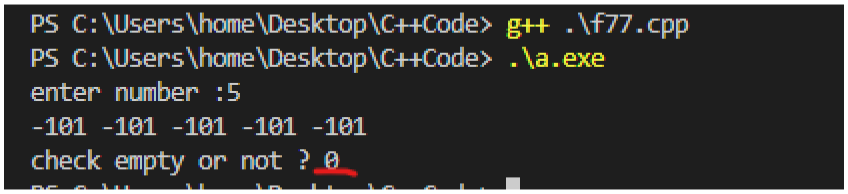


```cpp
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    vector<int> arr(10);

    cout << "size of arr :" << arr.size() << endl;
    cout << "capcity of arr :" << arr.capacity() << endl;
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;

    vector<int> brr(10, -1);//Intilaization by -1
    for (int i = 0; i < brr.size(); i++)
    {
        cout << brr[i] << " ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
size of arr :10
capcity of arr :10
0 0 0 0 0 0 0 0 0 0
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1
```

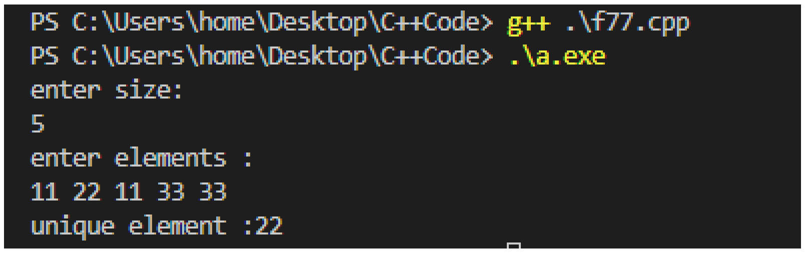
```cpp
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    int n;
    cout << "enter number :";
    cin >> n;
    vector<int> arr(n, -101);
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << " ";
    }
    cout<<endl;
    cout<<"check empty or not ? "<<arr.empty();//0 means false ,1 means true
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :5
-101 -101 -101 -101 -101
check empty or not ? 0
```

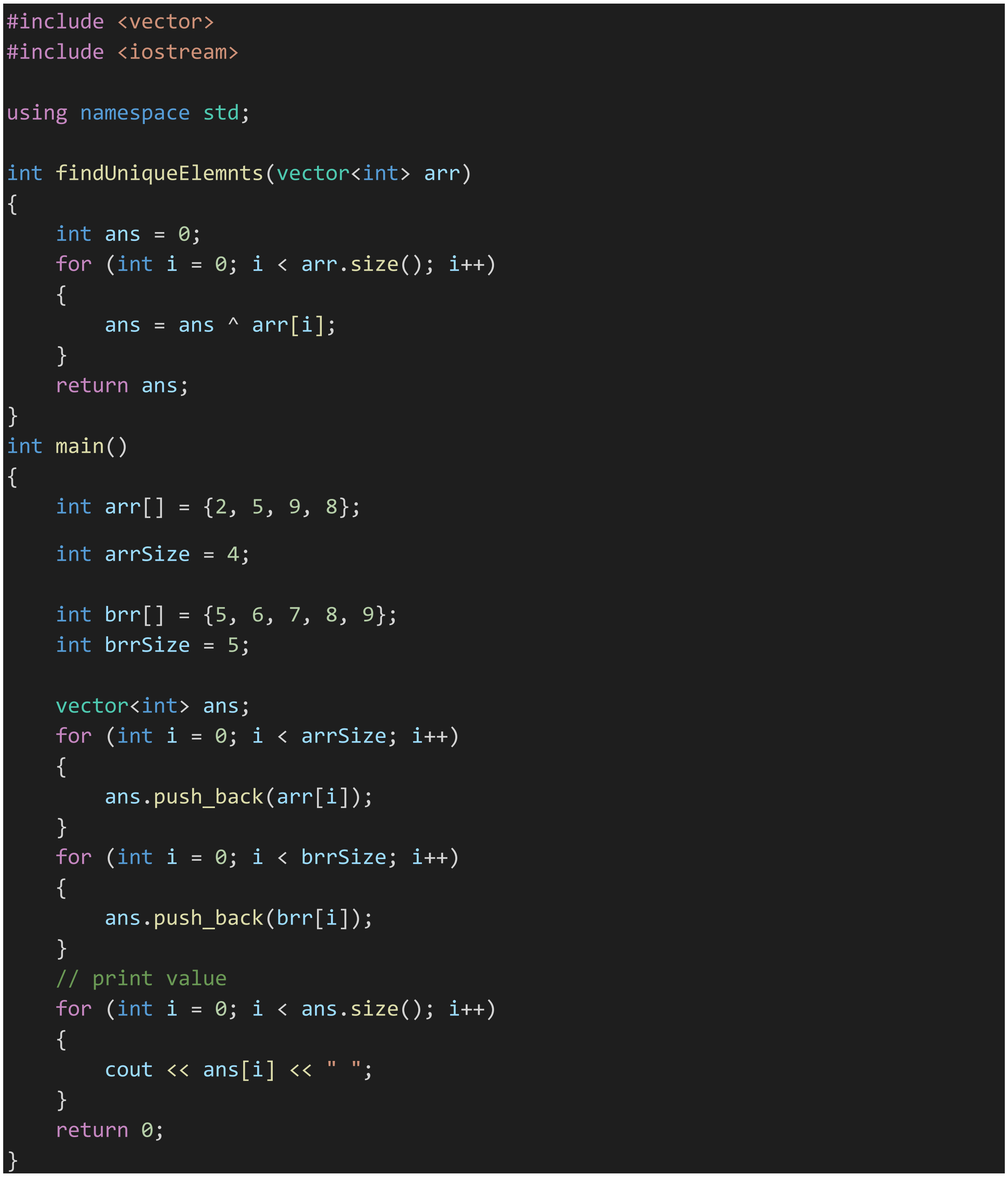
Q) Find Unique Elements

```cpp
#include <vector>
#include <iostream>

using namespace std;

int findUniqueElemnts(vector<int> arr)
{
    int ans = 0;
    for (int i = 0; i < arr.size(); i++)
    {
        ans = ans ^ arr[i];
    }
    return ans;
}
int main()
{
    int n;
    cout << "enter size: " << endl;
    cin >> n;
    vector<int> arr(n);
    cout << "enter elements :" << endl;
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
    int unique = findUniqueElemnts(arr);
    cout << "unique element :" << unique;
    return 0;
}
```

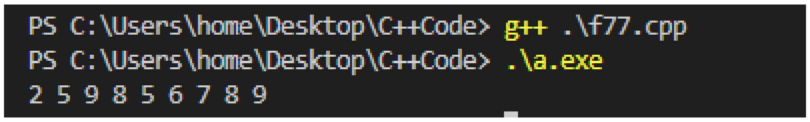


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter size:
5
enter elements :
11 22 11 33 33
unique element :22
```

Q) Add two array using Union Concept



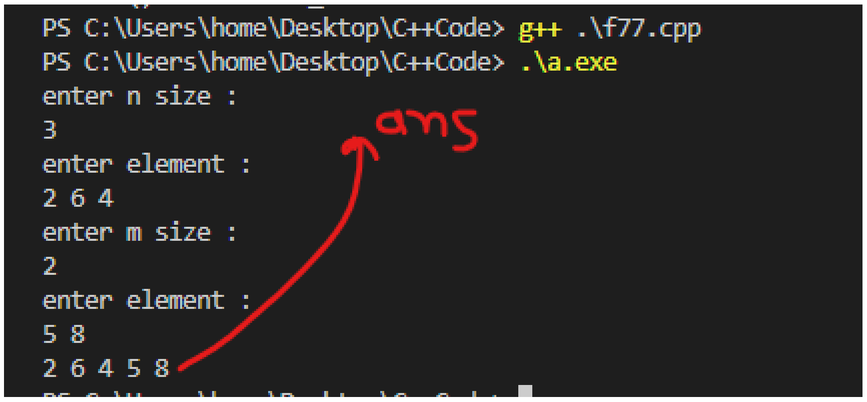
```cpp
#include <vector>
#include <iostream>

using namespace std;

int findUniqueElemnts(vector<int> arr)
{
    int ans = 0;
    for (int i = 0; i < arr.size(); i++)
    {
        ans = ans ^ arr[i];
    }
    return ans;
}
int main()
{
    int arr[] = {2, 5, 9, 8};

    int arrSize = 4;

    int brr[] = {5, 6, 7, 8, 9};
    int brrSize = 5;

    vector<int> ans;
    for (int i = 0; i < arrSize; i++)
    {
        ans.push_back(arr[i]);
    }
    for (int i = 0; i < brrSize; i++)
    {
        ans.push_back(brr[i]);
    }
    // print value
    for (int i = 0; i < ans.size(); i++)
    {
        cout << ans[i] << " ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
2 5 9 8 5 6 7 8 9
```

Q) Union all using vector

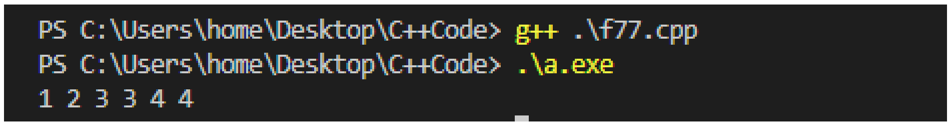
```
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    int n;
    cout<<"enter n size :"<<endl;
    cin>>n;
    vector<int> arr(n);
    cout<<"enter element :"<<endl;
    for(int i=0;i<arr.size();i++){
        cin>>arr[i];
    }

    int m;
    cout<<"enter m size :"<<endl;
    cin>>m;
    vector<int> brr(m);
    cout<<"enter element :"<<endl;
    for(int i=0;i<brr.size();i++){
        cin>>brr[i];
    }

    vector<int> ans;
    for(int i=0;i<arr.size();i++){
        ans.push_back(arr[i]);
    }
    for(int i=0;i<brr.size();i++){
        ans.push_back(brr[i]);
    }
    //print
    for(int i=0;i<n+m;i++){
        cout<<ans[i]<<" ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter n size :
3
enter element :
2 6 4
enter m size :
2
enter element :
5 8
2 6 4 5 8
```

Q) Find common element in two array (Intersection)

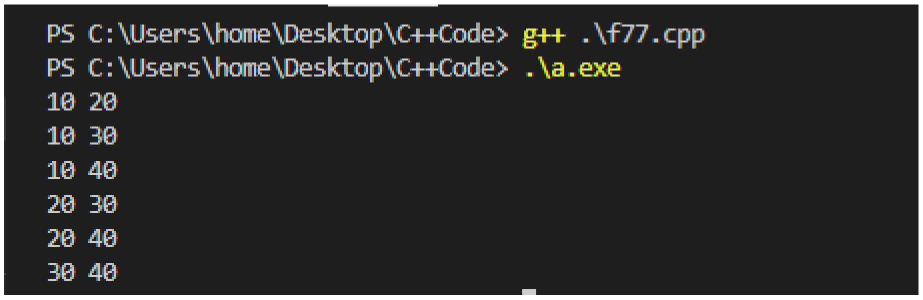
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
     vector<int> arr = {1, 2, 3,3,4,4,4};

    vector<int> brr = {1, 2, 3, 3,3,4,4};

    vector<int> ans;
    for (int i = 0; i < arr.size(); i++)
    {
        int element = arr[i];
        for (int j = 0; j < brr.size(); j++)
        {
            if (brr[j] == element)
            {
                brr[j] = INT_MIN; // mark karne ke liye
                ans.push_back(element);
                break;

            }
        }
    }
    for (auto value : ans)
    {
        cout << value << " ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 3 3 4 4
```

Q) Pair programs

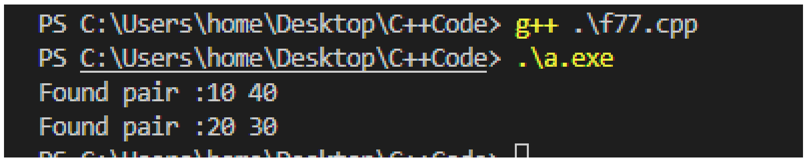
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    vector<int> arr = {10, 20, 30, 40};
    for (int i = 0; i < arr.size(); i++)
    {
        int element = arr[i];
        for (int j = i + 1; j < arr.size(); j++)
        {
            cout << element << " " << arr[j] << endl;
        }
    }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
10 20
10 30
10 40
20 30
20 40
30 40
```

Q) Find pair of sum program

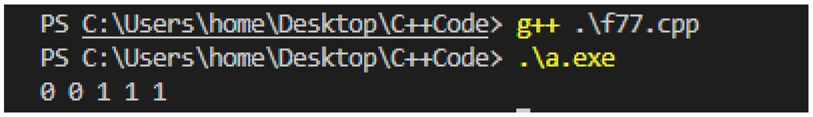
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    vector<int> arr = {10, 20, 30, 40};
    int sum=50;
    for (int i = 0; i < arr.size(); i++)
    {
        int element = arr[i];
        for (int j = i + 1; j < arr.size(); j++)
        {
            if(arr[j]+element==sum){
                cout <<"Found pair :"<< element << " " << arr[j] << endl;
            }

        }
    }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
Found pair :10 40
Found pair :20 30
PS C:\Users\home\Desktop\C++Code>
```

Q) Short 0's and 1's program

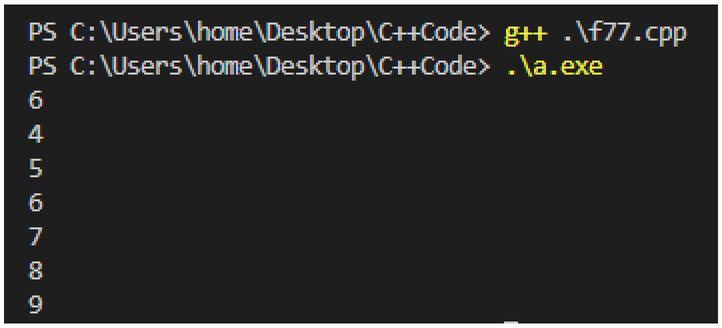
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    vector<int> arr = {1, 0, 1, 0, 1};
    int i = 0;
    int start = 0;
    int end = arr.size() - 1;
    while (i != end)
    {
        if (arr[i] == 0)
        {
            swap(arr[i], arr[start]);
            start++;
            i++;
        }
        else
        {
            swap(arr[i], arr[end]);
            end--;
        }
    }
    for (auto value : arr)
    {
        cout << value << " ";
    }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
0 0 1 1 1
```

Array –Class 3

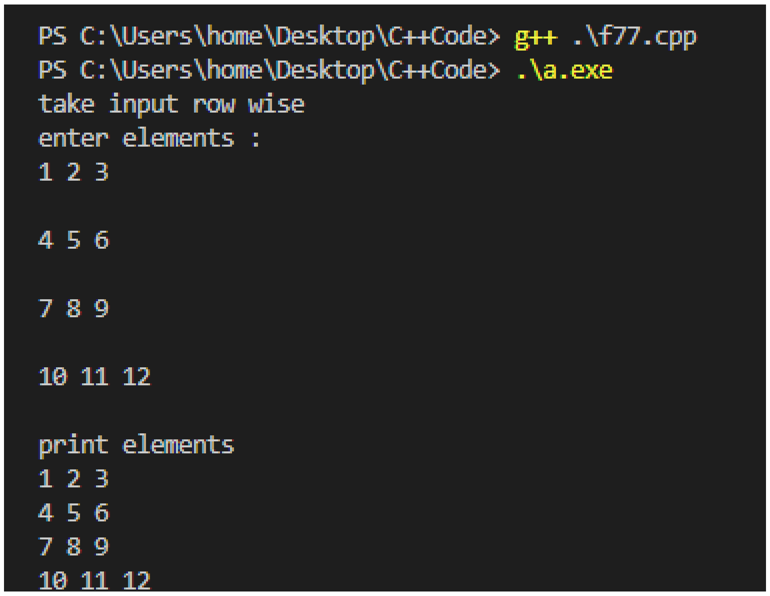
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    int arr[3][3]={
        {1,2,3},
        {4,5,6},
        {7,8,9}
    };
    cout<<arr[1][2]<<endl;
    cout<<arr[0][3]<<endl;//wrong index
    cout<<arr[0][4]<<endl;//wrong index
    cout<<arr[0][5]<<endl;//wrong index
    cout<<arr[0][6]<<endl;//wrong index
    cout<<arr[0][7]<<endl;//wrong index
    cout<<arr[0][8]<<endl;//wrong index
   return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
6
4
5
6
7
8
9
```

Q) Take input row –wise

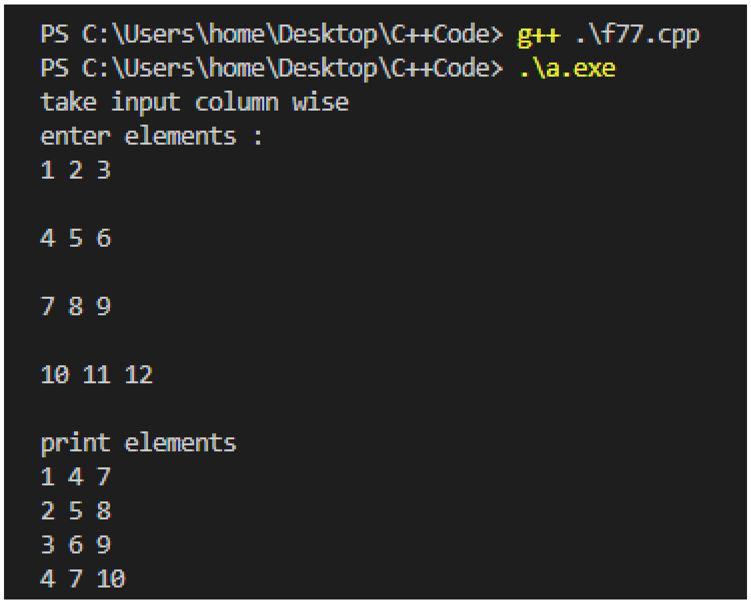
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    int arr[4][3];
    int row = 4;
    int col = 3;
    cout << "take input row wise " << endl;
    cout << "enter elements :" << endl;
    ;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin >> arr[i][j];
        }
        cout << endl;
    }
    cout << "print elements " << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
take input row wise
enter elements :
1 2 3

4 5 6

7 8 9

10 11 12

print elements
1 2 3
4 5 6
7 8 9
10 11 12
```

Q) Take input column – wise

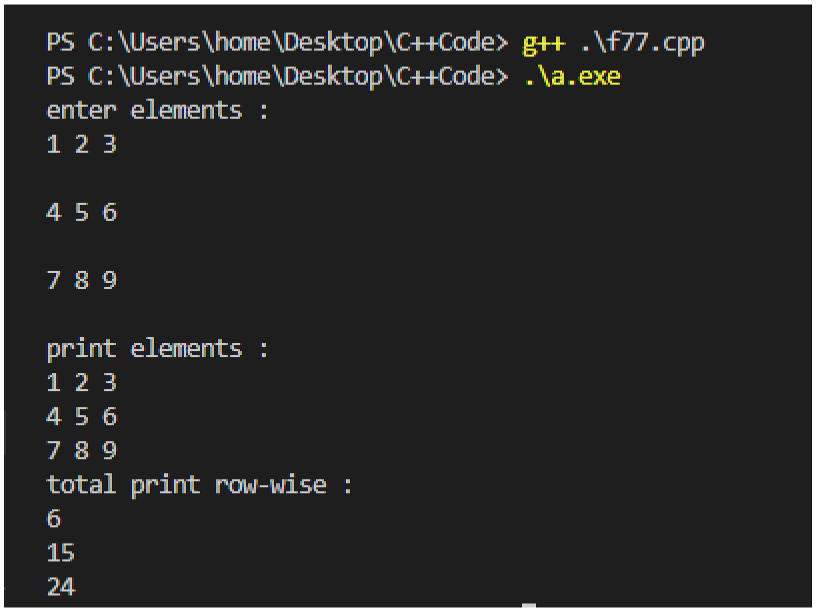
```cpp
#include <vector>
#include <iostream>
#include <limits.h>
using namespace std;

int main()
{
    int arr[4][3];
    int row = 4;
    int col = 3;
    cout << "take input column wise " << endl;
    cout << "enter elements :" << endl;
    ;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin >> arr[i][j];
        }
        cout << endl;
    }
    cout << "print elements " << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cout << arr[j][i] << " ";
        }
        cout << endl;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
take input column wise
enter elements :
1 2 3

4 5 6

7 8 9

10 11 12

print elements
1 4 7
2 5 8
3 6 9
4 7 10
```

Q) Row – wise sum print

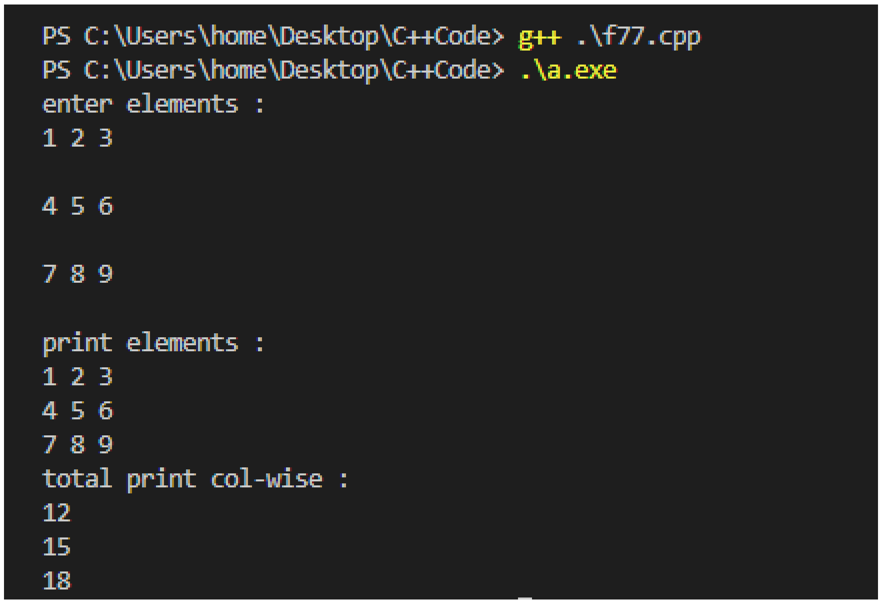
```cpp
#include <iostream>
using namespace std;

void printSumRow(int arr[][3], int row, int col)
{

    cout << "total print row-wise :" << endl;
    for (int i = 0; i < row; i++)
    {
        int sum = 0;
        for (int j = 0; j < col; j++)
        {
            sum = sum + arr[i][j];
        }
        cout << sum << endl;
    }
}
int main()
{
    int arr[3][3];
    int row = 3;
    int col = 3;
    cout << "enter elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin >> arr[i][j];
            cout << " ";
        }
        cout << endl;
    }
    cout << "print elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
    printSumRow(arr, row, col);
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter elements :
1 2 3

4 5 6

7 8 9

print elements :
1 2 3
4 5 6
7 8 9
total print row-wise :
6
15
24
```

Q) Col- wise sum print

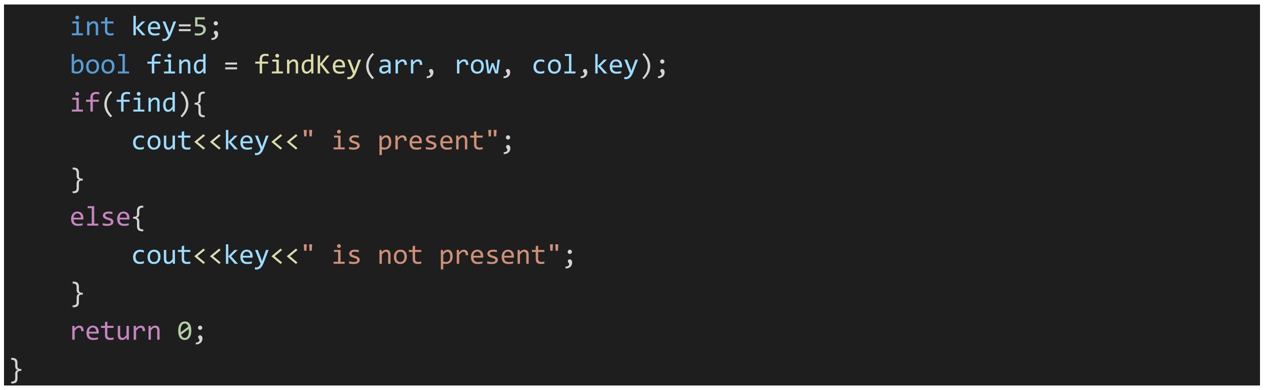
```cpp
#include <iostream>
using namespace std;

void printSumRow(int arr[][3], int row, int col)
{

    cout << "total print col-wise :" << endl;
    for (int i = 0; i < row; i++)
    {
        int sum = 0;
        for (int j = 0; j < col; j++)
        {
            sum = sum + arr[j][i];
        }
        cout << sum << endl;
    }
}
int main()
{
    int arr[3][3];
    int row = 3;
    int col = 3;
    cout << "enter elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin >> arr[i][j];
            cout << " ";
        }
        cout << endl;
    }
    cout << "print elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
    printSumRow(arr, row, col);
    return 0;
}
```



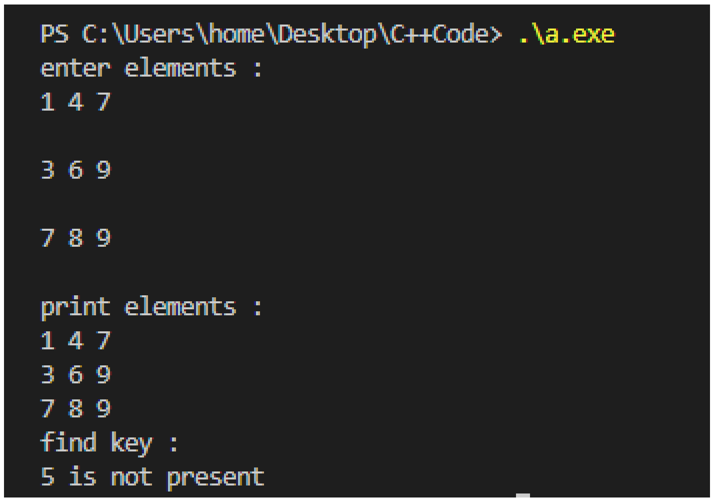
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter elements :
1 2 3

4 5 6

7 8 9

print elements :
1 2 3
4 5 6
7 8 9
total print col-wise :
12
15
18
```

Q)search element on array

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

bool findKey(int arr[][3], int row, int col,int key)//Rule-array first elements
ko chhod ka baki elements mai value dena hota hai
{

    cout << "find key :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            if(arr[i][j]==key){
                return true;
            }
        }
    }
    return false;
}
int main()
{
    int arr[3][3];
    int row = 3;
    int col = 3;
    cout << "enter elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin >> arr[i][j];
            cout << " ";
        }
        cout << endl;
    }
    cout << "print elements :" << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
```



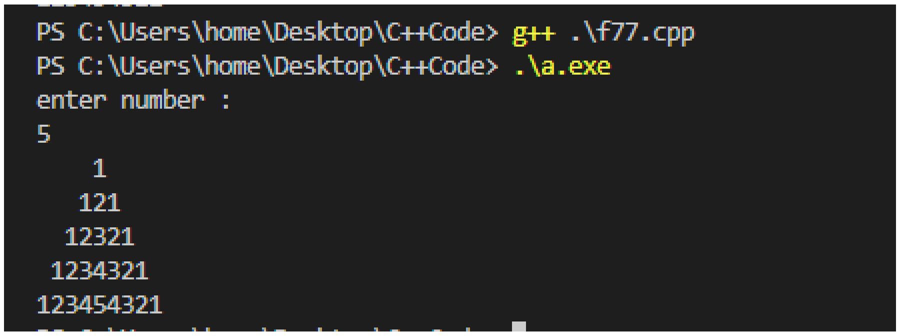
```cpp
    int key=5;
    bool find = findKey(arr, row, col,key);
    if(find){
        cout<<key<<" is present";
    }
    else{
        cout<<key<<" is not present";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter elements :
1 4 7

3 6 9

7 8 9

print elements :
1 4 7
3 6 9
7 8 9
find key :
5 is not present
```

Q) Print full pyramid

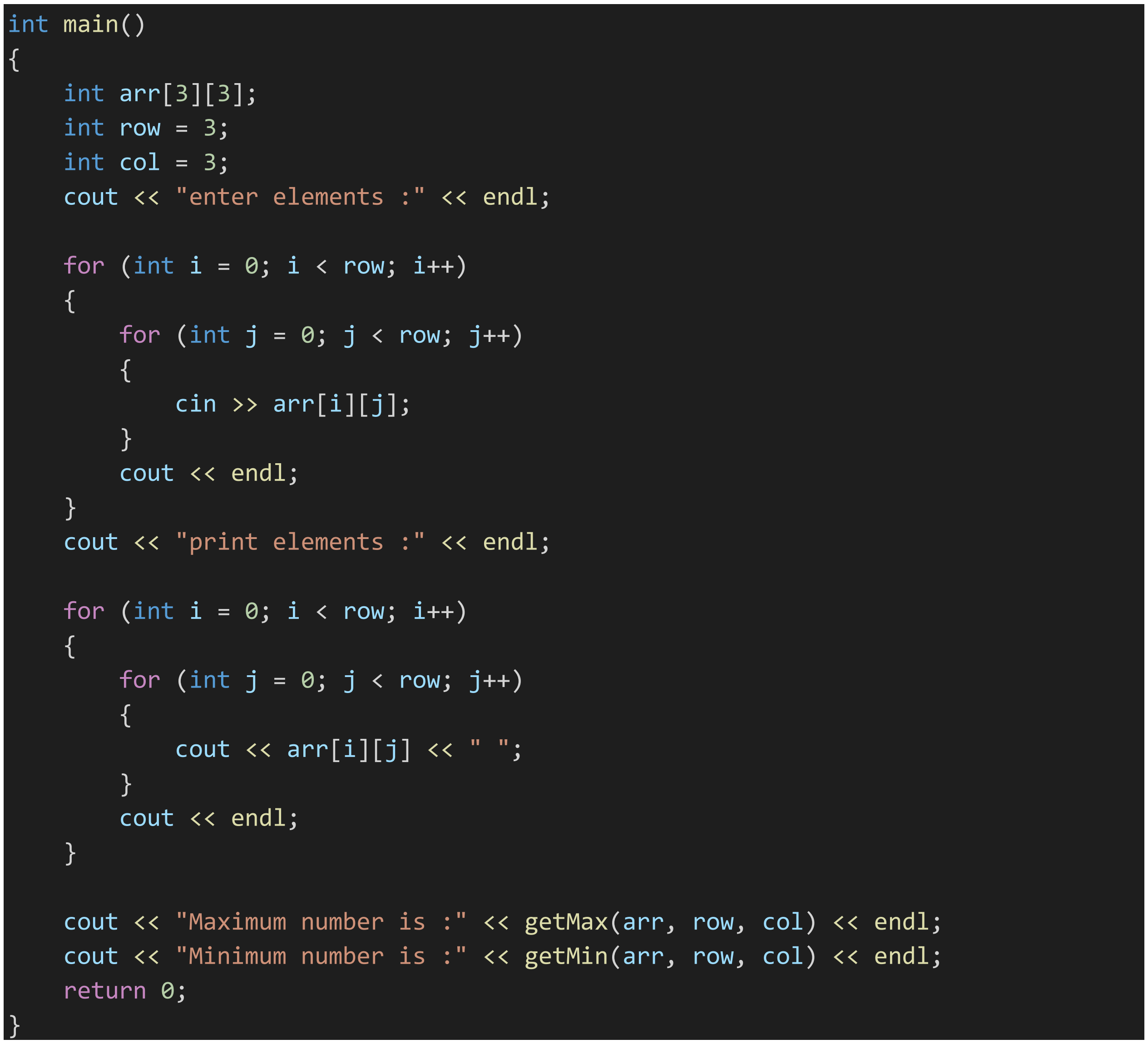
```cpp
#include <iostream>
using namespace std;

int main()
{
    int n;
    cout << "enter number :" << endl;
    cin >> n;

    for (int row = 1; row <= n; row++)
    {
        for (int col = 1; col <= n - row; col++)
        {
            cout << " ";
        }
        int i = 1;
        for (i = 1; i <= row; i++)
        {
            cout << i;
        }
        int start = row - 1;
        while (start != 0)
        {
            cout << start;
            start = start - 1;
        }

        cout << endl;
    }
    return 0;
}
```



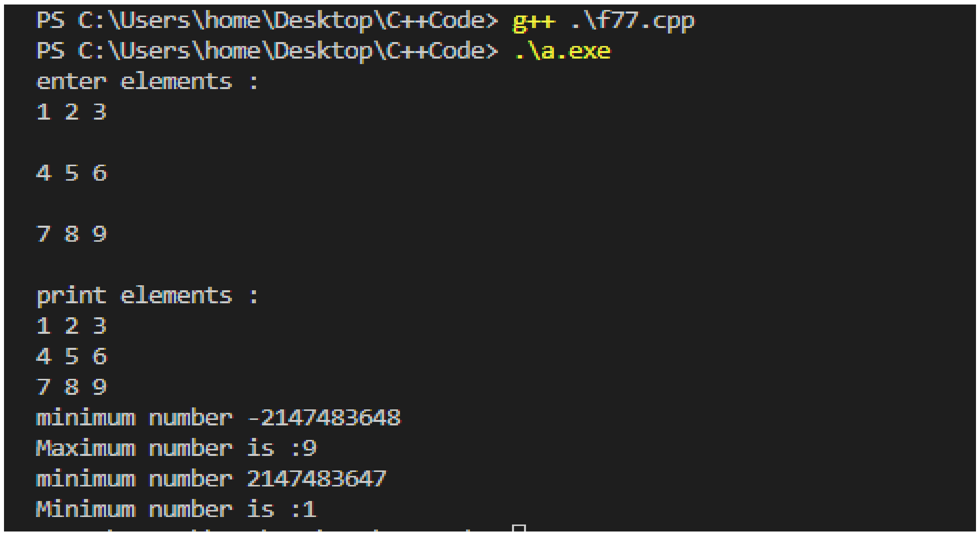
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :
5
    1
   121
  12321
 1234321
123454321
```

Q)Find Maximum and Minimum number in 2D array

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

int getMax(int arr[][3], int row, int col)
{
    int maxi = INT_MIN;
    cout << "minimum number " << INT_MIN << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            if (arr[i][j] > maxi)
            {
                maxi = arr[i][j];
            }
        }
    }

    return maxi;
}
int getMin(int arr[][3], int row, int col)
{
    int mini = INT_MAX;
    cout << "minimum number " << INT_MAX << endl;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            if (arr[i][j] < mini)
            {
                mini = arr[i][j];
            }
        }
    }

    return mini;
}
```



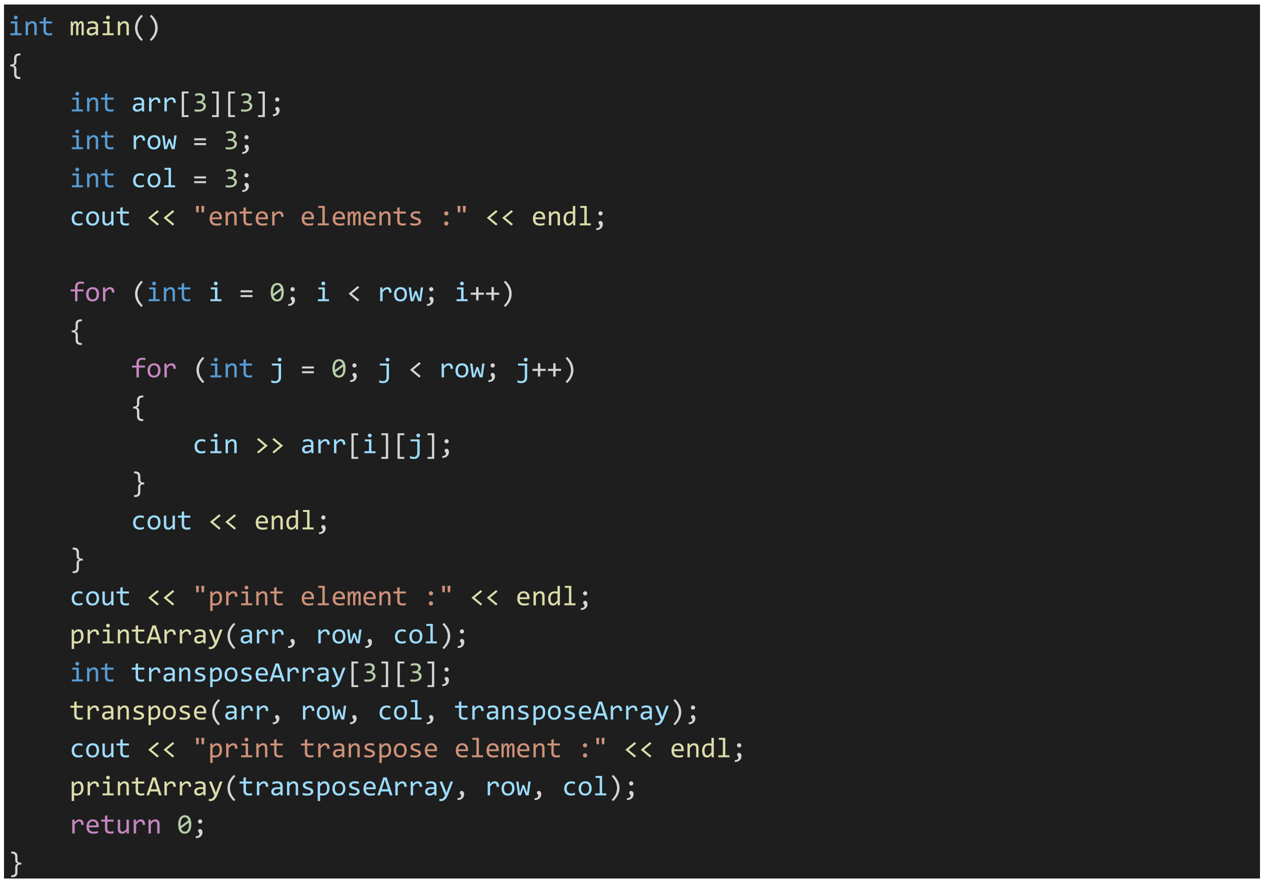
```
int main()
{
    int arr[3][3];
    int row = 3;
    int col = 3;
    cout << "enter elements :" << endl;

    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < row; j++)
        {
            cin >> arr[i][j];
        }
        cout << endl;
    }
    cout << "print elements :" << endl;

    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < row; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }

    cout << "Maximum number is :" << getMax(arr, row, col) << endl;
    cout << "Minimum number is :" << getMin(arr, row, col) << endl;
    return 0;
}
```



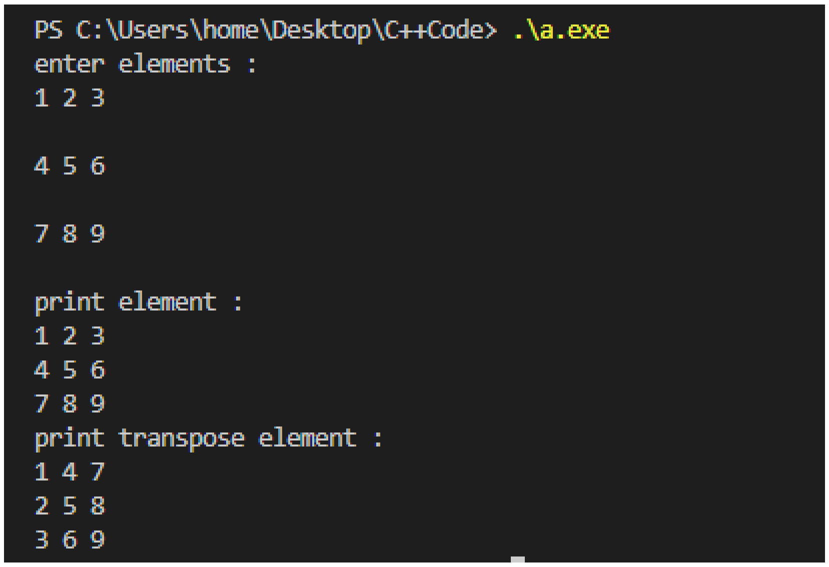
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter elements :
1 2 3

4 5 6

7 8 9

print elements :
1 2 3
4 5 6
7 8 9
minimum number -2147483648
Maximum number is :9
minimum number 2147483647
Minimum number is :1
```

Q) Transpose element in 2D array

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

void transpose(int arr[][3], int row, int col, int transposeArray[3][3])
{
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            transposeArray[j][i] = arr[i][j];
        }
    }
}

void printArray(int arr[][3], int row, int col)
{
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < row; j++)
        {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}
```



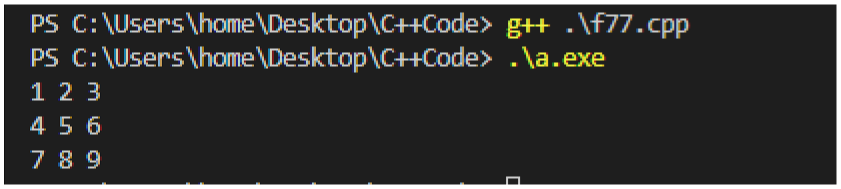
```cpp
int main()
{
    int arr[3][3];
    int row = 3;
    int col = 3;
    cout << "enter elements :" << endl;

    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < row; j++)
        {
            cin >> arr[i][j];
        }
        cout << endl;
    }
    cout << "print element :" << endl;
    printArray(arr, row, col);
    int transposeArray[3][3];
    transpose(arr, row, col, transposeArray);
    cout << "print transpose element :" << endl;
    printArray(transposeArray, row, col);
    return 0;
}
```



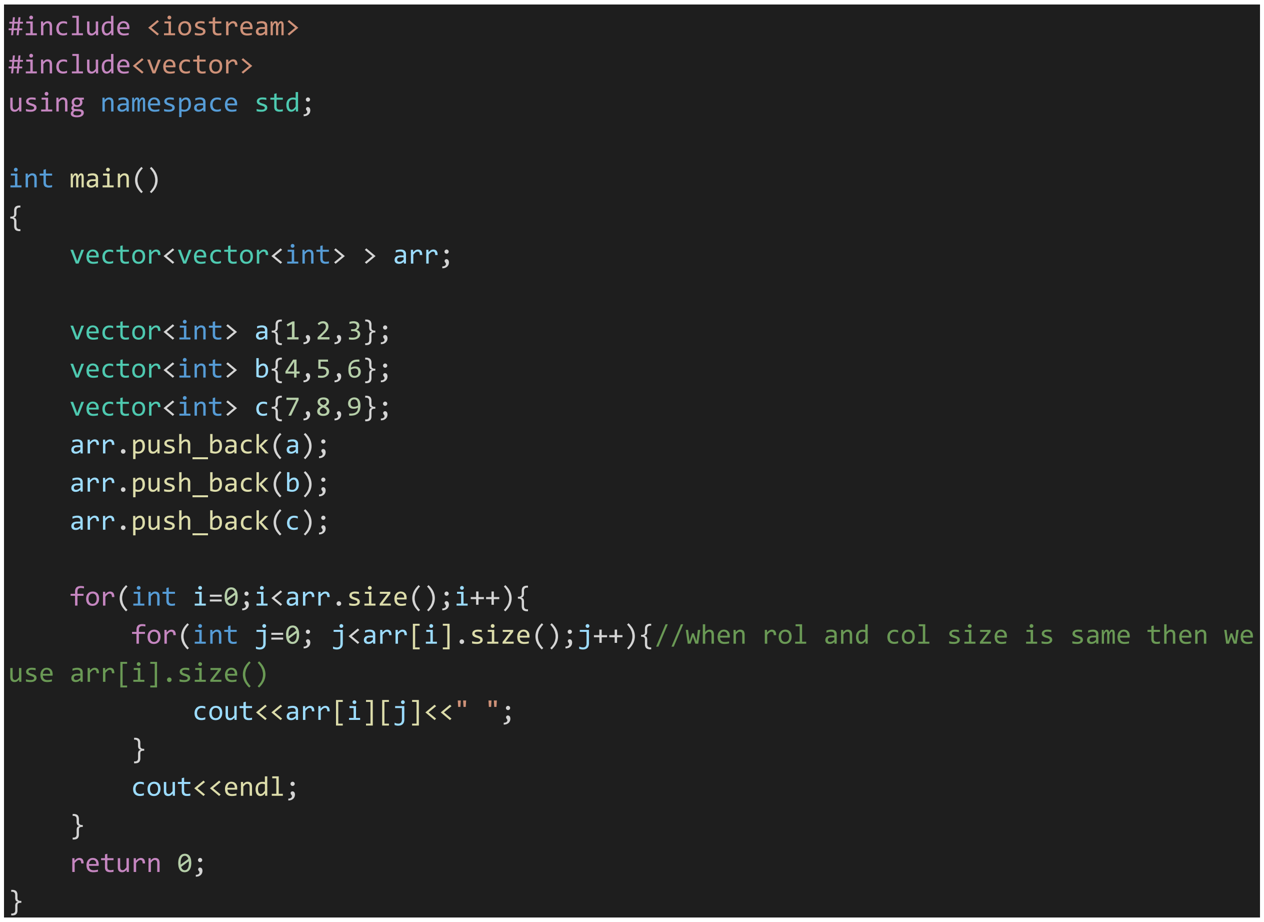
```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter elements :
1 2 3

4 5 6

7 8 9

print element :
1 2 3
4 5 6
7 8 9
print transpose element :
1 4 7
2 5 8
3 6 9
```

Q) Vector of vector

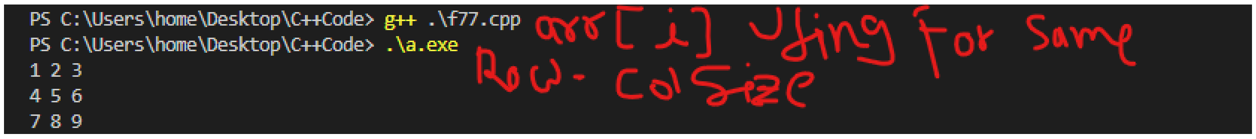
```cpp
#include <iostream>
#include<vector>
using namespace std;

int main()
{
    vector<vector<int> > arr;

    vector<int> a{1,2,3};
    vector<int> b{4,5,6};
    vector<int> c{7,8,9};
    arr.push_back(a);
    arr.push_back(b);
    arr.push_back(c);

    for(int i=0;i<arr.size();i++){
        for(int j=0; j<arr[0].size();j++){//when rol and col size is same then we use arr[0].size()
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 3
4 5 6
7 8 9
```



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

int main()
{
    vector<vector<int> > arr;

    vector<int> a{1,2,3};
    vector<int> b{4,5,6};
    vector<int> c{7,8,9};
    arr.push_back(a);
    arr.push_back(b);
    arr.push_back(c);

    for(int i=0;i<arr.size();i++){
        for(int j=0; j<arr[i].size();j++){//when rol and col size is same then we
use arr[i].size()
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

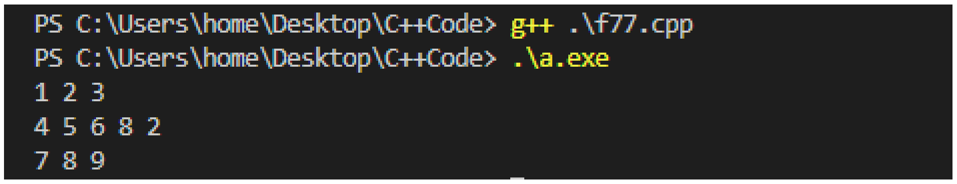


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 3
4 5 6
7 8 9
```

arr[i] Using For Same Row- Col Size

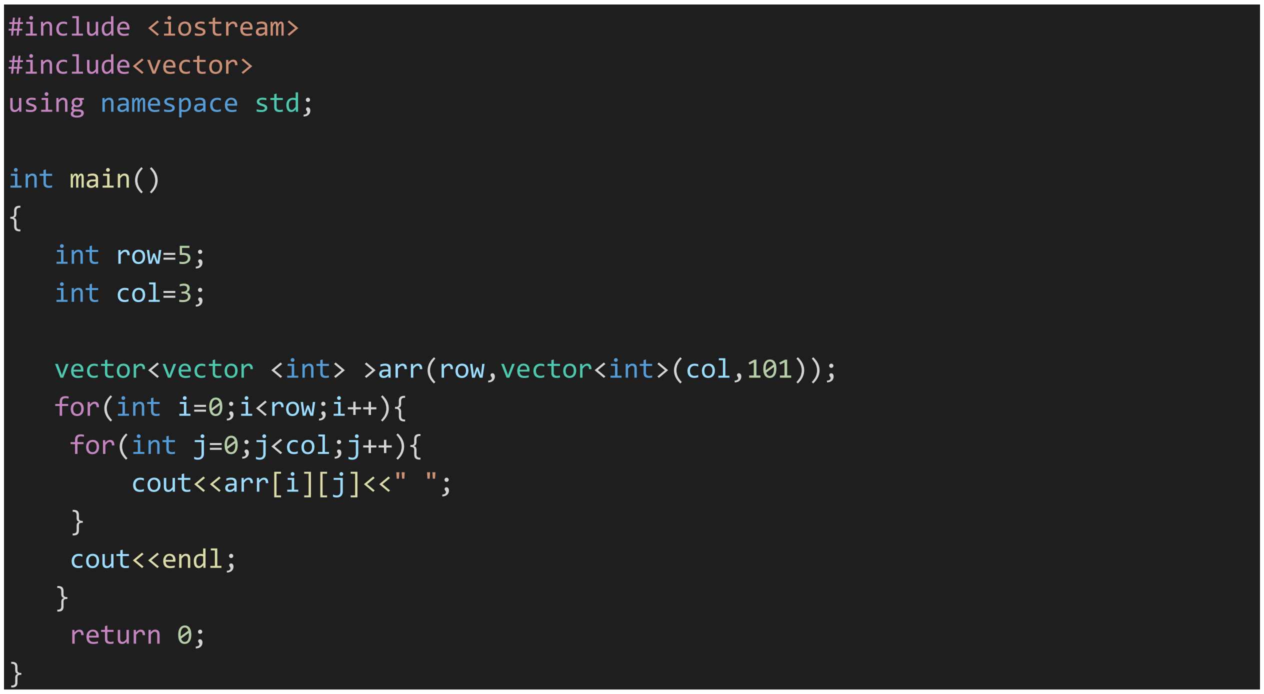
Q) When column and row are different

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

int main()
{
    vector<vector<int> > arr;

    vector<int> a{1,2,3};
    vector<int> b{4,5,6,8,2};
    vector<int> c{7,8,9};
    arr.push_back(a);
    arr.push_back(b);
    arr.push_back(c);

    for(int i=0;i<arr.size();i++){
        for(int j=0; j<arr[i].size();j++){//when rol and col size is different
then we use arr[i].size()
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

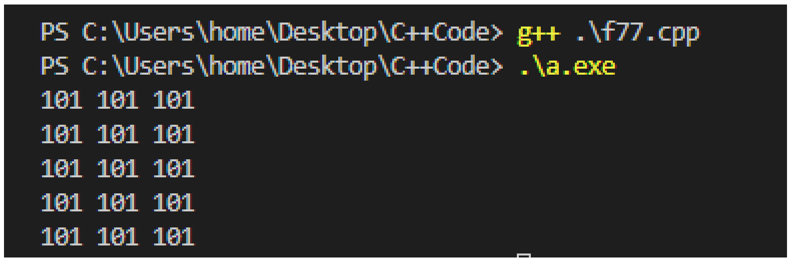


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 3
4 5 6 8 2
7 8 9
```

Q) vector of vector program



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

int main()
{
   int row=5;
   int col=3;

   vector<vector <int> >arr(row,vector<int>(col,101));
   for(int i=0;i<row;i++){
    for(int j=0;j<col;j++){
        cout<<arr[i][j]<<" ";
    }
    cout<<endl;
   }
    return 0;
}
```



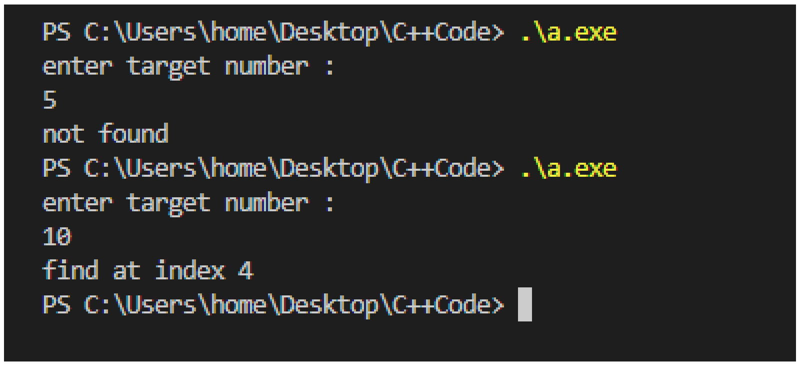
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
101 101 101
101 101 101
101 101 101
101 101 101
101 101 101
```

Week 04 – searching & sorting

Searching and sorting class – I

Q) search number in array using binary

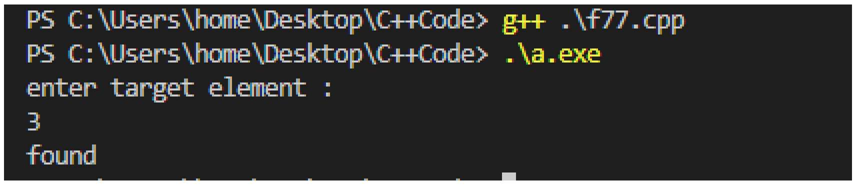
```
#include <iostream>
using namespace std;

int binarySearch(int arr[], int size, int target)
{
    int s = 0;
    int e = size - 1;
    int mid = s + (e - s) / 2;
    int element = target;
    while (s <= e)
    {
        if (arr[mid] == target)
        {
            return mid;
        }
        if (arr[mid] > target)
        {
            e = mid - 1;
        }
        else
        {
            s = mid + 1;
        }
        mid = s + (e - s) / 2;
    }
    return -1;
}
int main()
{
    int arr[] = {2, 4, 6, 8, 10, 12, 14};
    int target;
    cout << "enter target number :" << endl;
    cin >> target;
    int size = 7;
    int ans = binarySearch(arr, size, target);
    if (ans >= 0)
    {
        cout << "find at index " << ans;
    }
else
    {
        cout << "not found ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target number :
5
not found
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target number :
10
find at index 4
PS C:\Users\home\Desktop\C++Code>
```

Q) using pre-defined function for search number in vector

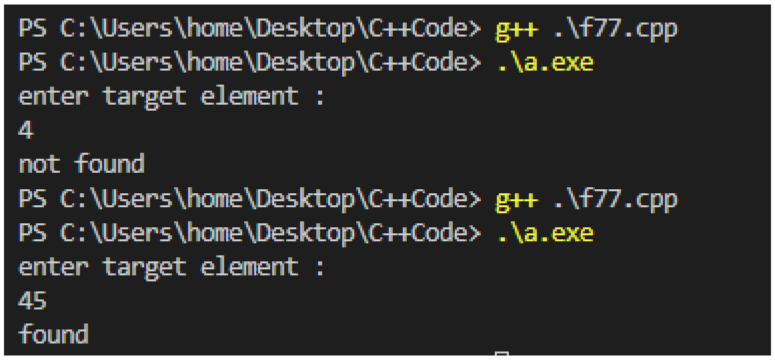
```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main()
{
    vector<int> v{1, 2, 3, 8, 9, 11, 15, 17};
    int target;
    cout << "enter target element :" << endl;
    cin >> target;
    if (binary_search(v.begin(), v.end(), target))
    {
        cout << "found ";
    }
    else
    {
        cout << "not found ";
    }
    return 0;
    // binary_search() is pre-define function ,with help of binary_search() we
can find number is present or not
    // we using  #include<algorithm>  for binary_search()
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target element :
3
found
```

Q) using pre-defined function for search number in array

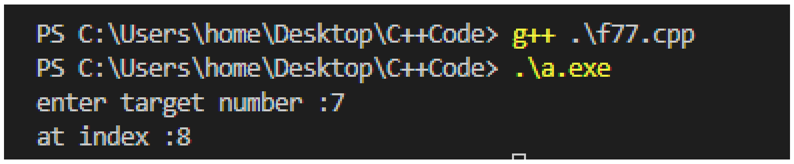
```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main()
{
    int arr[]={1,2,3,8,45,51};
    int size=6;
    int target;
    cout << "enter target element :" << endl;
    cin >> target;
    if (binary_search(arr, arr+size, target))
    {
        cout << "found ";
    }
    else
    {
        cout << "not found ";
    }
    return 0;
    // binary_search() is pre-define function ,with help of binary_search() we
can find number is present or not

}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target element :
4
not found
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target element :
45
found
```

Q) find out first occurrence

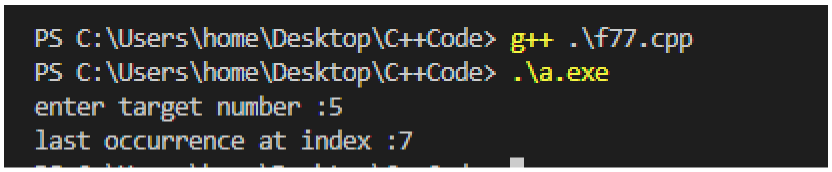
```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int binarySearch(vector<int> v, int target)
{
    int s = 0;
    int e = v.size() - 1;
    int mid = s + (e - s) / 2;
    int element = target;
    int ans = -1;
    while (s <= e)
    {
        if (v[mid] == element)
        {
            ans = mid;
            e = mid - 1;
        }
        else if (v[mid] < element)
        {
            s = mid + 1;
        }
        else if (v[mid] > element)
        {
            e = mid - 1;
        }
        mid = s + (e - s) / 2;
    }
    return ans;
}

int main()
{
    vector<int> v{1, 2, 3, 4, 5, 5, 5, 5, 7, 7, 7, 7, 8, 9};
    int target;
    cout << "enter target number :";
    cin >> target;

    int ans = binarySearch(v, target);
    cout << "at index :" << ans;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target number :7
at index :8
```

Q) find out first occurrence

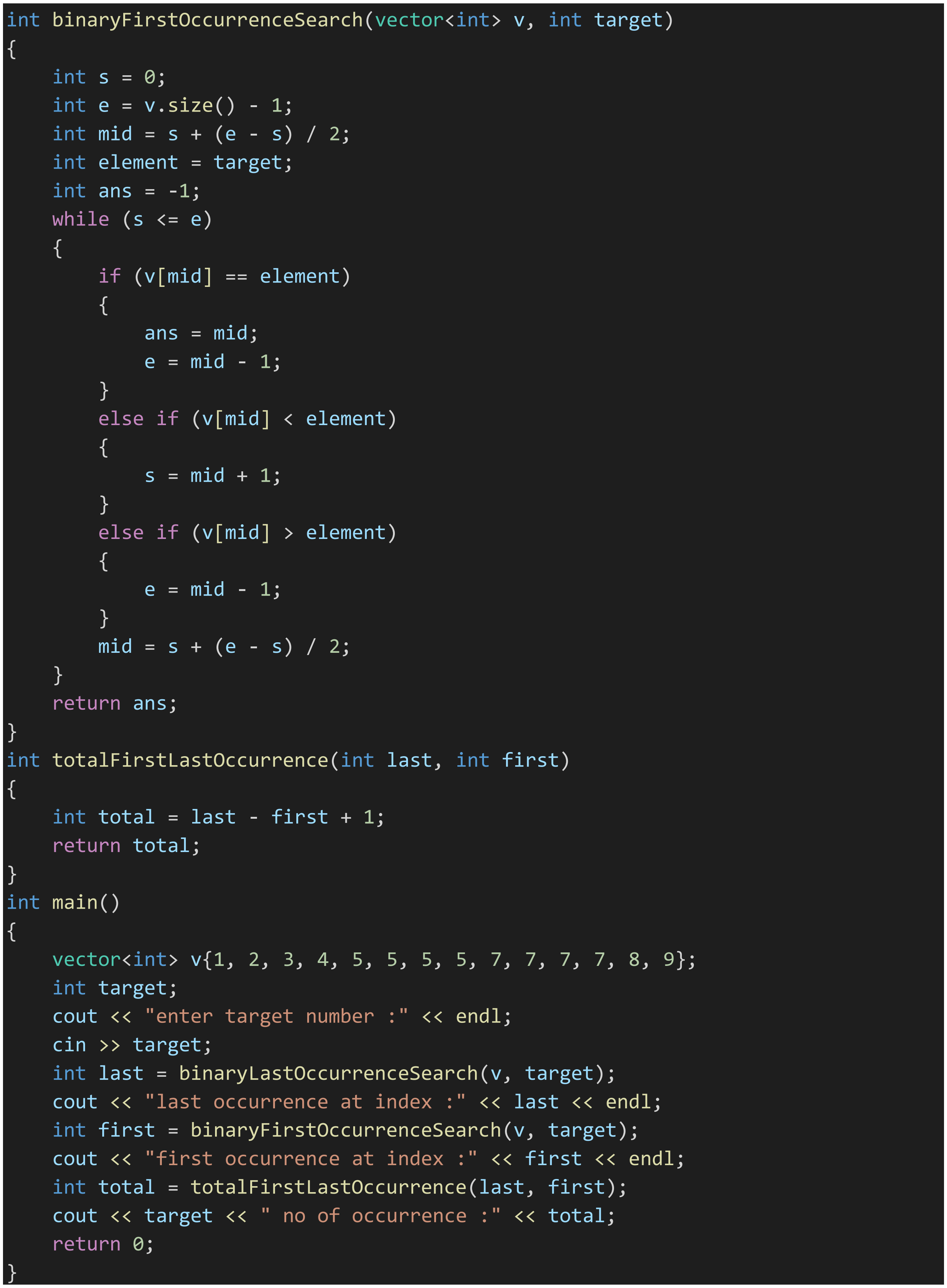
```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int binarySearch(vector<int> v, int target)
{
    int s = 0;
    int e = v.size() - 1;
    int mid = s + (e - s) / 2;
    int element = target;
    int ans = -1;
    while (s <= e)
    {
        if (v[mid] == element)
        {
            ans = mid;
            s=mid+1;
        }
        else if (v[mid] < element)
        {
            s = mid + 1;
        }
        else if (v[mid] > element)
        {
            e = mid - 1;
        }
        mid = s + (e - s) / 2;
    }
    return ans;
}

int main()
{
    vector<int> v{1, 2, 3, 4, 5, 5, 5, 5, 7, 7, 7, 7, 8, 9};
    int target;
    cout << "enter target number :";
    cin >> target;

    int ans = binarySearch(v, target);
    cout << "last occurrence at index :" << ans;
}
```



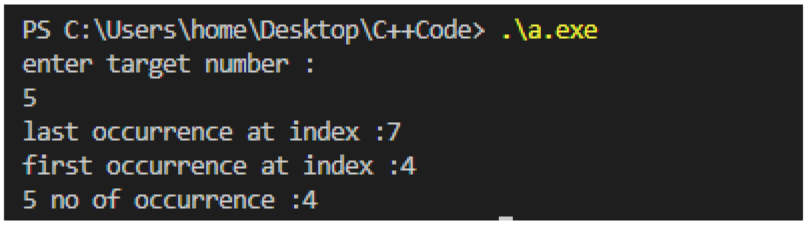
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target number :5
last occurrence at index :7
```

Q) find total number of occurrence

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int binaryLastOccurrenceSearch(vector<int> v, int target)
{
    int s = 0;
    int e = v.size() - 1;
    int mid = s + (e - s) / 2;
    int element = target;
    int ans = -1;
    while (s <= e)
    {
        if (v[mid] == element)
        {
            ans = mid;
            s = mid + 1;
        }
        else if (v[mid] < element)
        {
            s = mid + 1;
        }
        else if (v[mid] > element)
        {
            e = mid - 1;
        }
        mid = s + (e - s) / 2;
    }
    return ans;
}
```



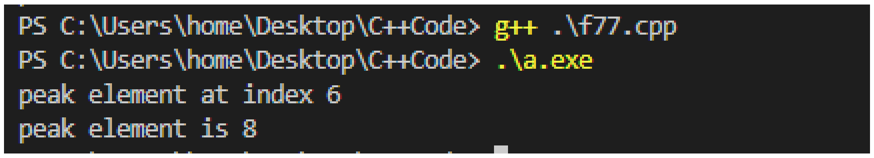
```
int binaryFirstOccurrenceSearch(vector<int> v, int target)
{
    int s = 0;
    int e = v.size() - 1;
    int mid = s + (e - s) / 2;
    int element = target;
    int ans = -1;
    while (s <= e)
    {
        if (v[mid] == element)
        {
            ans = mid;
            e = mid - 1;
        }
        else if (v[mid] < element)
        {
            s = mid + 1;
        }
        else if (v[mid] > element)
        {
            e = mid - 1;
        }
        mid = s + (e - s) / 2;
    }
    return ans;
}
int totalFirstLastOccurrence(int last, int first)
{
    int total = last - first + 1;
    return total;
}
int main()
{
    vector<int> v{1, 2, 3, 4, 5, 5, 5, 5, 7, 7, 7, 7, 8, 9};
    int target;
    cout << "enter target number :" << endl;
    cin >> target;
    int last = binaryLastOccurrenceSearch(v, target);
    cout << "last occurrence at index :" << last << endl;
    int first = binaryFirstOccurrenceSearch(v, target);
    cout << "first occurrence at index :" << first << endl;
    int total = totalFirstLastOccurrence(last, first);
    cout << target << " no of occurrence :" << total;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter target number :
5
last occurrence at index :7
first occurrence at index :4
5 no of occurrence :4
```

Q) Find peak element (leetcode-852. Peak Index in a Mountain Array)

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int findPeak(vector<int>arr){
    int s=0;
    int e=arr.size()-1;
    int mid=s+(e-s)/2;
    while(s<e){
        if(arr[mid]<arr[mid+1]){
            cout<<"arr[mid] = "<<arr[mid]<<endl;
            cout<<"arr[mid+1 ] = "<<arr[mid+1]<<endl;
            s=mid+1;
            cout<<"s ="<<s<<endl;
        }
        else{
            e=mid;
            cout<<"e ="<<e<<endl;
        }
        mid=s+(e-s)/2;
        cout<<"mid ="<<mid<<endl;
    }
    return s;
}

int main()
{
    vector<int> arr{1,2,3,4,5,6,8,4};
    int ans=findPeak(arr);
    cout<<"peak element at index "<<ans<<endl;
    cout<<"peak element is "<<arr[ans];
    return 0;
}
```



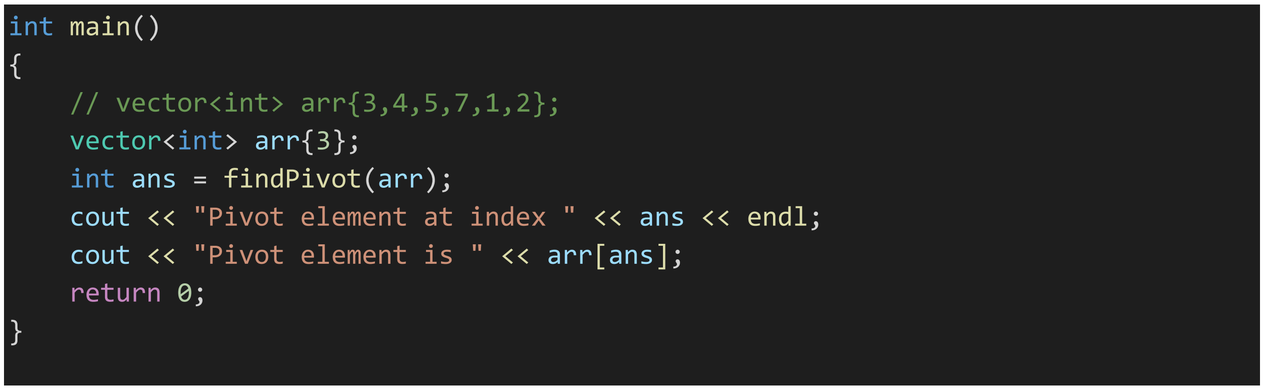
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
peak element at index 6
peak element is 8
```

Week 04 – searching & sorting

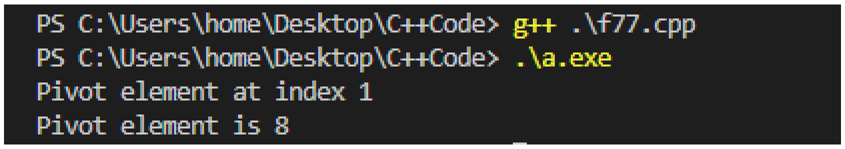
Searching and sorting class – II

Q) Find pivot number in array

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

int findPivot(vector<int> arr)
{
    int s = 0;
    int e = arr.size() - 1;
    int mid = s + (e - s) / 2;
    while (s < e)
    {
        if (mid + 1 < arr.size() && arr[mid] > arr[mid + 1])
        {
            return mid;
        }
        if (mid - 1 >= 0 && arr[mid] < arr[mid - 1])
        {
            return mid - 1;
        }
        if (arr[s] >= arr[mid])
        {
            e = mid - 1;
        }
        else
        {
            s = mid;
        }
        mid = s + (e - s) / 2;
    }
    return s;
}
```



```cpp
int main()
{
    // vector<int> arr{3,4,5,7,1,2};
    vector<int> arr{3};
    int ans = findPivot(arr);
    cout << "Pivot element at index " << ans << endl;
    cout << "Pivot element is " << arr[ans];
    return 0;
}
```

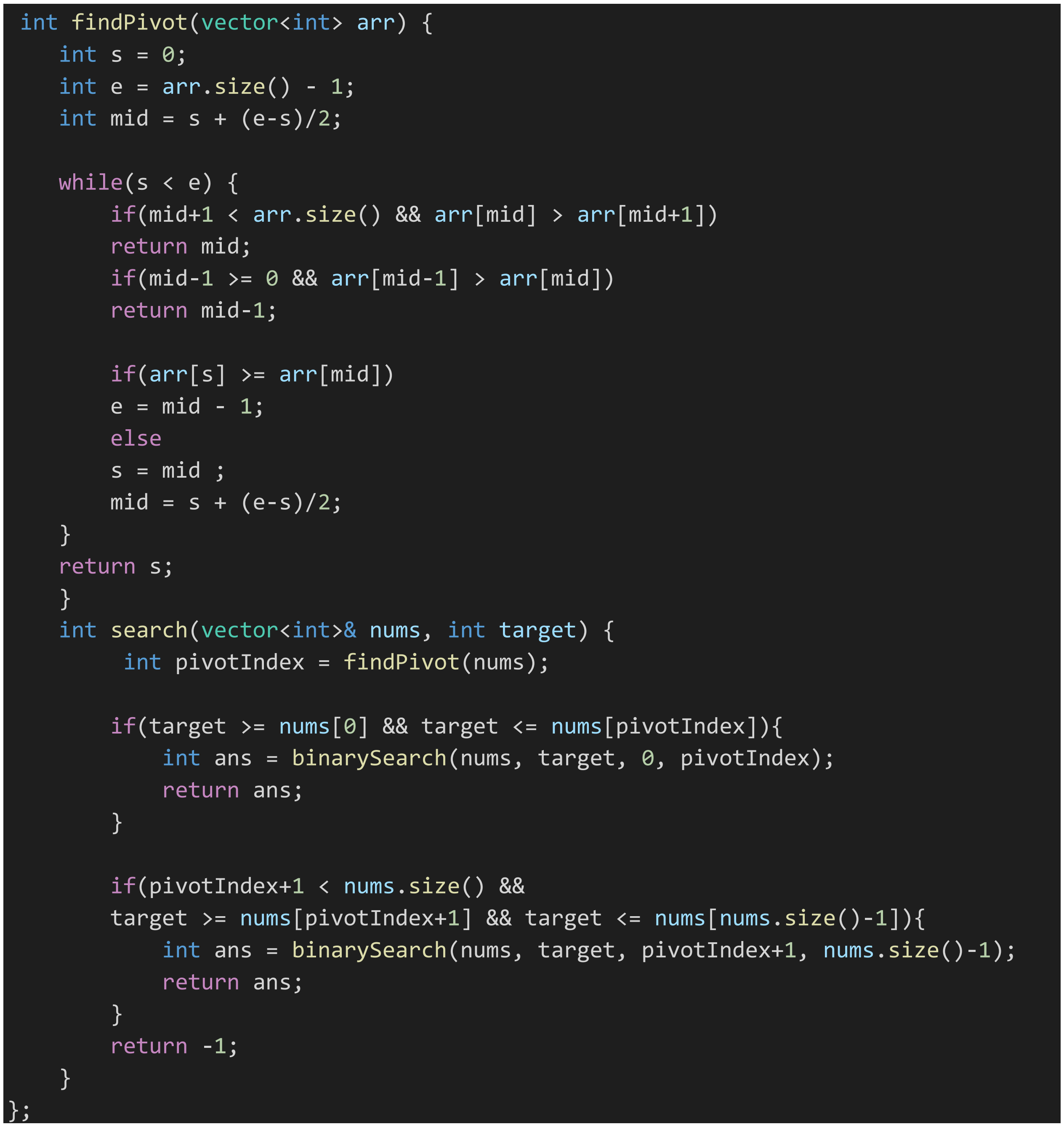


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
Pivot element at index 1
Pivot element is 8
```

Q) search in rotaed array (leetcode-33. Search in Rotated Sorted Array)

Leetcode-

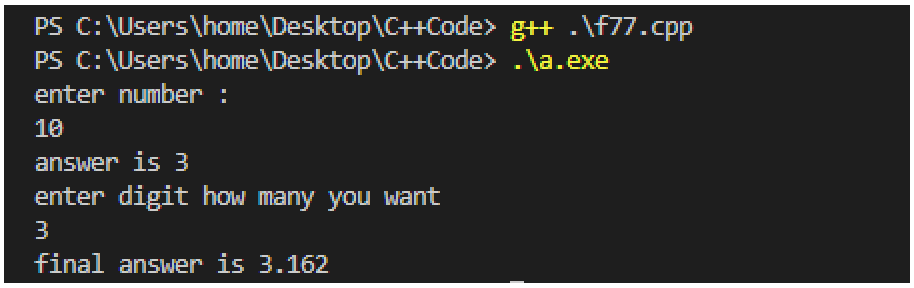
```
class Solution {
public:
int binarySearch(vector<int> arr, int target, int start, int end) {

    int mid = start + (end - start ) / 2;

    while(start <= end) {
        int element = arr[mid];

        if(element == target) {
        return mid;
        }

        if(target < element) {
        end = mid - 1;
        }
        else {
        start = mid + 1;
        }

        mid = start + (end - start ) / 2;

    }

  return -1;

}
```



```
 int findPivot(vector<int> arr) {
    int s = 0;
    int e = arr.size() - 1;
    int mid = s + (e-s)/2;

    while(s < e) {
        if(mid+1 < arr.size() && arr[mid] > arr[mid+1])
        return mid;
        if(mid-1 >= 0 && arr[mid-1] > arr[mid])
        return mid-1;

        if(arr[s] >= arr[mid])
        e = mid - 1;
        else
        s = mid ;
        mid = s + (e-s)/2;
    }
    return s;
    }
    int search(vector<int>& nums, int target) {
         int pivotIndex = findPivot(nums);

        if(target >= nums[0] && target <= nums[pivotIndex]){
            int ans = binarySearch(nums, target, 0, pivotIndex);
            return ans;
        }

        if(pivotIndex+1 < nums.size() &&
        target >= nums[pivotIndex+1] && target <= nums[nums.size()-1]){
            int ans = binarySearch(nums, target, pivotIndex+1, nums.size()-1);
            return ans;
        }
        return -1;
    }
};
```

Q) Square root of a number using binary search

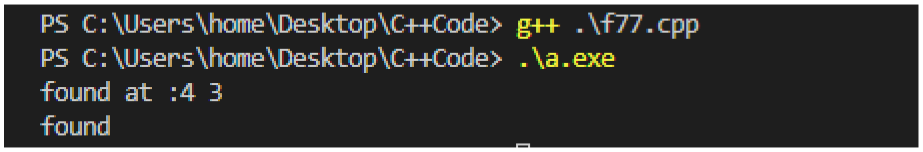
```cpp
#include <iostream>
using namespace std;

int square(int n){
    int s=0;
    int e=n-1;
    int mid=s+(e-s)/2;
    int ans;
    while(s<=e){
        if(mid==n){
            return mid;
        }
        if(mid*mid>n){
            e=mid-1;
        }
        else{
            ans=mid;
            s=mid+1;
        }
        mid=s+(e-s)/2;
    }
    return ans;
}
int main()
{
    int n;
    cout<<"enter number :"<<endl;;
    cin>>n;
    int ans=square(n);
    cout<<"answer is "<<ans<<endl;;
    double finalAns=ans;
    int precision;
    cout<<"enter digit how many you want "<<endl;
    cin>>precision;
    double step=0.1;
    for(int i=0;i<precision;i++){
        for(double j=finalAns;j*j<=n;j+=step){
            finalAns=j;
        }
        step=step/10;
    }
    cout<<"final answer is "<<finalAns<<endl;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter number :
10
answer is 3
enter digit how many you want
3
final answer is 3.162
```

Q) Binary Search in 2D array

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

bool binarySearch(int arr[][4],int row,int col,int target){
    int s=0;
    int e=row*col-1;
    int mid=s+(e-s)/2;
    while(s<=e){
        int rowIndex=mid/col;
        int colIndex=mid%col;
        if(arr[rowIndex][colIndex]==target){
            cout<<"found at :"<<rowIndex<<" "<<colIndex<<endl;
            return true;
        }
        if(arr[rowIndex][colIndex] < target){
           s=mid+1;
        }
        else{
            e=mid-1;
        }
        mid=s+(e-s)/2;
    }
    return false;
}

int main()
{
   int arr [5][4]={ {1,2,3,4},{5,6,7,8},
   {9,10,11,12},{13,14,15,16},{17,18,19,20} };
   int row=5;
   int col=4;
   int target =20;
   bool ans=binarySearch(arr,row,col,target);
   if(ans){
    cout<<"found ";
   }
   else{
    cout<<" not found ";
   }
    return 0;
}
```



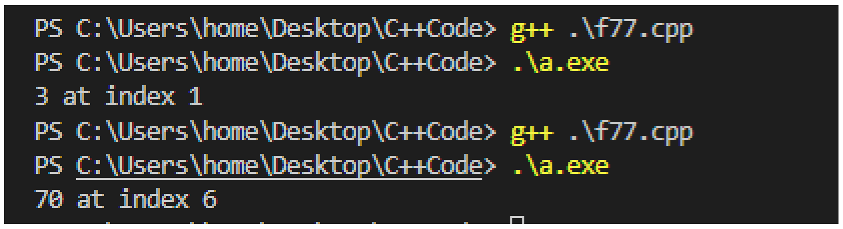
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
found at :4 3
found
```

Week 04 – searching & sorting

Searching and sorting class – III

Q) Find element in nearly sorted array

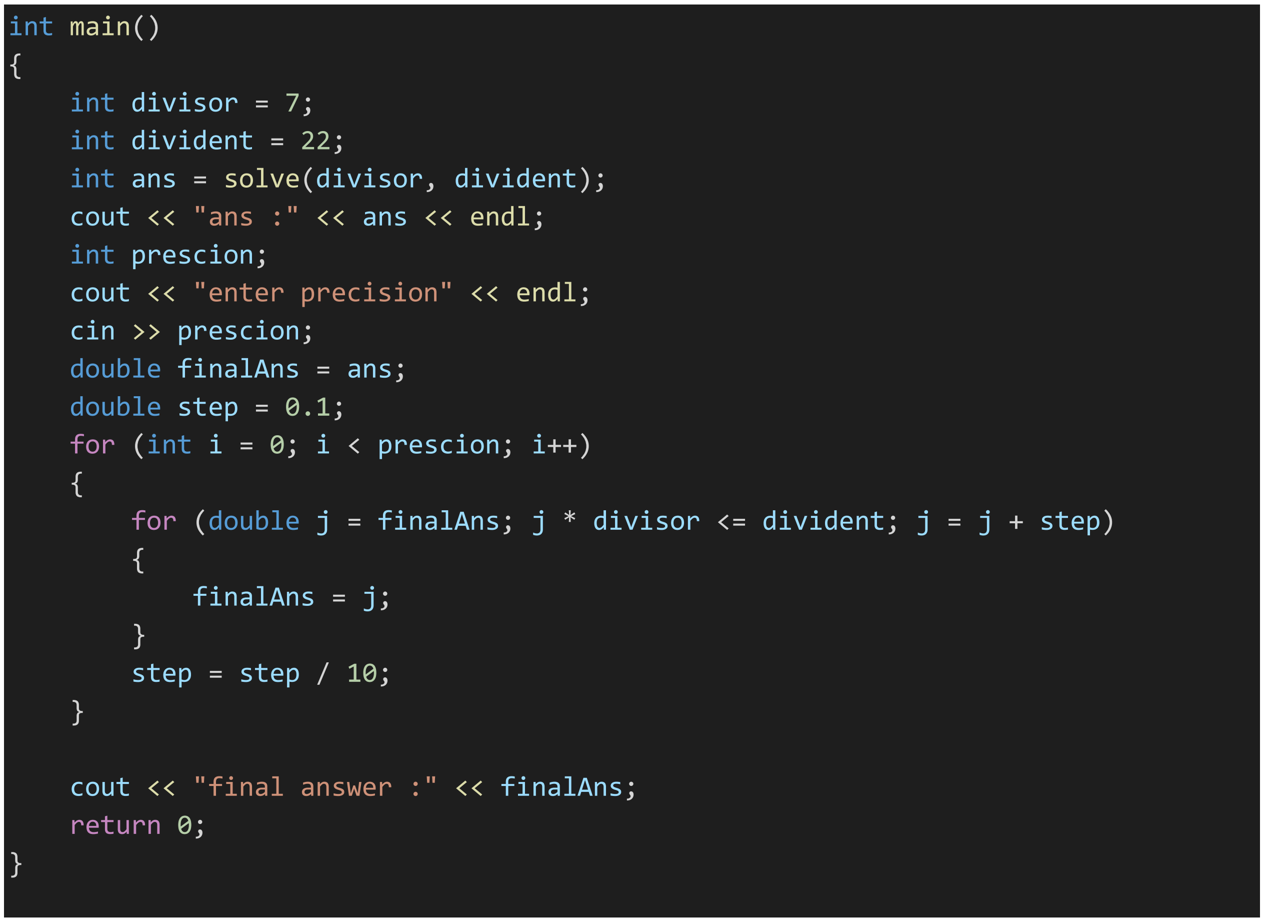
```
#include <iostream>
#include <vector>

using namespace std;

int binarySearch(vector<int>arr,int target){
    int s=0;
    int e=arr.size()-1;
    int mid=s+(e-s)/2;
    while(s<=e){
        if(arr[mid]==target){
            return mid;
        }
         if(arr[mid+1]==target){
            return mid+1;
        }
        if(arr[mid-1]==target){
            return mid-1;
        }
        if(arr[mid] < target ){
            s=mid+2;
        }
        else{
            e=mid-2;
        }
        mid=s+(e-s)/2;
    }
    return -1;
}
int main()
{
    vector<int>arr{10,3,40,20,50,80,70};
    int target=70;
    int ans=binarySearch(arr,target);
    cout<<target<<" at index "<<ans;
    return 0;
}
```



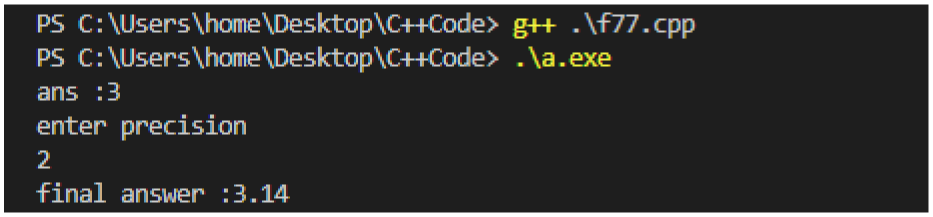
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
3 at index 1
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
70 at index 6
```

Q) Divide two numbers using binary search

```cpp
#include <iostream>
#include <vector>

using namespace std;

int solve(int divisor, int divident)
{
    int s = 0;
    int e = abs(divident);
    int mid = s + (e - s) / 2;
    int ans = 0;
    while (s <= e)
    {
        if (abs(divident) == abs(mid * divisor))
        {
            ans = mid;
            break;
        }
        if (abs(mid * divisor) > abs(divident))
        {
            e = mid - 1;
        }
        else
        {
            ans = mid;
            s = mid + 1;
        }
        mid = s + (e - s) / 2;
    }
    if ((divisor < 0 && divident < 0) || (divisor > 0 && divident > 0))
    {
        return ans;
    }
    else
    {
        return -ans;
    }
}
```



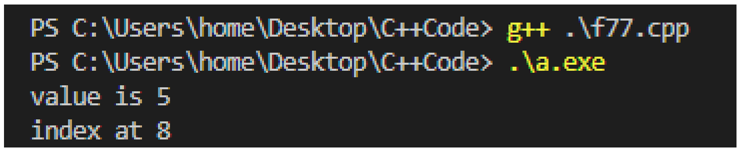
```cpp
int main()
{
    int divisor = 7;
    int divident = 22;
    int ans = solve(divisor, divident);
    cout << "ans :" << ans << endl;
    int prescion;
    cout << "enter precision" << endl;
    cin >> prescion;
    double finalAns = ans;
    double step = 0.1;
    for (int i = 0; i < prescion; i++)
    {
        for (double j = finalAns; j * divisor <= divident; j = j + step)
        {
            finalAns = j;
        }
        step = step / 10;
    }

    cout << "final answer :" << finalAns;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
ans :3
enter precision
2
final answer :3.14
```

Q) Find the odd occurring element in array

```cpp
#include <iostream>
#include <vector>

using namespace std;

int binarySearch(vector<int>arr){
    int s=0;
    int e=arr.size()-1;
    int mid=s+(e-s)/2;
    while(s<=e){
        if(s==e){
            return s;
        }
        if(mid % 2 == 0){
            if(arr[mid]==arr[mid+1]){
                s=mid+2;
            }
            else{
                e=mid;
            }
        }
        else{
            if(arr[mid]==arr[mid-1]){
                s=mid+1;
            }
            else{
                e=mid-1;
            }
        }
        mid=s+(e-s)/2;
    }
    return -1;
}
int main()
{
    vector <int>arr{1,1,2,2,3,3,4,4,5,6,6};
    int find=binarySearch(arr);
    cout<<"value is "<<arr[find]<<endl;
    cout<<"index at "<<find;
    return 0;
}
```

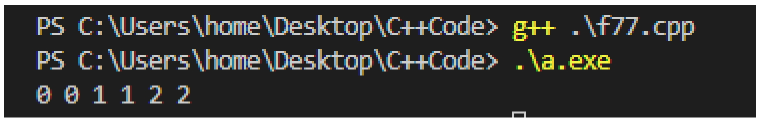


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
value is 5
index at 8
```

Week 3 – Assignments

Q sort colors(LeetCode -75. Sort Colors)

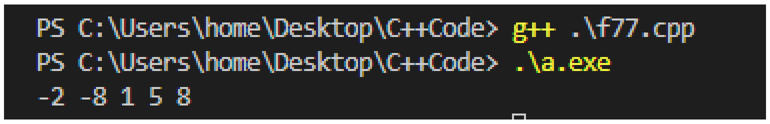
```cpp
#include <iostream>
#include <vector>

using namespace std;

void solveOrder(vector <int>arr){
    int m=0;
    int l=0;
    int h=arr.size()-1;
    while(m<=h){
        if(arr[m]==0){
            swap(arr[l],arr[m]);
            m++,l++;
        }
        else if(arr[m]==1){
            m++;
        }
        else{
            swap(arr[m],arr[h]);
            h--;
        }
    }
    for(int i=0;i<arr.size();i++){
        cout<<arr[i]<<" ";
    }
}
int main()
{
    vector <int>arr{2,0,2,1,1,0};
    solveOrder(arr);

}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
0 0 1 1 2 2
```

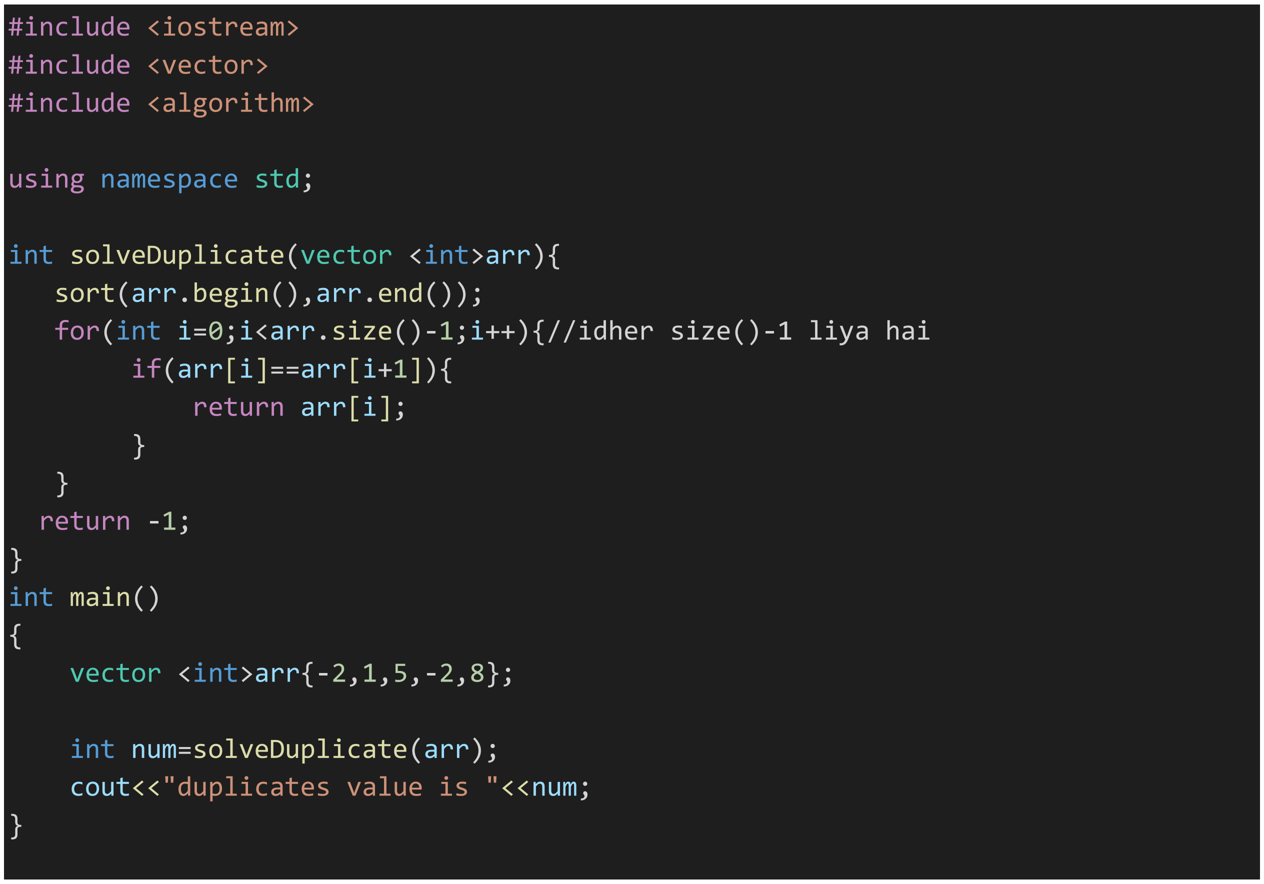
Q) Move all Negative number to the left side of an array

```cpp
#include <iostream>
#include <vector>

using namespace std;

void solveOrder(vector <int>arr){
    int l=0;
    int h=arr.size()-1;
    while(l<h){
        if(arr[l]<0){
            l++;
        }
        else if(arr[h]>0){
            h--;
        }
        else{
            swap(arr[l],arr[h]);
        }
    }
    for(int i=0;i<arr.size();i++){
        cout<<arr[i]<<" ";
    }
}
int main()
{
    vector <int>arr{-2,1,-8,5,8};
    solveOrder(arr);

}
```

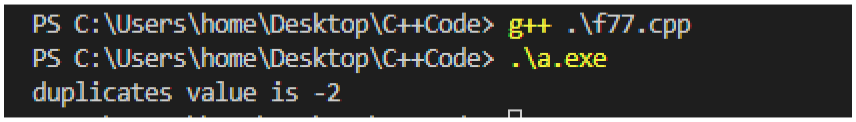


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
-2 -8 1 5 8
```

Q) Find duplicates in array using pre-defined algorithms(sort)



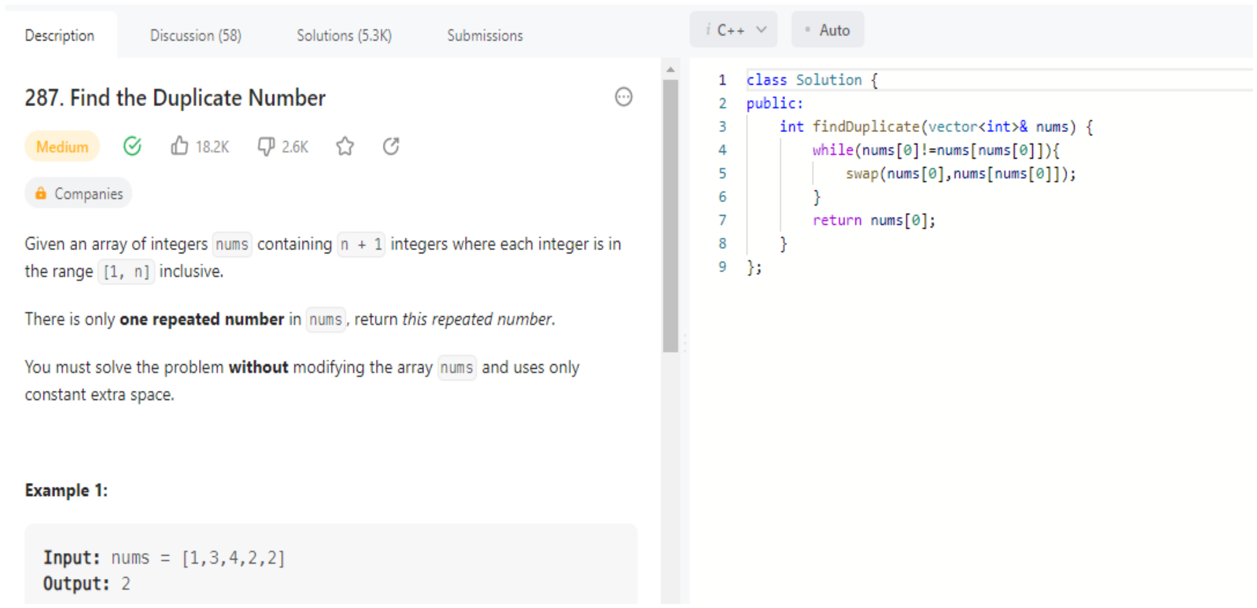
```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int solveDuplicate(vector <int>arr){
   sort(arr.begin(),arr.end());
   for(int i=0;i<arr.size()-1;i++){//idher size()-1 liya hai
        if(arr[i]==arr[i+1]){
            return arr[i];
        }
   }
  return -1;
}
int main()
{
    vector <int>arr{-2,1,5,-2,8};

    int num=solveDuplicate(arr);
    cout<<"duplicates value is "<<num;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
duplicates value is -2
```

Q) LeetCode（287. Find the Duplicate Number)

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int findDuplicate(vector <int>arr){
    int ans=-1;
    while(arr[0]!=arr[arr[0]]){
        cout<<"arr[0] = "<<arr[0]<<endl;
        cout<<"arr[arr[0]] = "<<arr[arr[0]]<<endl;
        swap(arr[0],arr[arr[0]]);

    }
    ans =arr[0];
    return ans;
}
int main()
{
    vector <int>arr{3,1,3,4,2};
    int find=findDuplicate(arr);
    cout<<"duplicate is "<<find;
}
```



287. Find the Duplicate Number

Given an array of integers `nums` containing `n + 1` integers where each integer is in the range `[1, n]` inclusive.

There is only **one repeated number** in `nums`, return *this repeated number.*

You must solve the problem **without** modifying the array `nums` and uses only constant extra space.

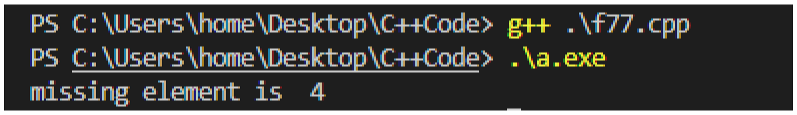
Example 1:

```
Input: nums = [1,3,4,2,2]
Output: 2
```

```cpp
class Solution {
public:
    int findDuplicate(vector<int>& nums) {
        while(nums[0]!=nums[nums[0]]){
            swap(nums[0],nums[nums[0]]);
        }
        return nums[0];
    }
};
```

Q) Missing elements from an array with duplicates

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

void findMissing(int *arr, int n)
{
    for (int i = 0; i < n; i++)
    {
        int index = abs(arr[i]);
        if (arr[index - 1] > 0)
        {
            arr[index - 1] *= -1;//mark
        }
    }

    for (int i = 0; i < n; i++)
    {
        if (arr[i] > 0)
        {
            cout << "missing element is  " <<i + 1 << " ";
        }
    }
}

int main()
{
    int arr[] = {1, 2, 3, 3, 5};
    int n = sizeof(arr) / sizeof(int); // 20/5
    findMissing(arr, n);
}
```

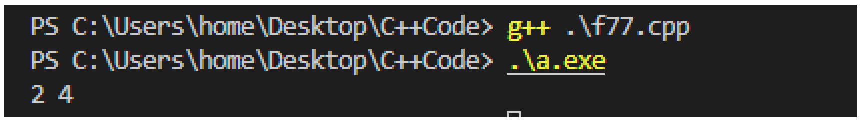


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
missing element is  4
```

Time complexity = O(n)

Space Complexity = O(1)

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

void findMissing(int *arr, int n)
{
   for(int i=0;i<n;i++){
    int index=arr[i]-1;
    if(arr[i]!=arr[index]){
        swap(arr[i],arr[index]);
    }
    else{
        i++;
    }
   }
   for(int i=0;i<n;i++){
    if(arr[i]!=i+1){
        cout<<i+1<<" ";
    }
   }
}

int main()
{
    int arr[] = {1, 3, 3, 3, 5};
    int n = sizeof(arr) / sizeof(int); // 20/5
    findMissing(arr, n);
}
```

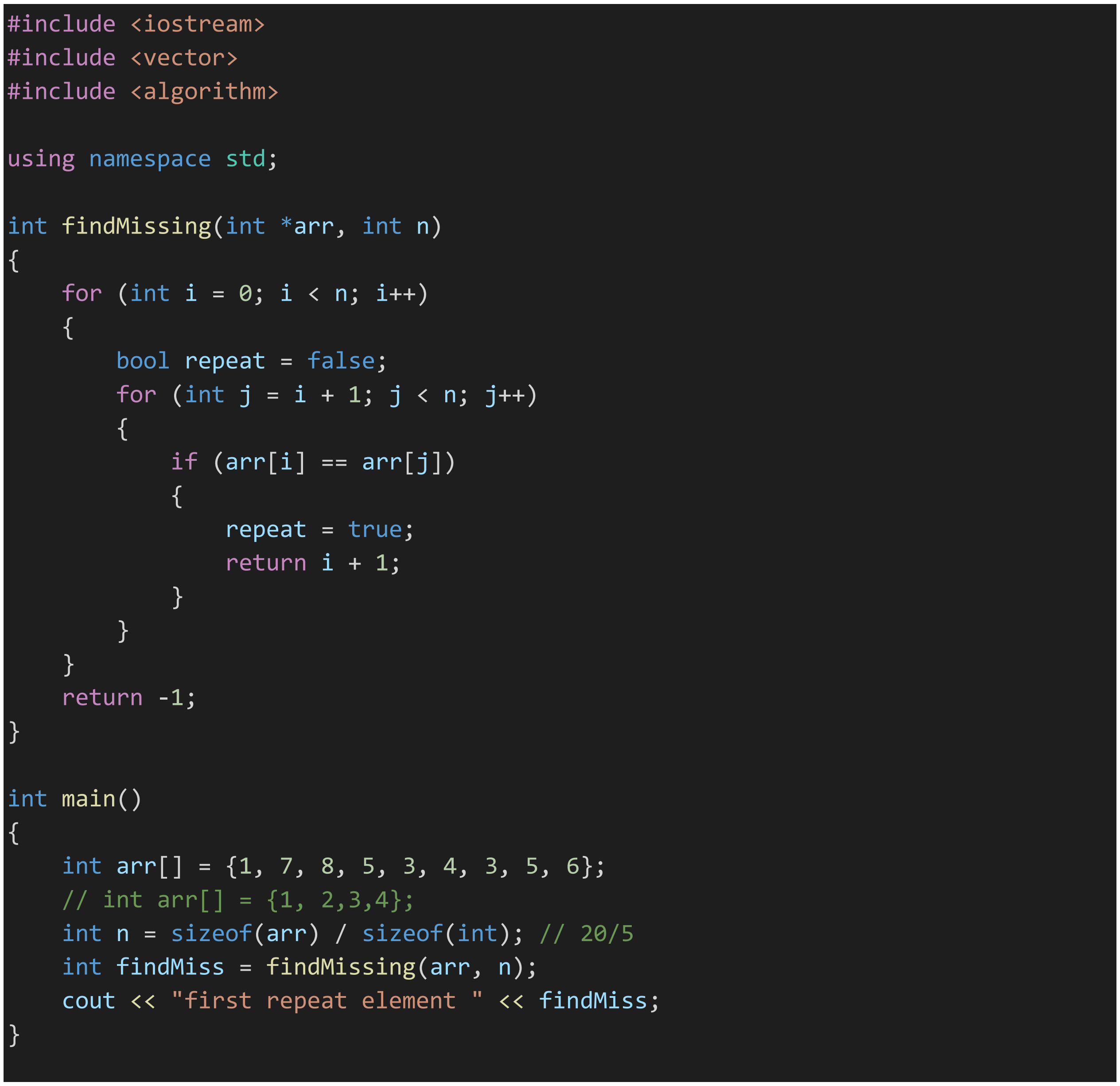


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
2 4
```

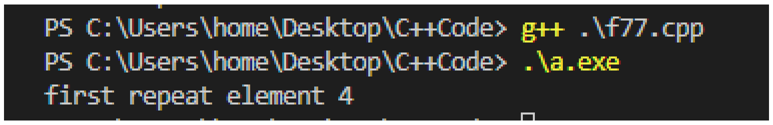
Time complexity = O(n)

Space Complexity = O(1)

Q)Find first repeating element



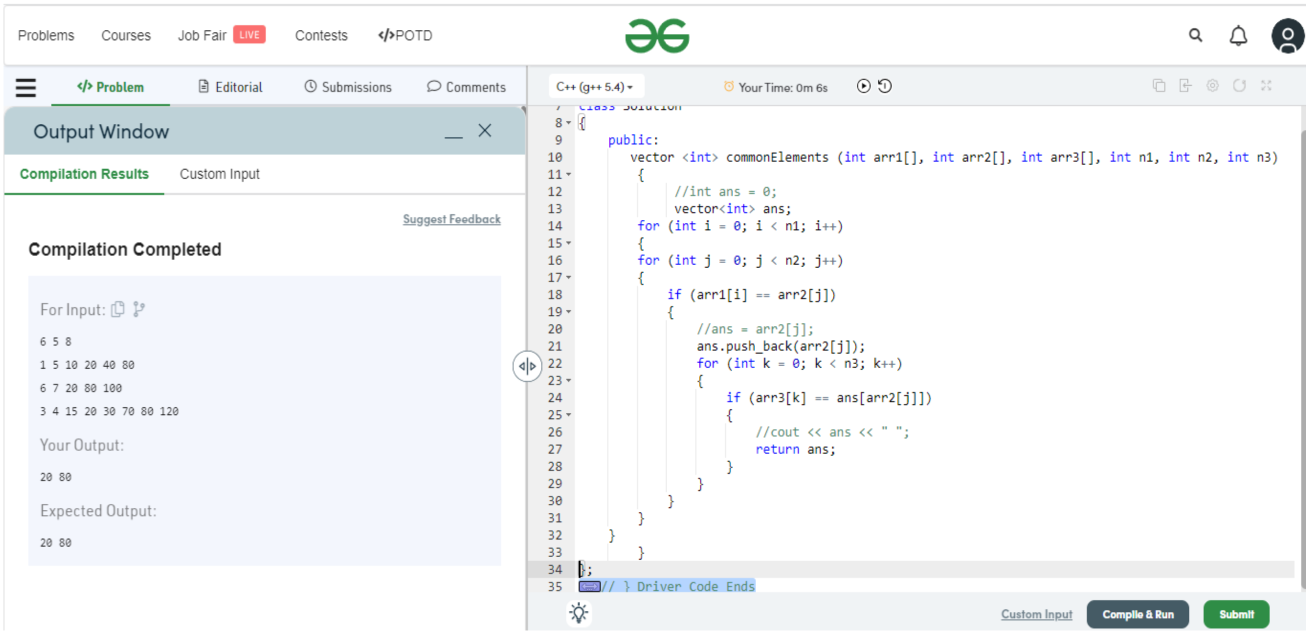
```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int findMissing(int *arr, int n)
{
    for (int i = 0; i < n; i++)
    {
        bool repeat = false;
        for (int j = i + 1; j < n; j++)
        {
            if (arr[i] == arr[j])
            {
                repeat = true;
                return i + 1;
            }
        }
    }
    return -1;
}

int main()
{
    int arr[] = {1, 7, 8, 5, 3, 4, 3, 5, 6};
    // int arr[] = {1, 2,3,4};
    int n = sizeof(arr) / sizeof(int); // 20/5
    int findMiss = findMissing(arr, n);
    cout << "first repeat element " << findMiss;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
first repeat element 4
```

Q) Common elements in 3 sorted array

1Method



Compilation Completed

For Input:

```
6 5 8
1 5 10 20 40 80
6 7 20 80 100
3 4 15 20 30 70 80 120
```

Your Output:

```
20 80
```

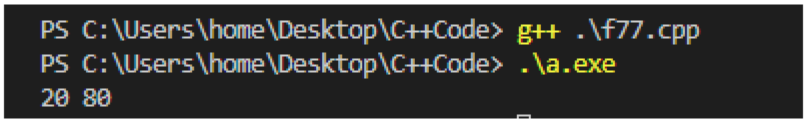
Expected Output:

```
20 80
```

```cpp
{
    public:
      vector <int> commonElements (int arr1[], int arr2[], int arr3[], int n1, int n2, int n3)
       {
           //int ans = 0;
           vector<int> ans;
       for (int i = 0; i < n1; i++)
       {
       for (int j = 0; j < n2; j++)
       {
           if (arr1[i] == arr2[j])
           {
               //ans = arr2[j];
               ans.push_back(arr2[j]);
               for (int k = 0; k < n3; k++)
               {
                   if (arr3[k] == ans[arr2[j]])
                   {
                       //cout << ans << " ";
                       return ans;
                   }
               }
           }
       }
    }
       }
};
// } Driver Code Ends
```

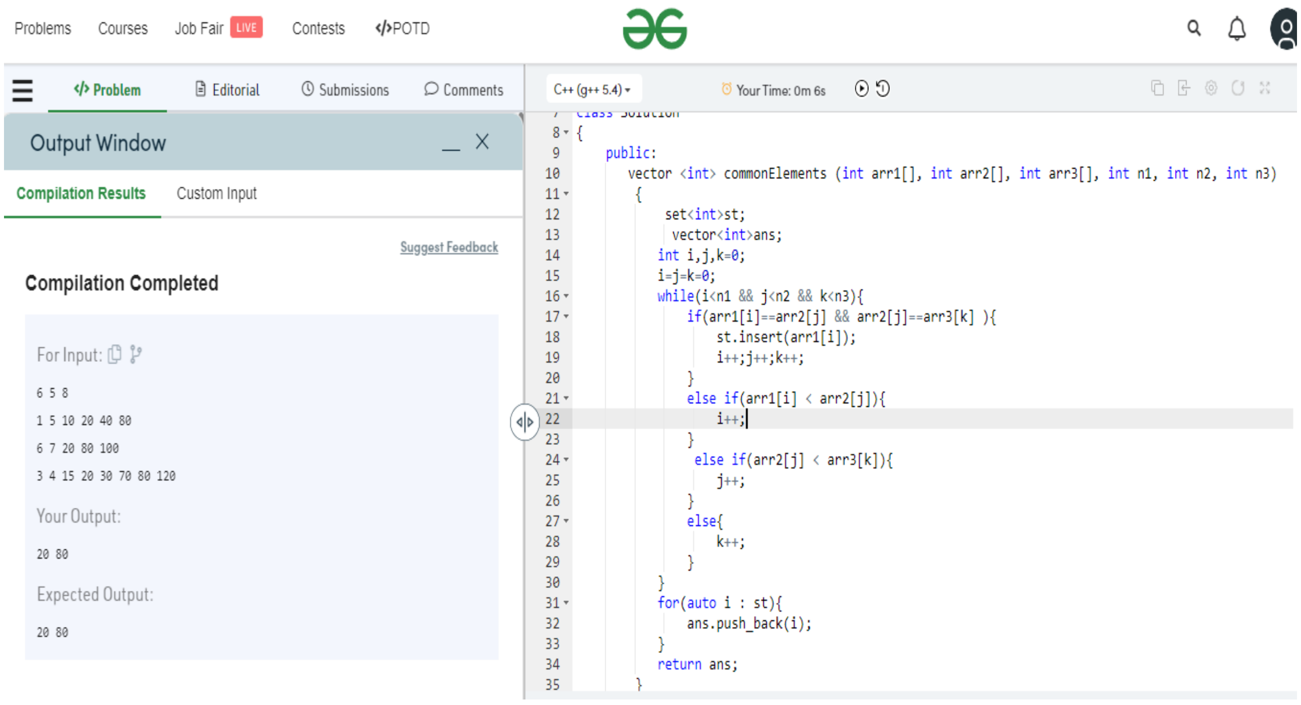
2 Method

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main()
{
    int arr1[] = {1, 5, 10, 20, 40, 80};
    int n = sizeof(arr1) / sizeof(int);
    int arr2[] = {6, 7, 20, 80, 100};
    int m = sizeof(arr2) / sizeof(int);
    int arr3[] = {3, 4, 15, 20, 30, 70, 80, 120};
    int o = sizeof(arr3) / sizeof(int);

    int ans = 0;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            if (arr1[i] == arr2[j])
            {
                ans = arr2[j];
                for (int k = 0; k < o; k++)
                {
                    if (arr3[k] == ans)
                    {
                        cout << ans << " ";
                    }
                }
            }
        }
    }

    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
20 80
```

3Method



Compilation Completed

For Input:

```
6 5 8
1 5 10 20 40 80
6 7 20 80 100
3 4 15 20 30 70 80 120
```

Your Output:

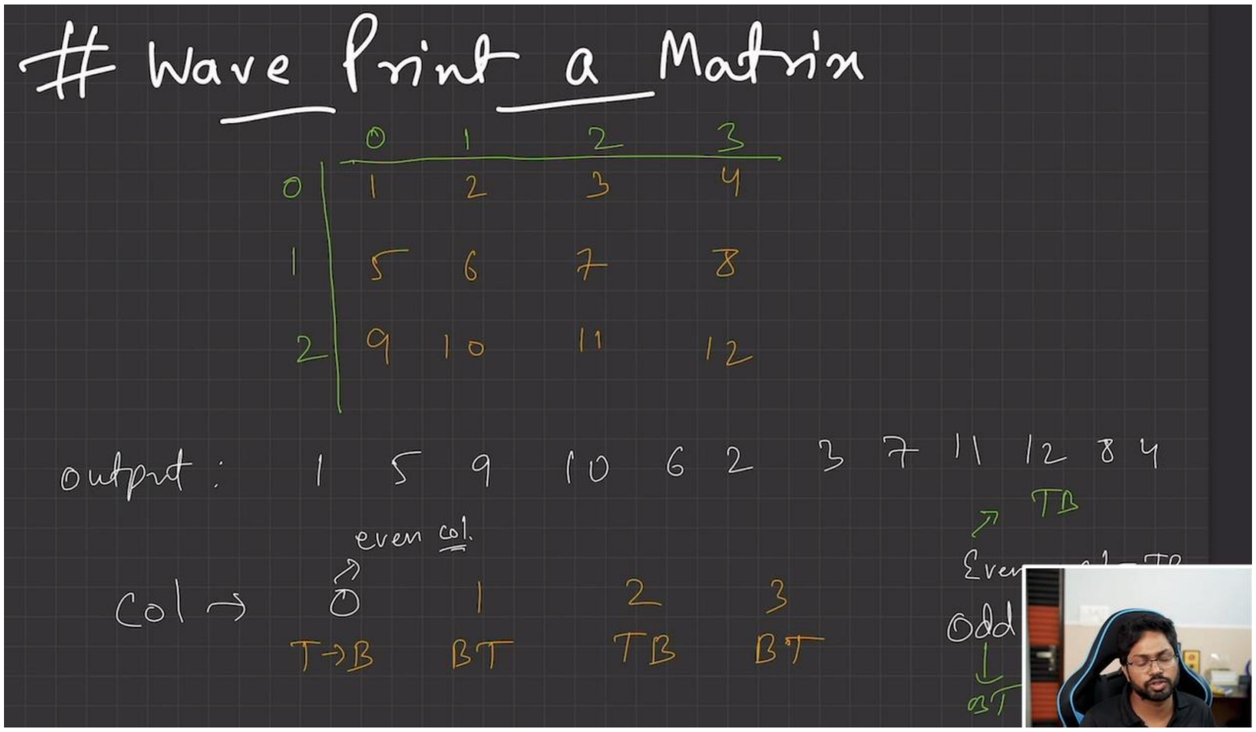
```
20 80
```

Expected Output:

```
20 80
```

```cpp
class Solution
{
    public:
       vector <int> commonElements (int arr1[], int arr2[], int arr3[], int n1, int n2, int n3)
        {
            set<int>st;
             vector<int>ans;
           int i,j,k=0;
           i=j=k=0;
           while(i<n1 && j<n2 && k<n3){
               if(arr1[i]==arr2[j] && arr2[j]==arr3[k] ){
                   st.insert(arr1[i]);
                   i++;j++;k++;
               }
               else if(arr1[i] < arr2[j]){
                   i++;
               }
                else if(arr2[j] < arr3[k]){
                   j++;
               }
               else{
                   k++;
               }
           }
           for(auto i : st){
               ans.push_back(i);
           }
           return ans;
        }
```

Q) Wave print matrix



Wave Print a Matrix

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 2 & 3 & 4 \\ 1 & 5 & 6 & 7 & 8 \\ 2 & 9 & 10 & 11 & 12 \end{array}$$

output : 1 5 9 10 6 2 3 7 11 12 8 4

even col.

col → 0 1 2 3

T→B BT TB BT

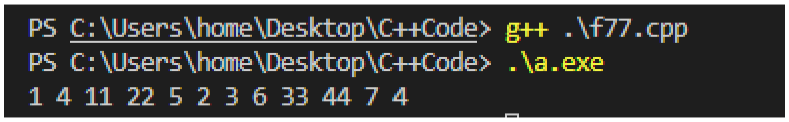
TB

Even

Odd

BT

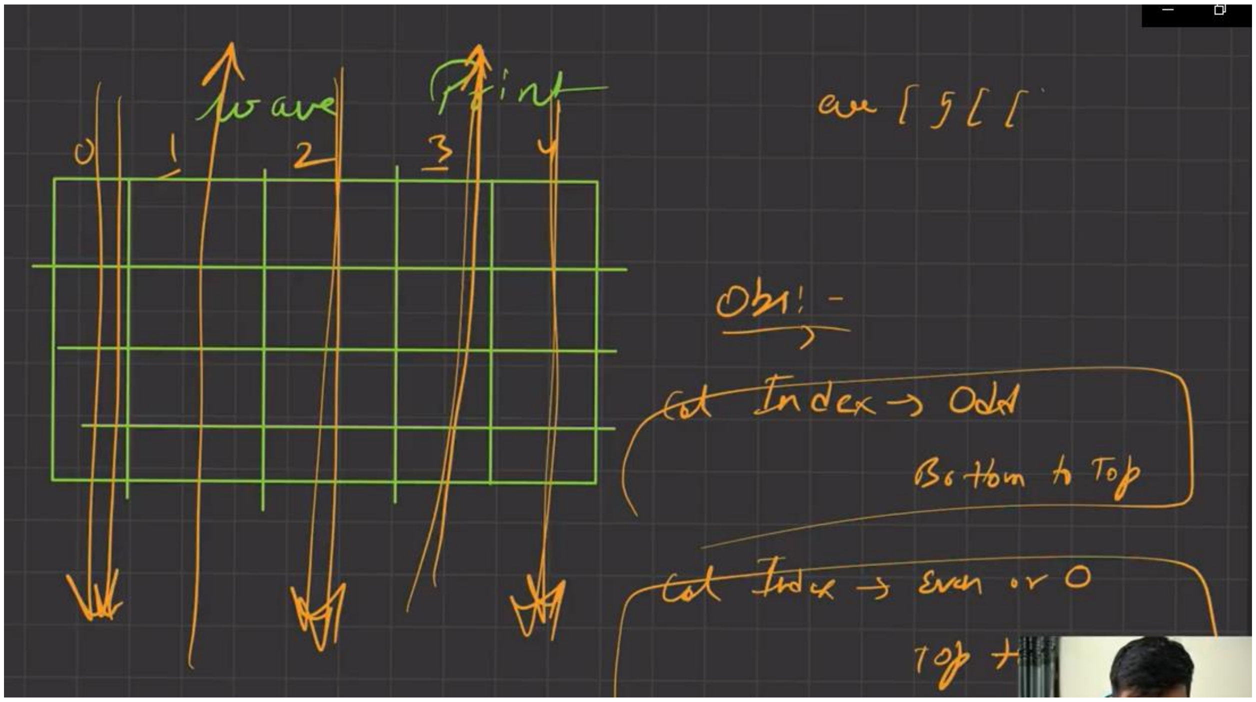
```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
using namespace std;
void waveMatrix(vector<vector<int> >v){
    int m=v.size();//row size
    int n=v[0].size();//col size
    for(int startCol=0;startCol<n;startCol++){
        if((startCol & 1)==0){
            for(int i=0;i<m;i++){
                cout<<v[i][startCol]<<" ";
            }
        }
        else{
            for(int i=m-1;i>=0;i--){
                 cout<<v[i][startCol]<<" ";
            }
        }
        //cout<<endl;
    }
}
int main()
{
   vector<vector<int> >v{
    {1,2,3,4},
    {4,5,6,7},
    {11,22,33,44,55}
   };
   waveMatrix(v);
    return 0;
}
```

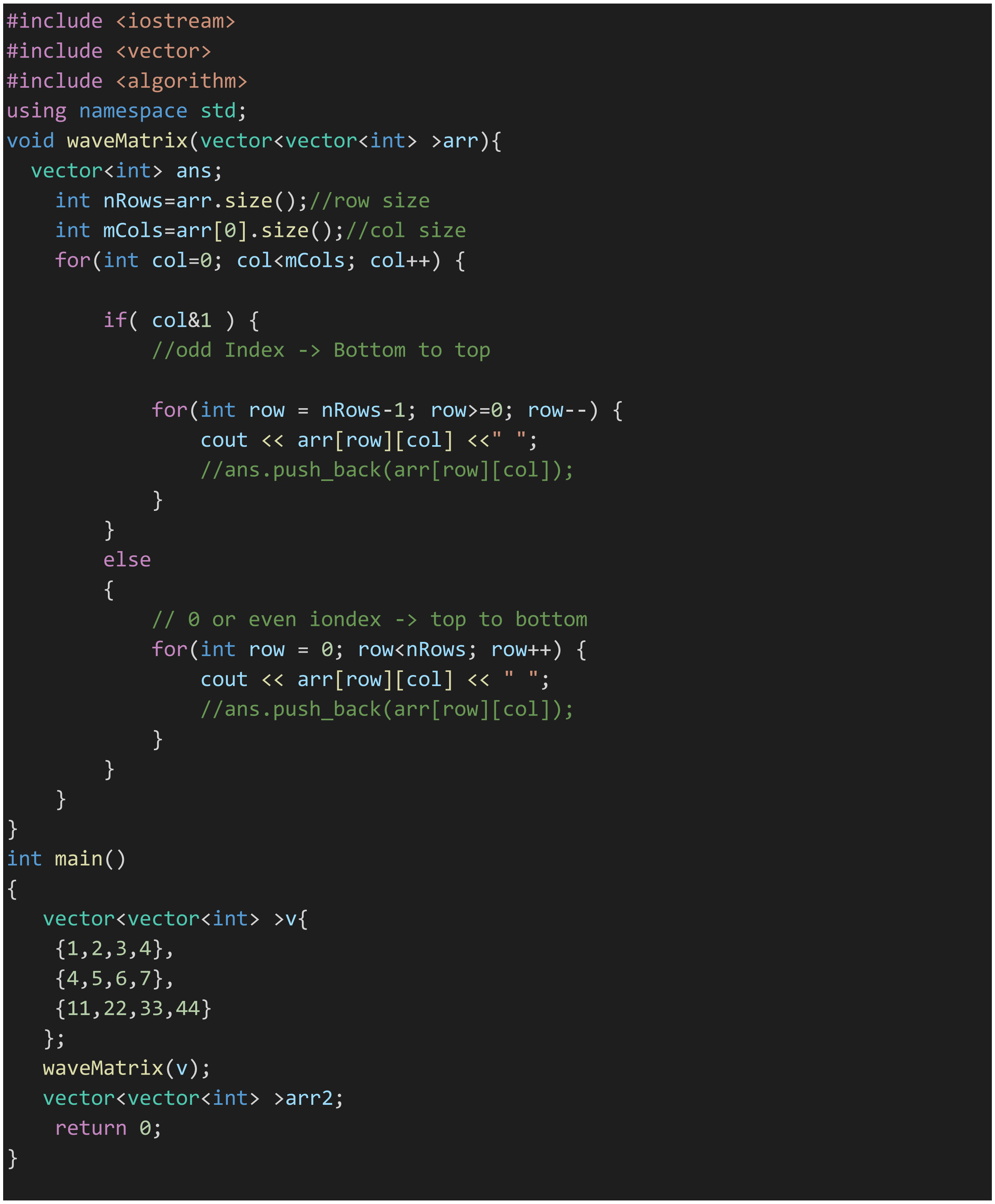


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 4 11 22 5 2 3 6 33 44 7 4
```

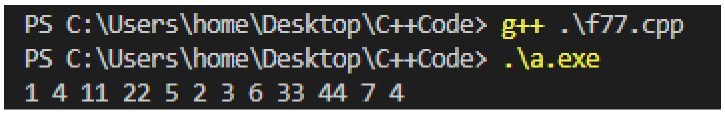
Dsa lecture (F:\Complete C++ Placement DSA Course\CodeHelp-DSA-Busted-Series-main\CodeHelp-DSA-Busted-Series-main\Lecture023 2D arrays)

wavePrint





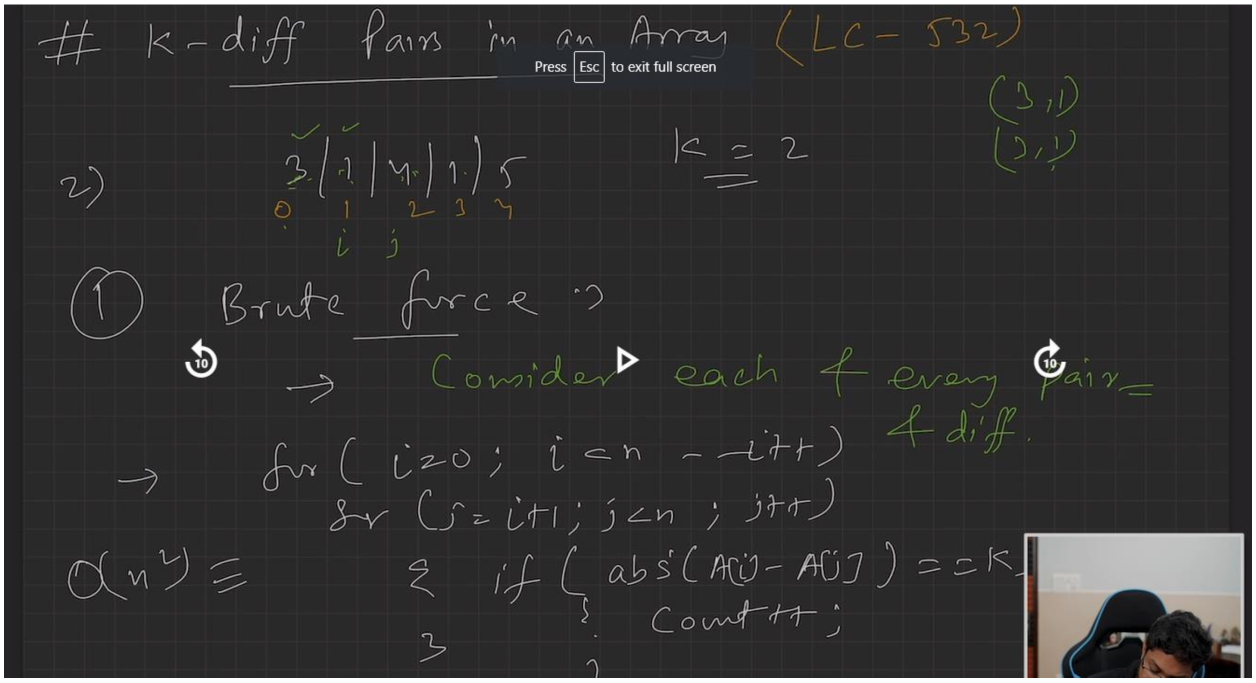
```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
void waveMatrix(vector<vector<int> >arr){
  vector<int> ans;
    int nRows=arr.size();//row size
    int mCols=arr[0].size();//col size
    for(int col=0; col<mCols; col++) {

        if( col&1 ) {
            //odd Index -> Bottom to top

            for(int row = nRows-1; row>=0; row--) {
                cout << arr[row][col] <<" ";
                //ans.push_back(arr[row][col]);
            }
        }
        else
        {
            // 0 or even iondex -> top to bottom
            for(int row = 0; row<nRows; row++) {
                cout << arr[row][col] << " ";
                //ans.push_back(arr[row][col]);
            }
        }
    }
}
int main()
{
   vector<vector<int> >v{
    {1,2,3,4},
    {4,5,6,7},
    {11,22,33,44}
   };
   waveMatrix(v);
   vector<vector<int> >arr2;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 4 11 22 5 2 3 6 33 44 7 4
```

Week 04 Assignment

Q) 532. K-diff Pairs in an Array



K-diff Pairs in an Array (LC-532)

Press Esc to exit full screen

(3,1)
(3,1)

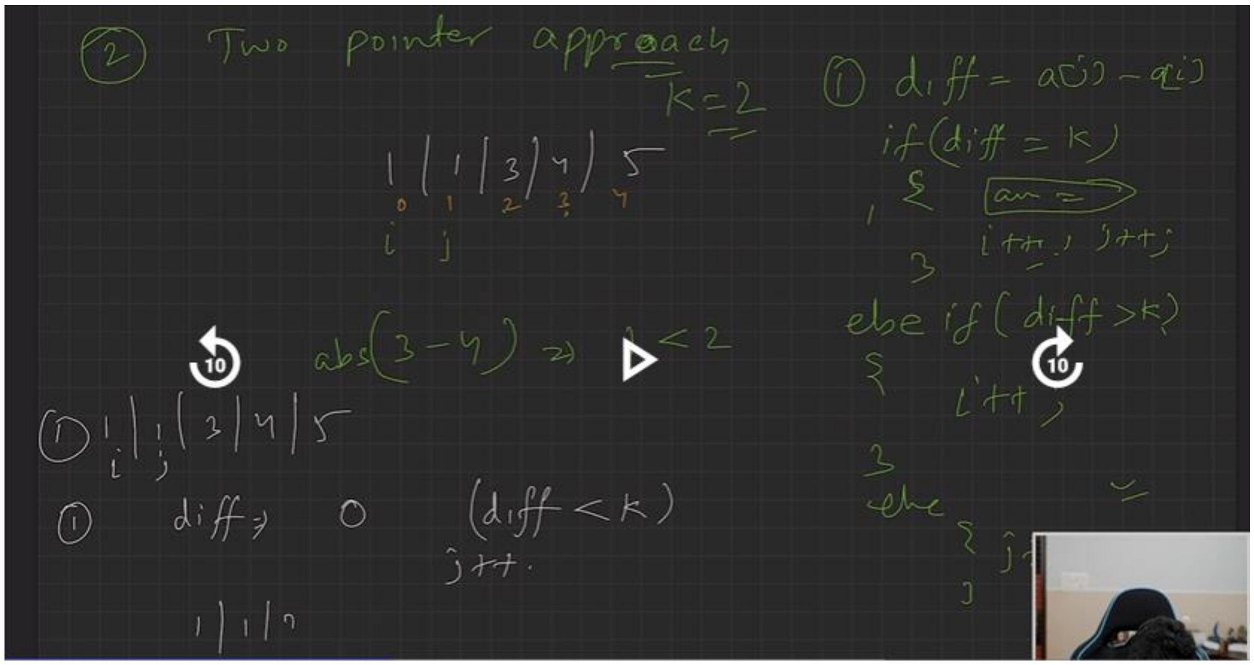
2) 3 | 1 | 4 | 1 | 5   K = 2

   0   1   2   3   4
       i   j

① Brute force :-

→ Consider each & every pair & diff.

```
→ for ( i=0; i<n --i++)
    for (j=i+1; j<n ; j++)
O(n^2) =      { if ( abs(A[i]- A[j]) ==K
                 { count++;
              }
```



② Two pointer approach

K=2

1 | 1 | 3 | 4 | 5
0   1   2   3   4
i   j

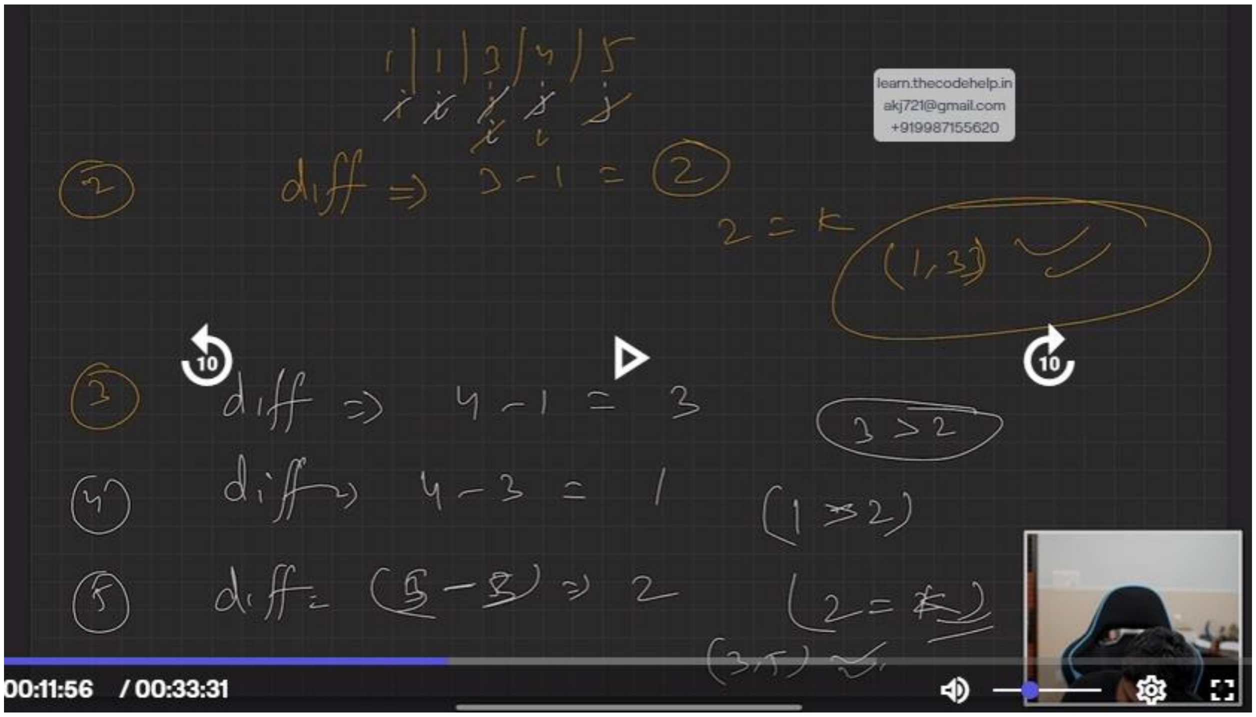
abs(3-4) ⇒ 1 < 2

① 1 | 1 | 3 | 4 | 5
  i   j

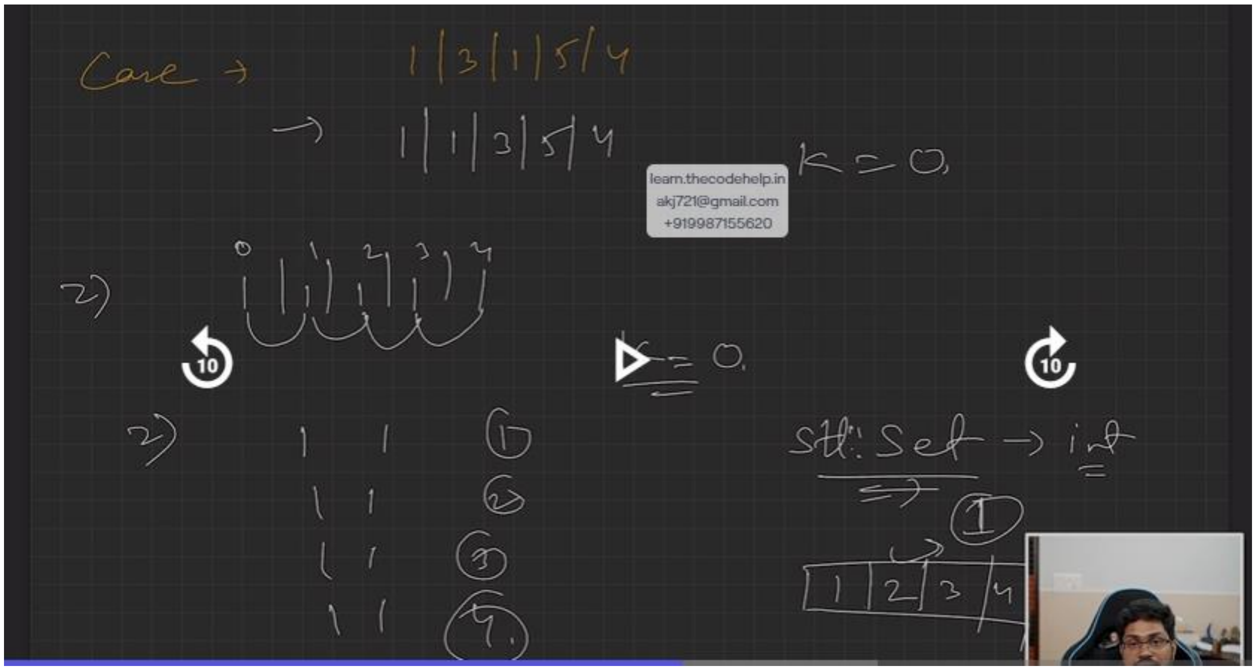
① diff ⇒ 0   (diff < k)
              j++.

1 | 1 | 7

```
① diff = a[j] - a[i]
  if (diff = k)
  {  ans =
     i++ , j++;
  }
  else if ( diff >k)
  {
     i++ ;
  }
  else
  { j
  }
```

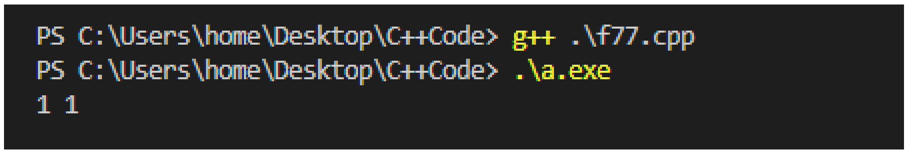


1 | 1 | 3 | 4 | 5
② diff ⇒ 3 − 1 = ②
2 = k
(1,3) ✓
③ diff ⇒ 4 − 1 = 3
3 > 2
④ diff ⇒ 4 − 3 = 1
(1 > 2)
⑤ diff = (5 − 3) ⇒ 2
(2 = k)
(3,5) ✓
learn.thecodehelp.in
akj721@gmail.com
+919987155620
00:11:56 / 00:33:31



Case →
1 | 3 | 1 | 5 | 4
→ 1 | 1 | 3 | 5 | 4
k = 0.
learn.thecodehelp.in
akj721@gmail.com
+919987155620
k = 0.
std::Set → int
1 | 2 | 3 | 4

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
using namespace std;

 void findPairs(vector<int> nums, int k) {
    sort(nums.begin(),nums.end());
    int i=0;int j=1;
    set <pair<int,int> >ans;
    while(j<nums.size()){
        int diff=nums[j]-nums[i];
        if(diff==k){
            //ans.insert(nums[i],nums[j]);
            cout<<nums[i]<<" "<<nums[j]<<endl;
            i++;j++;
        }
        else if(diff > k){
            i++;
        }
        else{
            j++;
        }
        if(i==j){
            j++;
        }

    }
     //cout<<nums[i],nums[j];
     cout<<endl;
    }
int main()
{
   vector<int>nums{1,3,1,5,4};
   int diff=0;
   findPairs(nums, diff);
    return 0;
}
```

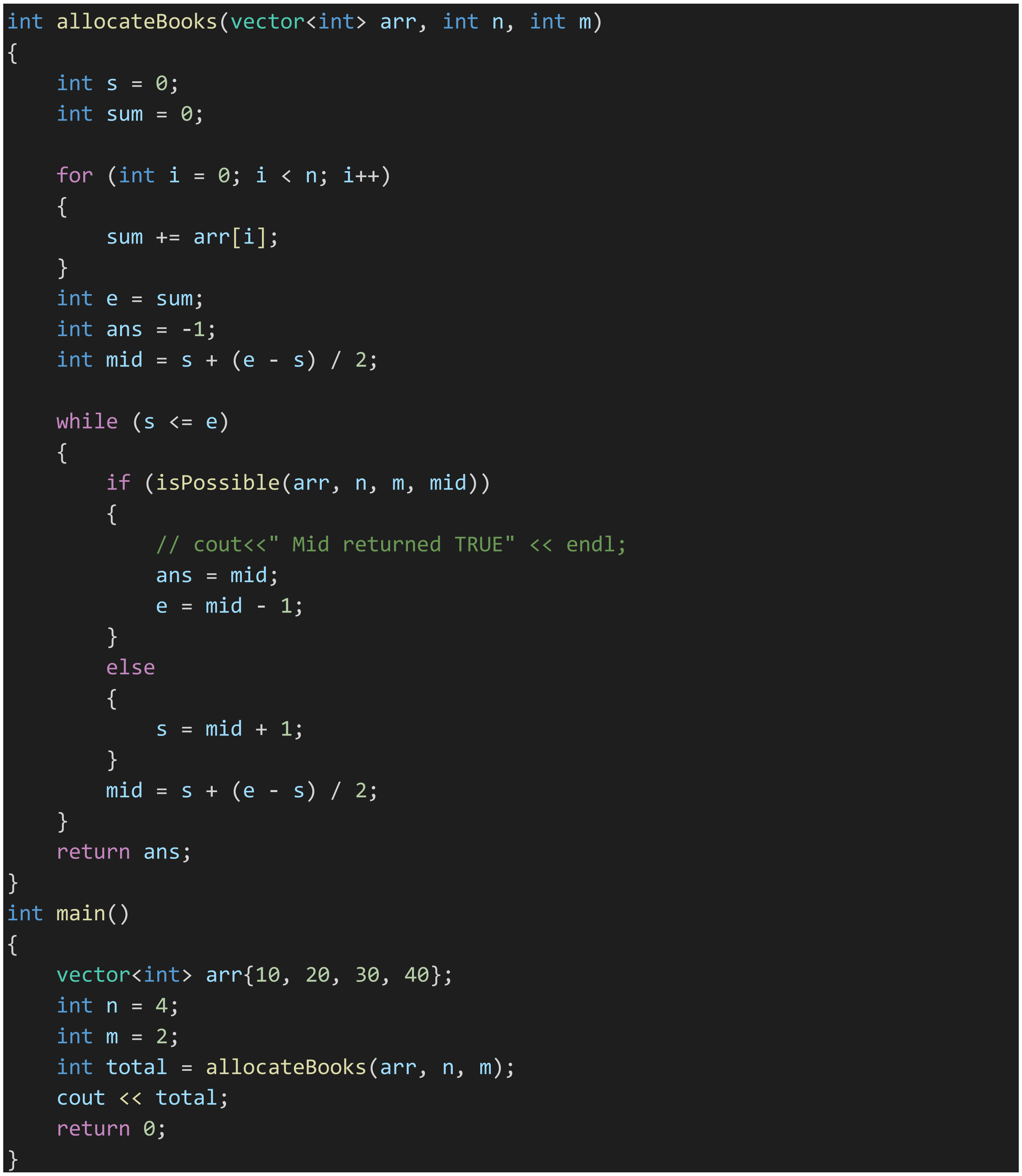


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 1
```

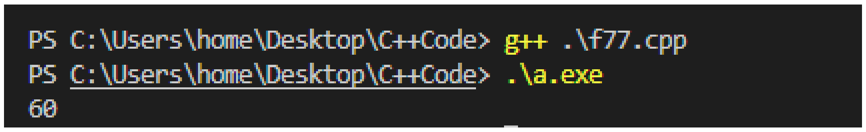
Week 04 Assignment

Q) Book Allocation Problem

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

bool isPossible(vector<int> arr, int n, int m, int mid)
{
    int studentCount = 1;
    int pageSum = 0;
    // cout << "checking for mid "<< mid <<endl;

    for (int i = 0; i < n; i++)
    {
        if (pageSum + arr[i] <= mid)
        {
            pageSum += arr[i];
        }
        else
        {
            studentCount++;
            if (studentCount > m || arr[i] > mid)
            {
                return false;
            }
            pageSum = arr[i];
        }
        if (studentCount > m)
        {
            return false;
        }
        // cout << " for i " << i << " Student "<< studentCount << " pageSum " << pageSum << endl;
    }
    return true;
}
```



```cpp
int allocateBooks(vector<int> arr, int n, int m)
{
    int s = 0;
    int sum = 0;

    for (int i = 0; i < n; i++)
    {
        sum += arr[i];
    }
    int e = sum;
    int ans = -1;
    int mid = s + (e - s) / 2;

    while (s <= e)
    {
        if (isPossible(arr, n, m, mid))
        {
            // cout<<" Mid returned TRUE" << endl;
            ans = mid;
            e = mid - 1;
        }
        else
        {
            s = mid + 1;
        }
        mid = s + (e - s) / 2;
    }
    return ans;
}
int main()
{
    vector<int> arr{10, 20, 30, 40};
    int n = 4;
    int m = 2;
    int total = allocateBooks(arr, n, m);
    cout << total;
    return 0;
}
```

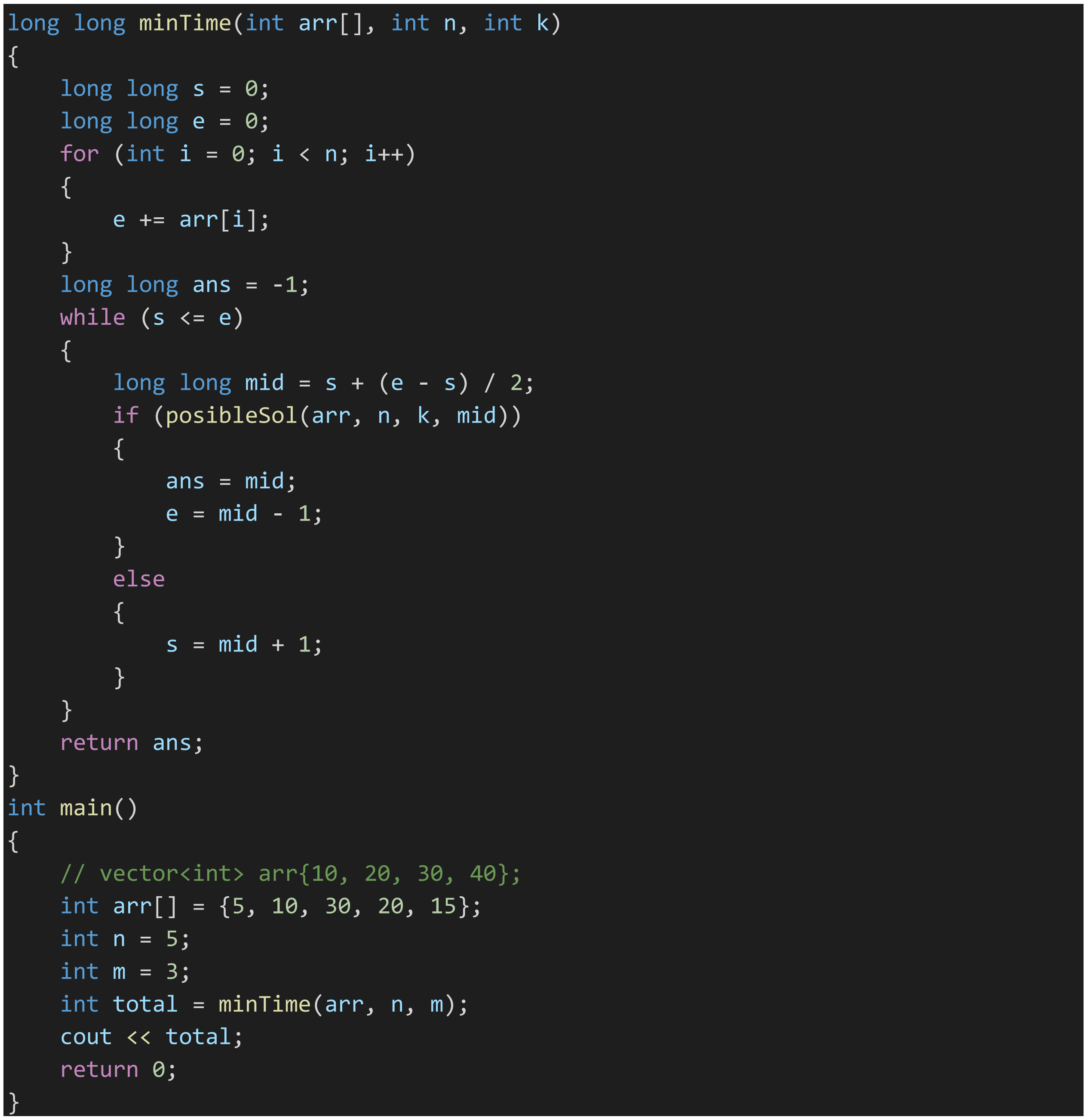


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
60
```

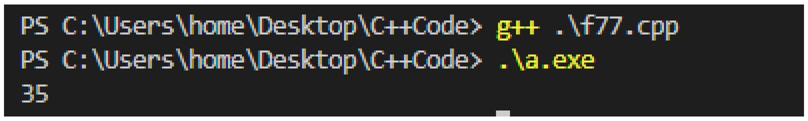
Week 04 Assignment

Q) Painter Partition Problem

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

bool posibleSol(int arr[], int n, int k, long long mid)
{
    long long totalSum = 0;
    int count = 1;
    for (int i = 0; i < n; i++)
    {
        if (arr[i] > mid)
        {
            return false;
        }
        if (totalSum + arr[i] > mid)
        {
            count++;
            totalSum = arr[i];
            if (count > k)
            {
                return false;
            }
        }
        else
        {
            totalSum += arr[i];
        }
    }
    return true;
}
```



```
long long minTime(int arr[], int n, int k)
{
    long long s = 0;
    long long e = 0;
    for (int i = 0; i < n; i++)
    {
        e += arr[i];
    }
    long long ans = -1;
    while (s <= e)
    {
        long long mid = s + (e - s) / 2;
        if (posibleSol(arr, n, k, mid))
        {
            ans = mid;
            e = mid - 1;
        }
        else
        {
            s = mid + 1;
        }
    }
    return ans;
}
int main()
{
    // vector<int> arr{10, 20, 30, 40};
    int arr[] = {5, 10, 30, 20, 15};
    int n = 5;
    int m = 3;
    int total = minTime(arr, n, m);
    cout << total;
    return 0;
}
```

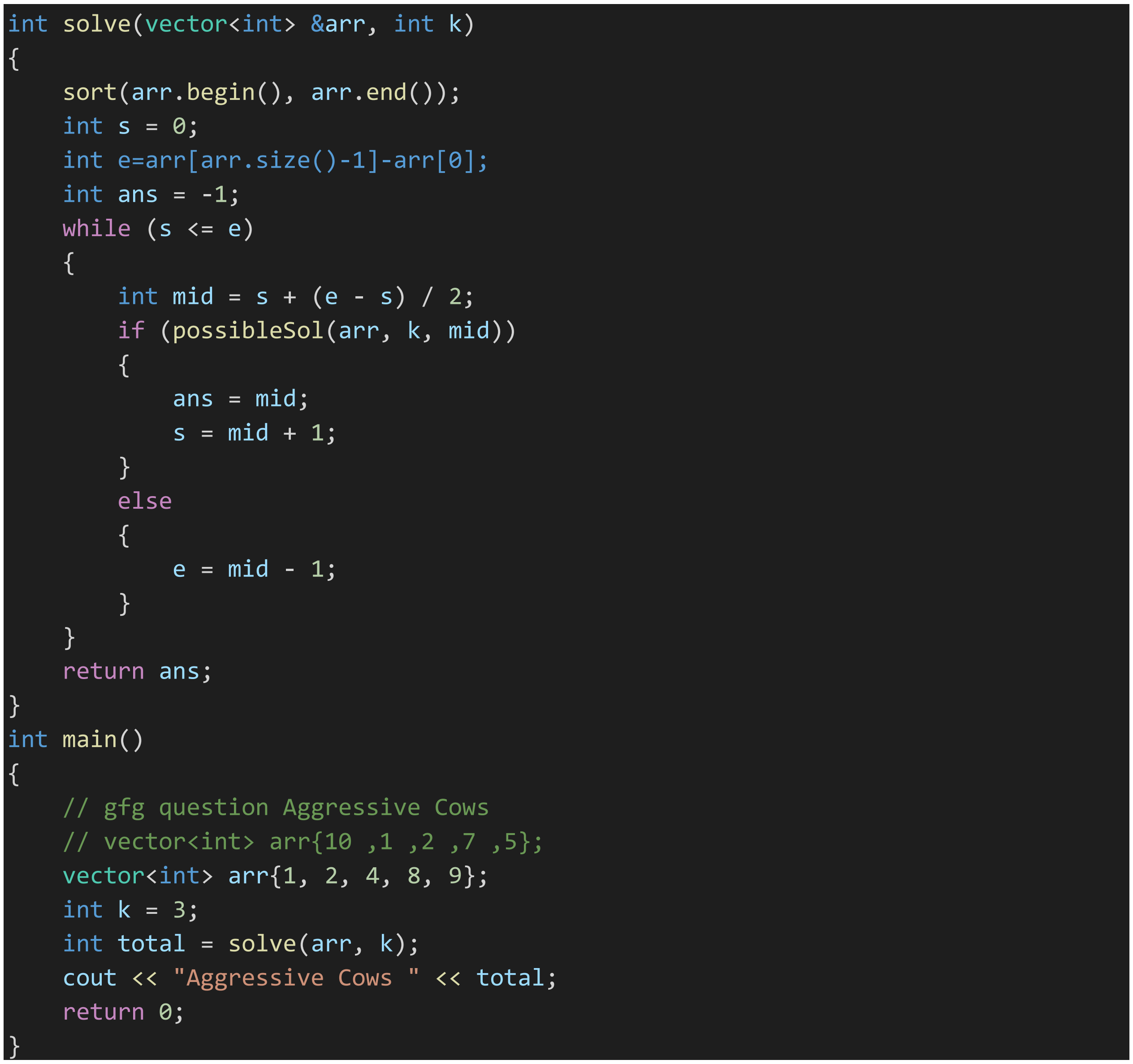


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
35
```

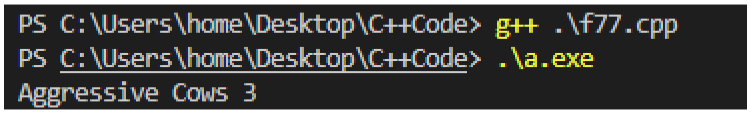
Week 04 Assignment

Q) **Aggressive Cows**

```
#include <vector>
#include <iostream>
#include <algorithm>

using namespace std;

bool possibleSol(vector<int> &arr, int k, int mid)
{
    int s = arr[0];
    int c = 1;
    for (int i = 1; i < arr.size(); i++)
    {
        if (arr[i] - s >= mid)
        {
            c++;
            s = arr[i];
        }
        if (k == c)
        {
            return true;
        }
    }
    return false;
}
```



```cpp
int solve(vector<int> &arr, int k)
{
    sort(arr.begin(), arr.end());
    int s = 0;
    int e=arr[arr.size()-1]-arr[0];
    int ans = -1;
    while (s <= e)
    {
        int mid = s + (e - s) / 2;
        if (possibleSol(arr, k, mid))
        {
            ans = mid;
            s = mid + 1;
        }
        else
        {
            e = mid - 1;
        }
    }
    return ans;
}
int main()
{
    // gfg question Aggressive Cows
    // vector<int> arr{10 ,1 ,2 ,7 ,5};
    vector<int> arr{1, 2, 4, 8, 9};
    int k = 3;
    int total = solve(arr, k);
    cout << "Aggressive Cows " << total;
    return 0;
}
```

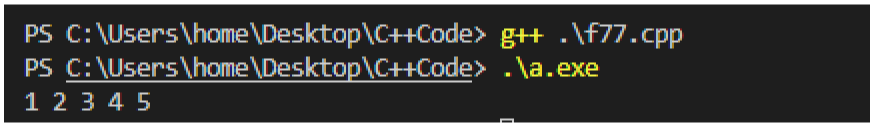


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
Aggressive Cows 3
```

Searching & Sorting – Sorting Techniques

Q) Selection sort

```cpp
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main()
{
    vector<int> arr{5, 4, 3, 2, 1};
    int n = arr.size();
    for (int i = 0; i < n - 1; i++)
    {
        int minIndex = i;
        for (int j = i + 1; j < n; j++)
        {
            if (arr[j] < arr[minIndex])
            {
                minIndex = j;
            }
        }
        swap(arr[i], arr[minIndex]);
    }
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```

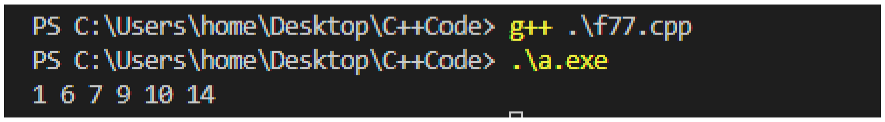


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 2 3 4 5
```

Searching & Sorting – Sorting Techniques

Q) Bubble Sort

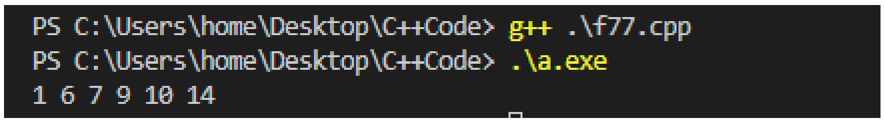
```cpp
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main()
{
    vector<int> arr{10, 1, 7, 6, 14, 9};
    int n = arr.size();
    for (int round = 1; round < n; round++)
    {
        for (int i = 0; i < n - round; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                swap(arr[i], arr[i + 1]);
            }
        }
    }
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 6 7 9 10 14
```

Another method for bubble sort

```cpp
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main()
{
    vector<int> arr{10, 1, 7, 6, 14, 9};
    int n = arr.size();
    for (int round = 1; round < n; round++)
    {
        bool swapped=false;
        for (int i = 0; i < n - round; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                swapped=true;
                swap(arr[i], arr[i + 1]);
            }
        }
        if(swapped==false){
            break;
        }
    }
    //for print
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```

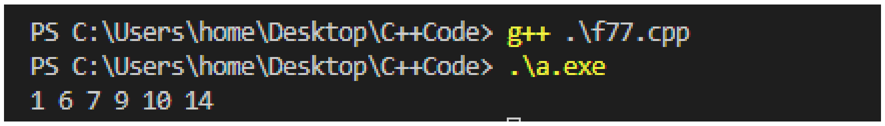


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 6 7 9 10 14
```

Searching & Sorting – Sorting Techniques

Q) Insertion Sort

```cpp
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main()
{
    vector<int> arr{10, 1, 7, 6, 14, 9};
    int n = arr.size();
    for(int round=1;round<n;round++){
        int value=arr[round];
        int j=round-1;
        for(;j>=0;j--){
            if(arr[j]>value){
                arr[j+1]=arr[j];
            }
            else{
                break;
            }
        }
        arr[j+1]=value;
    }
    //for print
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```

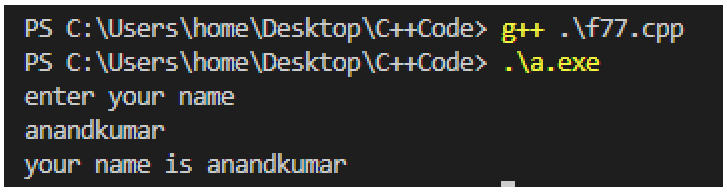


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 6 7 9 10 14
```

Char Array & String

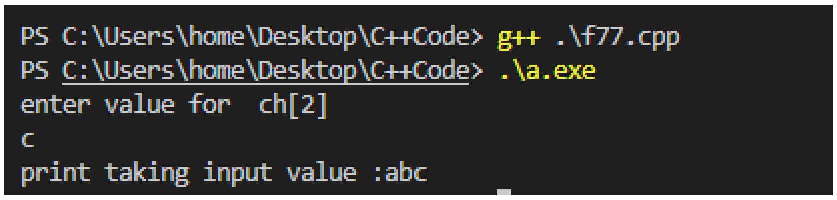
Char Arrays & String- Class I

```cpp
#include <iostream>

using namespace std;

int main()
{
    char ch[100];
    cout<<"enter your name "<<endl;
    cin>>ch;
     cout<<"your name is "<<ch<<endl;
    return 0;
}
```



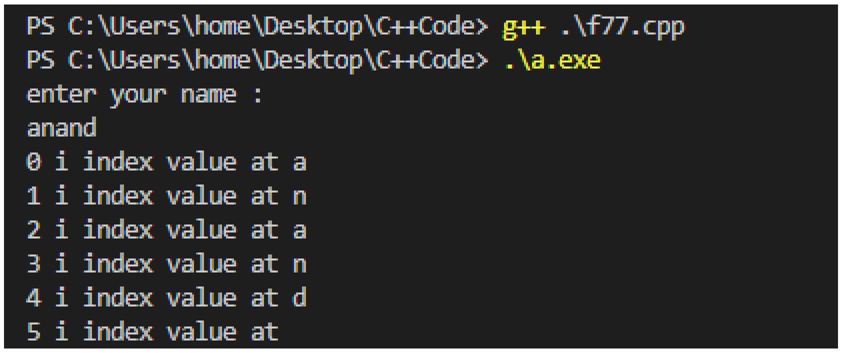
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name
anandkumar
your name is anandkumar
```

```cpp
#include <iostream>

using namespace std;

int main()
{
    char ch[100];
    ch[0]='a';
    ch[1]='b';
    cout<<"enter value for  ch[2] "<<endl;
    cin>>ch[2];
    cout<<"print taking input value :"<<ch[0]<<ch[1]<<ch[2];
    return 0;
}
```



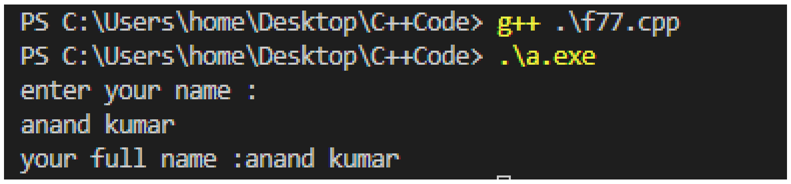
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter value for  ch[2]
c
print taking input value :abc
```

```cpp
#include <iostream>

using namespace std;

int main()
{
    char ch[100];
    cout<<"enter your name :"<<endl;
    cin>>ch;
    for(int i=0;i<6;i++){
        cout<<i<<" i index value at "<<ch[i]<<endl;;
    }
    return 0;
}
```



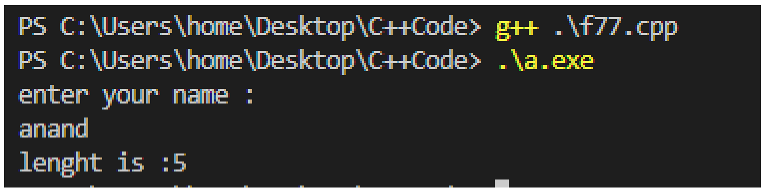
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand
0 i index value at a
1 i index value at n
2 i index value at a
3 i index value at n
4 i index value at d
5 i index value at
```

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

int main()
{
    char ch[100];
    cout<<"enter your name :"<<endl;
    cin.getline(ch,50);
    cout<<"your full name :"<<ch;
    return 0;
}
```



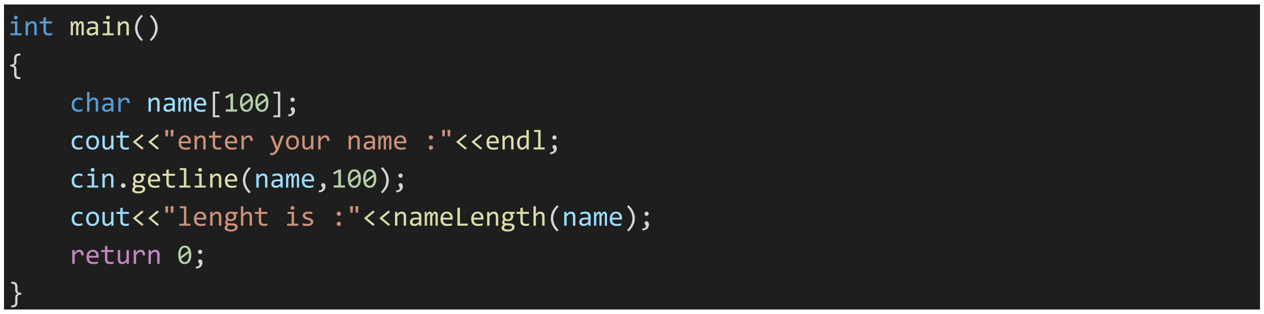
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand kumar
your full name :anand kumar
```

```cpp
#include <iostream>
#include<string.h>
using namespace std;
int nameLength(char name[]){
    int count=0;
    int i=0;
    while(name[i]!='\0'){
        i++;
        count++;
    }
    return count;
}
int main()
{
    char name[100];
    cout<<"enter your name :"<<endl;
    cin>>name;
    cout<<"lenght is :"<<nameLength(name);
    return 0;
}
```

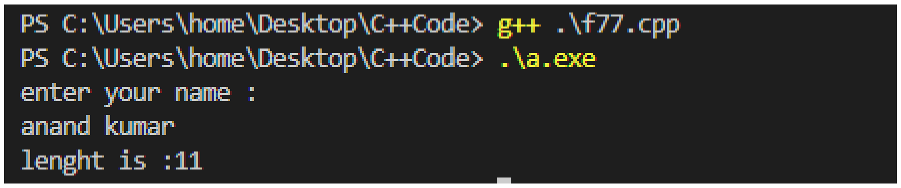


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand
lenght is :5
```

```cpp
#include <iostream>
#include<string.h>
using namespace std;
int nameLength(char name[]){
    int count=0;
    int i=0;
    while(name[i]!='\0'){
        i++;
        count++;
    }
    return count;
}
```



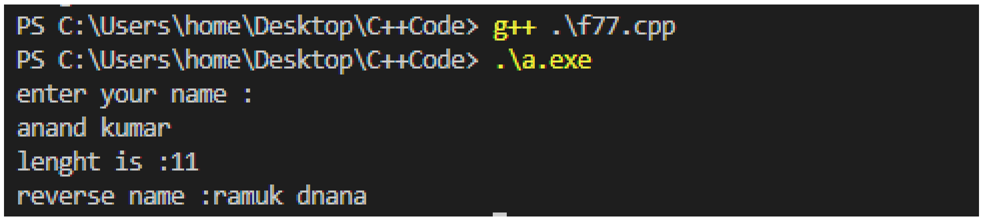
```cpp
int main()
{
    char name[100];
    cout<<"enter your name :"<<endl;
    cin.getline(name,100);
    cout<<"lenght is :"<<nameLength(name);
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand kumar
lenght is :11
```

Reverse name

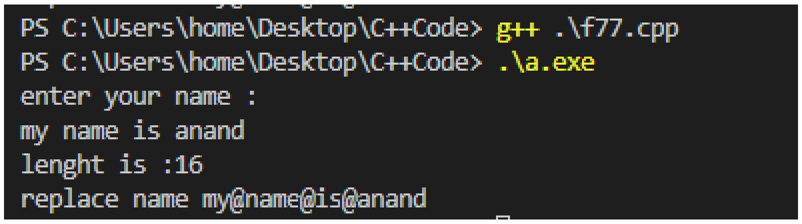
```cpp
#include <iostream>
#include<string.h>
using namespace std;
int nameLength(char name[]){
    int count=0;
    int i=0;
    while(name[i]!='\0'){
        i++;
        count++;
    }
    return count;
}
void reverseName(char name[]){
    int i=0;
    int n=nameLength(name);
    int j=n-1;
    while(i<=j){
        swap(name[i],name[j]);
        i++;
        j--;
    }
}
int main()
{
    char name[100];
    cout<<"enter your name :"<<endl;
    cin.getline(name,100);
    cout<<"lenght is :"<<nameLength(name)<<endl;
    reverseName(name);
    cout<<"reverse name :"<<name;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand kumar
lenght is :11
reverse name :ramuk dnana
```

Q) replace space with @

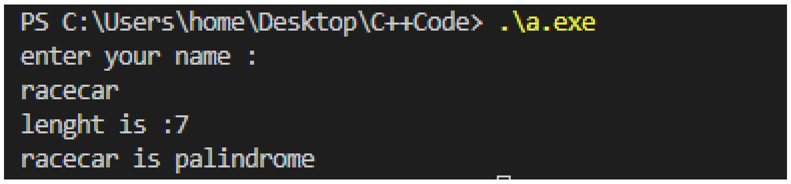
```cpp
#include <iostream>
#include <string.h>
using namespace std;
int nameLength(char name[])
{
    int count = 0;
    int i = 0;
    while (name[i] != '\0')
    {
        i++;
        count++;
    }
    return count;
}
void replaceSpaces(char name[])
{
    int n = nameLength(name);
    for (int i = 0; i < n; i++)
    {
        if (name[i] == ' ')
        {
            name[i] = '@';
        }
    }
}
int main()
{
    char name[100];
    cout << "enter your name :" << endl;
    cin.getline(name, 100);
    cout << "lenght is :" << nameLength(name) << endl;
    replaceSpaces(name);
    cout << "replace name " << name;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
my name is anand
lenght is :16
replace name my@name@is@anand
```

Q) Check Palindrome

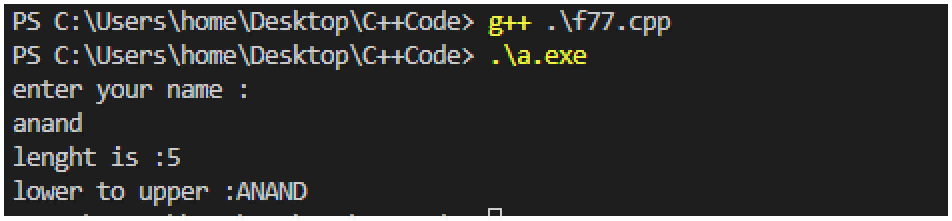
```cpp
#include <iostream>
#include <string.h>
using namespace std;
int nameLength(char name[])
{
    int count = 0;
    int i = 0;
    while (name[i] != '\0')
    {
        i++;
        count++;
    }
    return count;
}
bool checkPalindrome(char name[]){
    int i=0;
    int j=nameLength(name)-1;
    while(i<=j){
        if(name[i]!=name[j]){
            return false;
        }
        else{
            i++;
            j--;
        }
    }
    return true;
}
int main()
{
    char name[100];
    cout << "enter your name :" << endl;
    cin.getline(name, 100);
    cout << "lenght is :" << nameLength(name) << endl;
    //checkPalindrome(name);
    if(checkPalindrome(name)){
        cout<<name<<" is palindrome ";
    }
    else{
        cout<<name<<"is not palindrome";
    }
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
racecar
lenght is :7
racecar is palindrome
```

Q) lower to upper

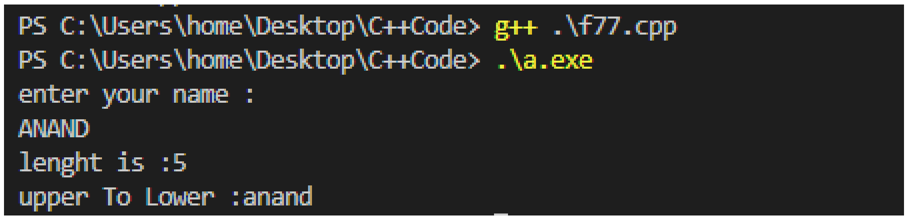
```cpp
#include <iostream>
#include <string.h>
using namespace std;
int nameLength(char name[])
{
    int count = 0;
    int i = 0;
    while (name[i] != '\0')
    {
        i++;
        count++;
    }
    return count;
}
void lowerToUpper(char name[]){
    int i=0;
    int n=nameLength(name)-1;
    while(i<=n){
        name[i]=name[i]-'a'+'A';
        i++;
    }
}
int main()
{
    char name[100];
    cout << "enter your name :" << endl;
    cin.getline(name, 100);
    cout << "lenght is :" << nameLength(name) << endl;
    lowerToUpper( name);
    cout<<"lower to upper :"<<name;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
anand
lenght is :5
lower to upper :ANAND
```

Q) upper to lower

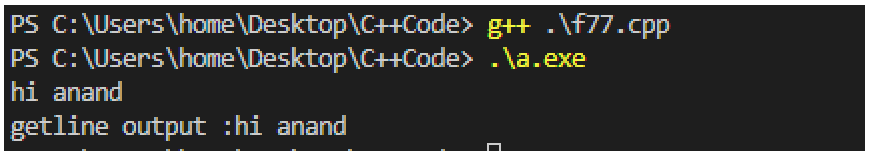
```cpp
#include <iostream>
#include <string.h>
using namespace std;
int nameLength(char name[])
{
    int count = 0;
    int i = 0;
    while (name[i] != '\0')
    {
        i++;
        count++;
    }
    return count;
}
void upperToLower(char name[]){
    int i=0;
    int n=nameLength(name)-1;
    while(i<=n){
        name[i]=name[i]-'A'+'a';
        i++;
    }
}
int main()
{
    char name[100];
    cout << "enter your name :" << endl;
    cin.getline(name, 100);
    cout << "lenght is :" << nameLength(name) << endl;
    upperToLower( name);
    cout<<"upper To Lower :"<<name;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
ANAND
lenght is :5
upper To Lower :anand
```

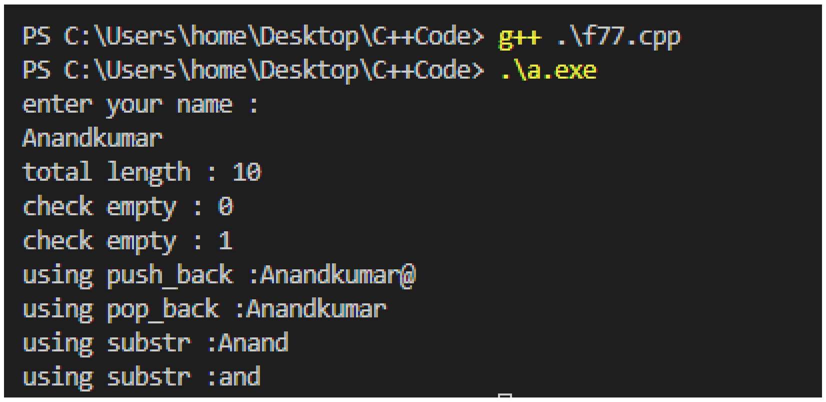
String

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

int main()
{
    string str;
    getline(cin,str);
     cout<<"getline output :"<<str<<endl;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
hi anand
getline output :hi anand
```

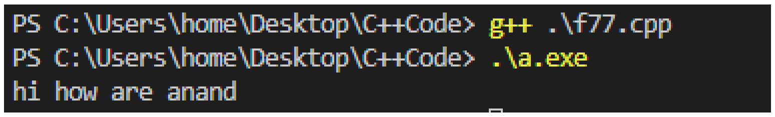
```cpp
#include <iostream>
#include <string.h>
using namespace std;

int main()
{
    string str;
    cout << "enter your name : " << endl;
    cin >> str;
    cout << "total length : " << str.length() << endl;
    cout << "check empty : " << str.empty() << endl;
    string name;
    cout << "check empty : " << name.empty() << endl;
    str.push_back('@');
    cout << "using push_back :" << str << endl;
    str.pop_back();
    cout << "using pop_back :" << str << endl;
    cout << "using substr :" << str.substr(0, 5) << endl;
    cout << "using substr :" << str.substr(2, 3) << endl;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter your name :
Anandkumar
total length : 10
check empty : 0
check empty : 1
using push_back :Anandkumar@
using pop_back :Anandkumar
using substr :Anand
using substr :and
```

Q) replace

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

int main()
{
    string str = "hi how are you";
    string word = "anand";
    str.replace(11, 3, word);
    cout << str << endl;
    return 0;
}
```



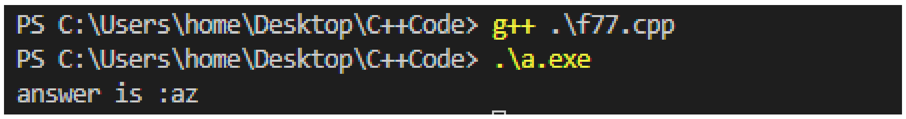
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
hi how are anand
```

Char Array & String

Char Arrays & String- Class II

1047. Remove All Adjacent Duplicates In String

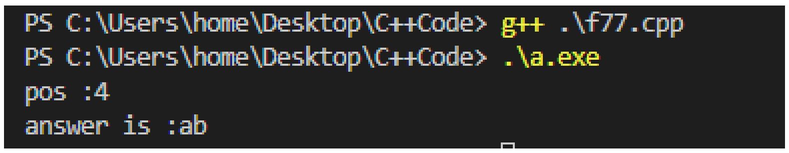
```cpp
#include <iostream>
#include <string.h>
using namespace std;

string removeDuplicates(string str)
{
    int i = 0;
    string ans = "";
    while (i < str.length())
    {
        if (ans.length() > 0)
        {
            if (ans[ans.length() - 1] == str[i])
            {
                ans.pop_back();
            }
            else
            {
                ans.push_back(str[i]);
            }
        }
        else
        {
            ans.push_back(str[i]);
        }
        i++;
    }
    return ans;
}
int main()
{
    string str = "axxybbyz";
    string ans = removeDuplicates(str);
    cout << "answer is :" << ans;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
answer is :az
```

Q) Remove occurances

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

string removeOccurance(string str, string part)
{
    int pos = str.find(part);
    cout << "pos :" << pos << endl;
    while (pos != string::npos)
    {
        str.erase(pos, part.length());
        pos = str.find(part);
    }
    return str;
}
int main()
{
    string str = "axxxxyyyyb";
    string part = "xy";
    string ans = removeOccurance(str, part);
    cout << "answer is :" << ans;
    return 0;
}
```

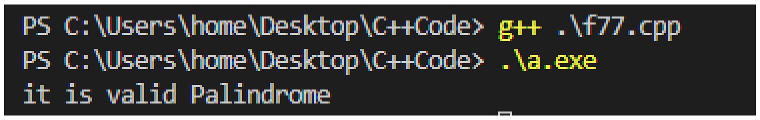


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
pos :4
answer is :ab
```

680. Valid Palindrome II(LeetCode)

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

bool checkPalindrome(string s, int i, int j)
{
    while (i <= j)
    {
        if (s[i] != s[j])
            return false;
        //else part
            i++;
            j--;
    }
    return true;
}
bool validPalindrome(string s)
{
    int i = 0;
    int j = s.length() - 1;
    while (i <= j)
    {
        if (s[i] != s[j])
        {
            return checkPalindrome(s, i + 1, j) || checkPalindrome(s, i, j - 1);
        }
        else
        {
            i++;
            j--;
        }
    }
    return true;
}
```

```cpp
int main()
{
    string s = "leverl";
    bool true1 = validPalindrome(s);
    if (true1)
    {
        cout << "it is valid Palindrome" << endl;
    }
    else
    {
        cout << "it is NOT valid Palindrome" << endl;
    }
    return 0;
}
```

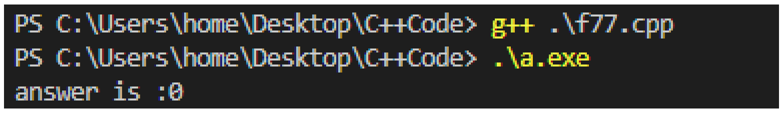


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
it is valid Palindrome
```

539. Minimum Time Difference(LeetCode)

```cpp
#include <iostream>
#include <string.h>
#include <vector>
#include <algorithm>
#include <limits.h>
using namespace std;

int findMinDifference(vector<string> timesPoints)
{
    vector<int> mintues;
    for (int i = 0; i < timesPoints.size(); i++)
    {
        string curr = timesPoints[i];
        cout << "curr :" << curr << endl;
        int hours = stoi(curr.substr(0, 2));
        cout << "hours :" << hours << endl;
        int min2 = stoi(curr.substr(3, 2));
        cout << "min2 :" << min2 << endl;
        int totalMinutes = hours * 60 + min2;
        cout << "totalMinutes :" << totalMinutes << endl;
        mintues.push_back(totalMinutes);
    }

    sort(mintues.begin(), mintues.end());
    int mini = INT_MAX;
    cout << "mini :" << mini << endl;
    int n = mintues.size();
    for (int i = 0; i < n - 1; i++)
    {
        int diff = mintues[i + 1] - mintues[i];
        mini = min(mini, diff);
        cout << "mini :" << mini << endl;
    }

    int lastDiff = (mintues[0] + 1440) - mintues[n - 1];
    mini = min(mini, lastDiff);
    cout << "mini :" << mini << endl;
    return mini;
}
```

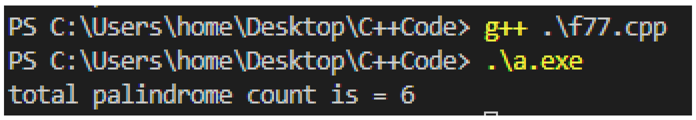
```cpp
int main()
{
    vector<string> timesPoints{
        "00:00", "23:59", "00:00"};
    int minTotal = findMinDifference(timesPoints);
    cout << "answer is :" << minTotal;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
answer is :0
```

647. Palindromic Substrings(LeetCode)

```cpp
#include <iostream>
#include <algorithm>

using namespace std;

int countPalindrome(string s,int i,int j){
     int count=0;
     while(i>=0 && j<s.length() && s[i]==s[j]){
        count++;
        i--;
        j++;
     }
     return count;
}
int main()
{
    string s="noon";
    int count=0;
    for(int i=0;i<s.length();i++){
        int odd=countPalindrome(s,i,i);
        count=count+odd;

         int even=countPalindrome(s,i,i+1);
        count=count+even;
    }
    cout<<"total palindrome count is = "<<count;
    return 0;
}
```



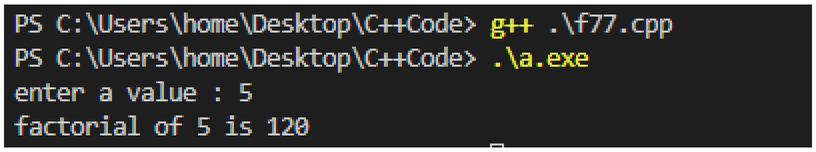
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
total palindrome count is = 6
```

Recursion & Backtracking

Recursion Level – 1

Q) factorial

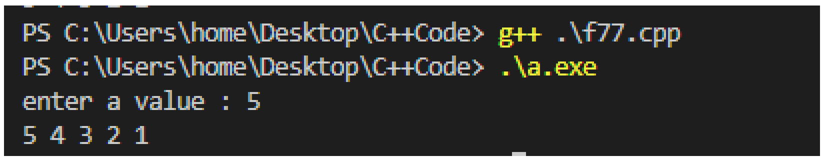
```cpp
#include <iostream>
#include <algorithm>

using namespace std;

int fact(int n){
    if(n==1)
        return 1;

    int factNumber1=fact(n-1);
    int factNumber2=n*factNumber1;
    return factNumber2;

}
int main()
{
   int n;
   cout<<"enter a value : ";
   cin>>n;
   int ans=fact(n);
   cout<<"factorial of "<<n<<" is "<<ans;
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter a value : 5
factorial of 5 is 120
```

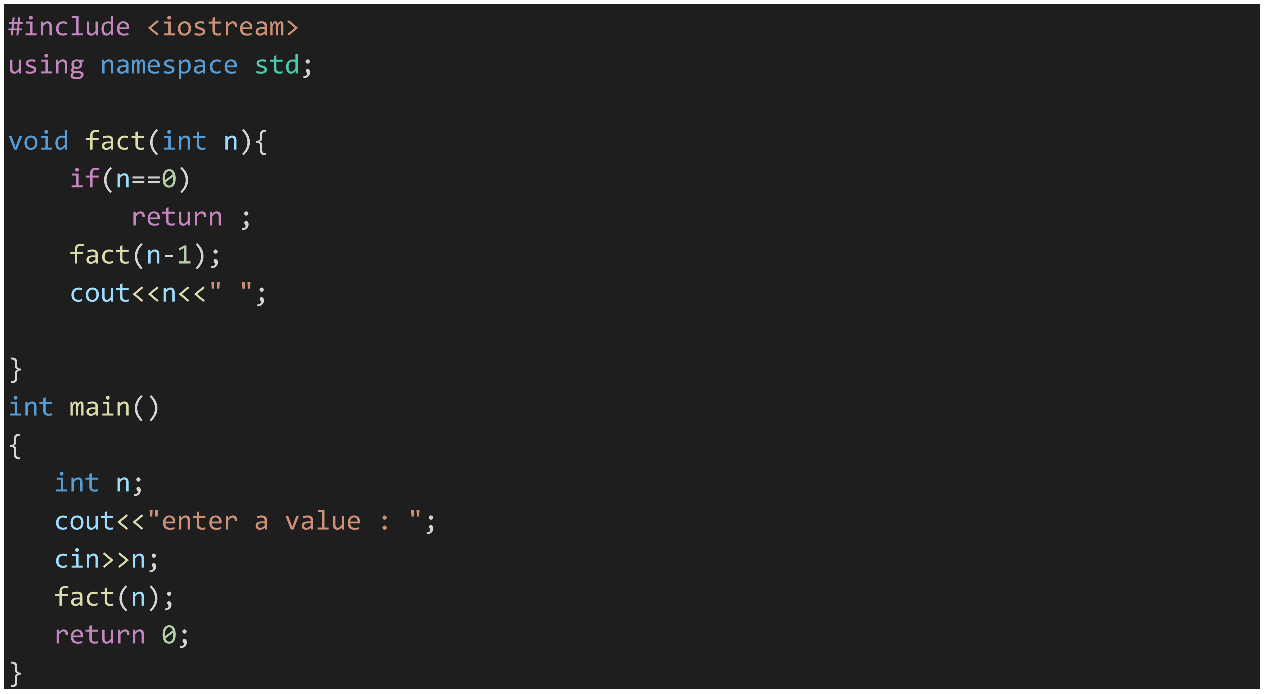
Q) reverse number

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

void fact(int n){
    if(n==0)
        return ;
    cout<<n<<" ";
    fact(n-1);

}
int main()
{
   int n;
   cout<<"enter a value : ";
   cin>>n;
   fact(n);
   return 0;
}
```

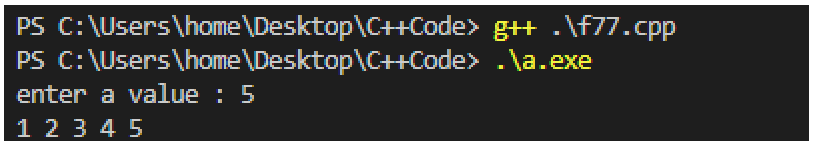


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter a value : 5
5 4 3 2 1
```

Q)Print Number

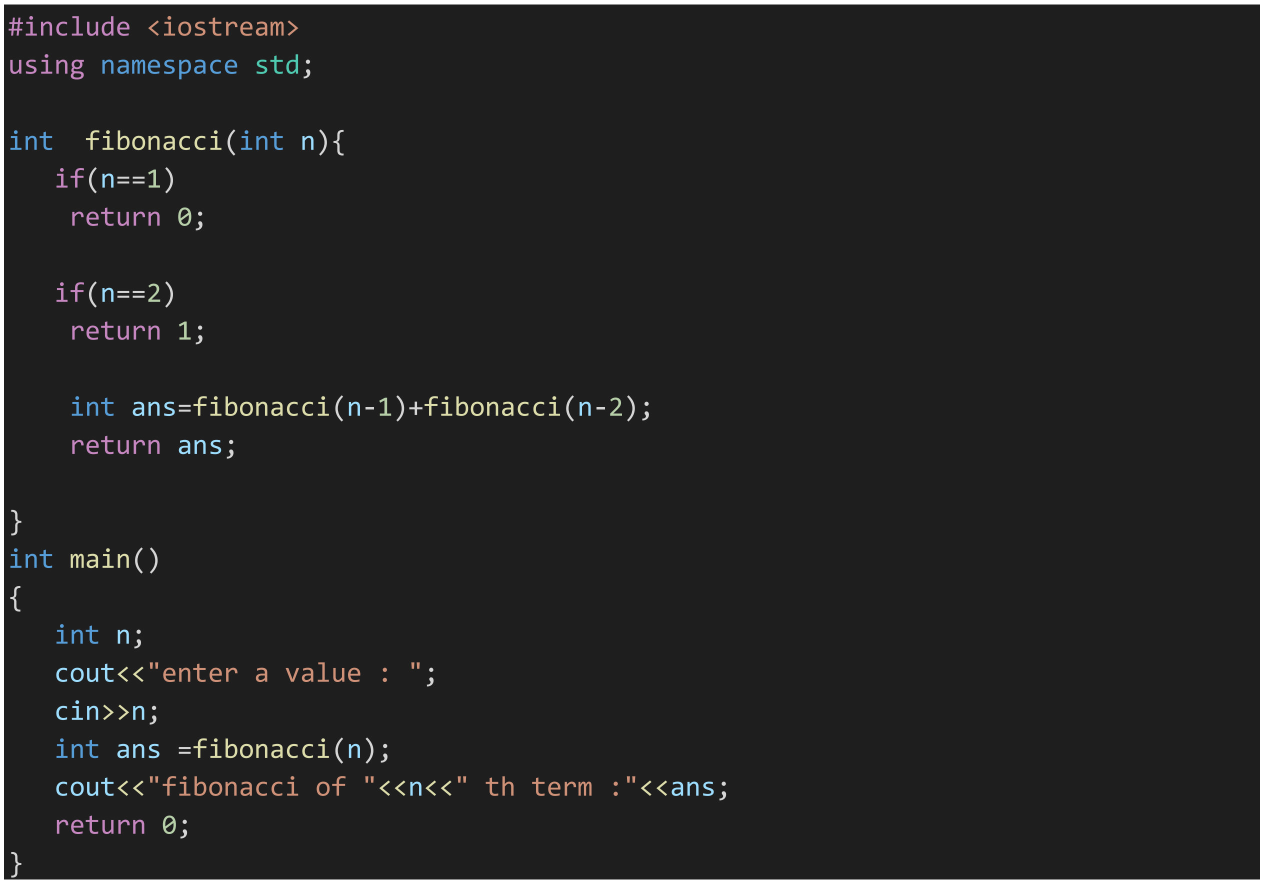


```cpp
#include <iostream>
using namespace std;

void fact(int n){
    if(n==0)
        return ;
    fact(n-1);
    cout<<n<<" ";

}
int main()
{
   int n;
   cout<<"enter a value : ";
   cin>>n;
   fact(n);
   return 0;
}
```

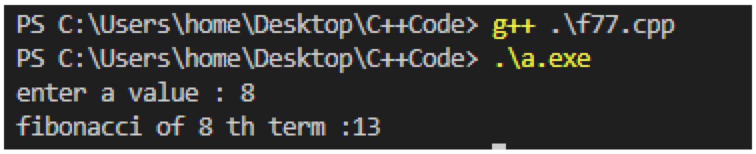


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter a value : 5
1 2 3 4 5
```

Q) Fibonacci



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

int  fibonacci(int n){
   if(n==1)
    return 0;

   if(n==2)
    return 1;

    int ans=fibonacci(n-1)+fibonacci(n-2);
    return ans;

}
int main()
{
   int n;
   cout<<"enter a value : ";
   cin>>n;
   int ans =fibonacci(n);
   cout<<"fibonacci of "<<n<<" th term :"<<ans;
   return 0;
}
```

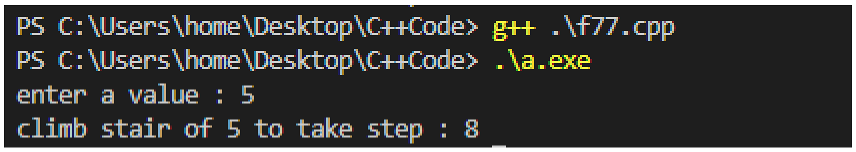


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter a value : 8
fibonacci of 8 th term :13
```

Recursion Level – 2

70. Climbing Stairs(LeetCode)

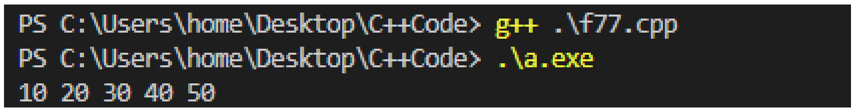
```cpp
#include <iostream>
using namespace std;

int climbStair(int n){
   if(n==0 || n==1)
      return 1;

   int ans=climbStair(n-1)+climbStair(n-2);
   return ans;
}
int main()
{
   int n;
   cout<<"enter a value : ";
   cin>>n;
   int ans =climbStair(n);
   cout<<"climb stair of "<<n<<" to take step : "<<ans;
   return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
enter a value : 5
climb stair of 5 to take step : 8 _
```

Q) Print Number In Array

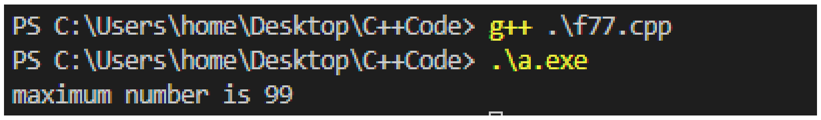
```cpp
#include <iostream>
using namespace std;

void printArray(int array[],int n,int i){
   if(i>=n){
      return;
   }
   cout<<array[i]<<" ";
   printArray(array,n,i+1);
}
int main()
{
   int array[]={10,20,30,40,50};
   int n=5;
   int i=0;
   printArray(array,n,i);
   return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
10 20 30 40 50
```

Q) Find Maximum number in array

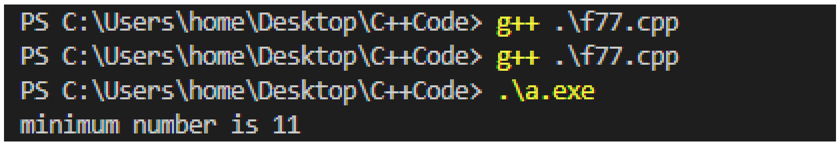
```cpp
#include <iostream>
#include<limits.h>
using namespace std;

void printArray(int arr[],int i,int n,int & maxi){
  if(i>=n){
      return;
  }
  if(arr[i]>maxi){
   maxi=arr[i];
  }
  printArray(arr,i+1,n,maxi);
}
int main()
{
   int array[]={22,66,44,99,11};

   int n=5;
   int i=0;
   int maxi=INT_MIN;
   printArray(array,i,n,maxi);
   cout<<"maximum number is "<<maxi;
   return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
maximum number is 99
```

Q) Find Minimum number in array

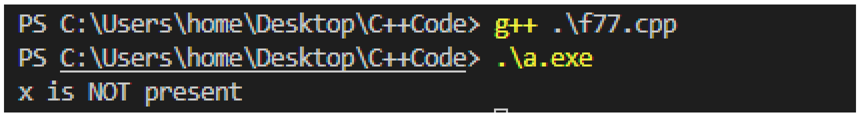
```
#include <iostream>
#include<limits.h>
using namespace std;

void printArray(int arr[],int i,int n,int & mini){
  if(i>=n){
      return;
  }
  if(arr[i]<mini){
   mini=arr[i];
  }
  printArray(arr,i+1,n,mini);
}
int main()
{
   int array[]={22,66,44,99,11};
   int n=5;
   int i=0;
   int mini=INT_MAX;
   printArray(array,i,n,mini);
   cout<<"minimum number is "<<mini;
   return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
minimum number is 11
```

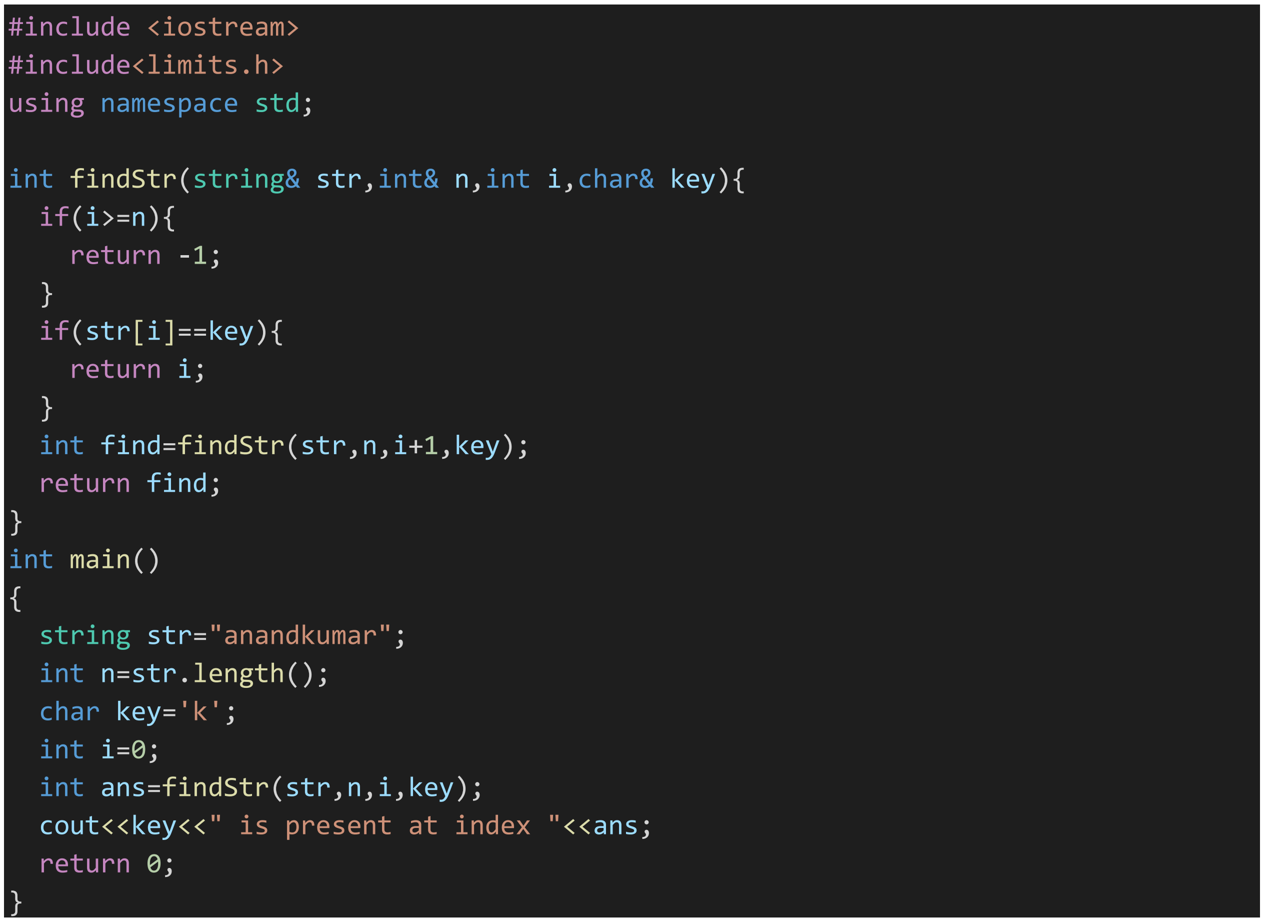
Q) Find Character in String

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

bool findStr(string str,int n,int i,char key){
  if(i>=n){
    return false;
  }
  if(str[i]==key){
    return true;
  }
  bool find=findStr(str,n,i+1,key);
  return find;
}
int main()
{
  string str="anandkumar";
  int n=str.length();
  char key='x';
  int i=0;
  bool ans=findStr(str,n,i,key);
  if(ans){
    cout<<key<<" is present ";
  }
  else{
    cout<<key<<" is NOT present ";
  }
   return 0;
}
```

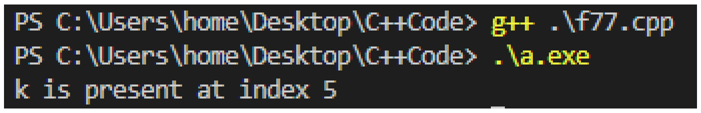


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
x is NOT present
```

Q) Find Character in String at Index



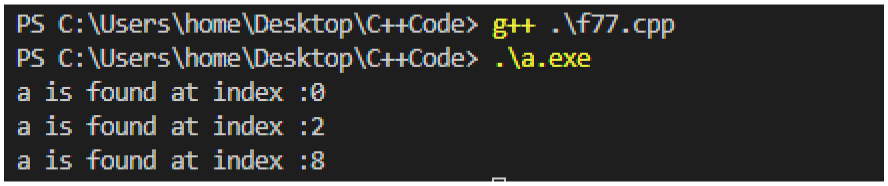
```cpp
#include <iostream>
#include<limits.h>
using namespace std;

int findStr(string& str,int& n,int i,char& key){
  if(i>=n){
    return -1;
  }
  if(str[i]==key){
    return i;
  }
  int find=findStr(str,n,i+1,key);
  return find;
}
int main()
{
  string str="anandkumar";
  int n=str.length();
  char key='k';
  int i=0;
  int ans=findStr(str,n,i,key);
  cout<<key<<" is present at index "<<ans;
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
k is present at index 5
```

Q) key find in string

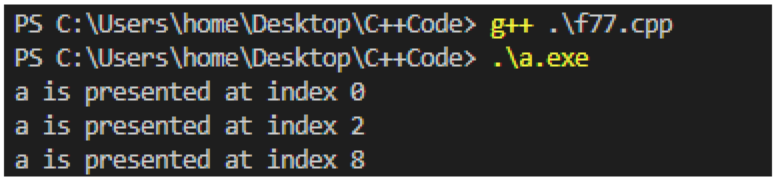
```cpp
#include <iostream>
#include<limits.h>
using namespace std;

void checkKey(string& str,int& n,int i,char& key){
  if(i>=n){
    return ;
  }
  if(str[i]==key){
    cout<<key<<" is found at index :"<<i<<endl;
  }
  return checkKey(str,n,i+1,key);

}
int main()
{
  string str="anandkumar";
  int n=str.length();
  char key='a';
  int i=0;
  checkKey(str,n,i,key);
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
a is found at index :0
a is found at index :2
a is found at index :8
```

Q) key find in string and store value to vector

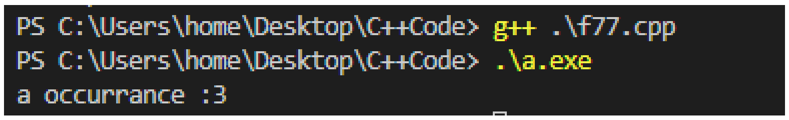
```cpp
#include <iostream>
#include<vector>
using namespace std;

void checkKey(string& str,int& n,int i,char& key,vector<int>& ans){
  if(i>=n){
    return ;
  }
  if(str[i]==key){
    ans.push_back(i);
  }
  return checkKey(str,n,i+1,key,ans);

}
int main()
{
  string str="anandkumar";
  int n=str.length();
  char key='a';
  int i=0;
  vector<int>ans;
  checkKey(str,n,i,key,ans);
  //print value
  for(auto val:ans){
    cout<<key<<" is presented at index "<<val <<" "<<endl;
  }
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
a is presented at index 0
a is presented at index 2
a is presented at index 8
```

Q) find number of occurance

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

void checkKey(string& str,int& n,int i,char& key,int& count){
  if(i>=n){
    return ;
  }
  if(str[i]==key){
    count++;
  }
  return checkKey(str,n,i+1,key,count);

}
int main()
{
  string str="anandkumar";
  int n=str.length();
  char key='a';
  int i=0;
  int count =0;
  checkKey(str,n,i,key,count);
  cout<<key<<" occurrance :"<<count;
  return 0;
}
```

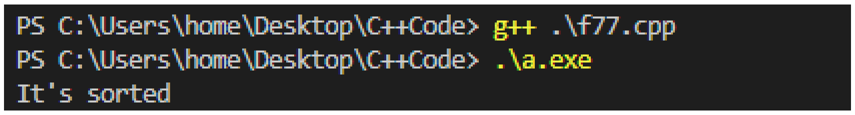


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
a occurrance :3
```

Recursion Level – 3

Q) check vector sorted

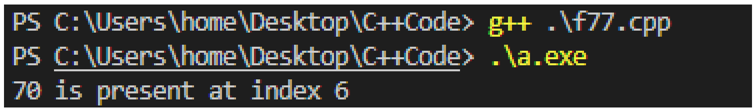
```cpp
#include <iostream>
#include<vector>
using namespace std;

bool checkSorted(vector<int>& arr,int& n,int i){
  if(i==n-1){
    return true;
  }
  if(arr[i]>arr[i+1]){
    return false;
  }
  checkSorted(arr,n,i+1);
}
int main()
{
  vector<int>arr{10,20,30,40};
  int n=arr.size();
  int i=0;
  bool check=checkSorted(arr,n,i);
  if(check){
    cout<<"It's sorted";
  }
  else{
    cout<<"It's NOT sorted";
  }
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
It's sorted
```

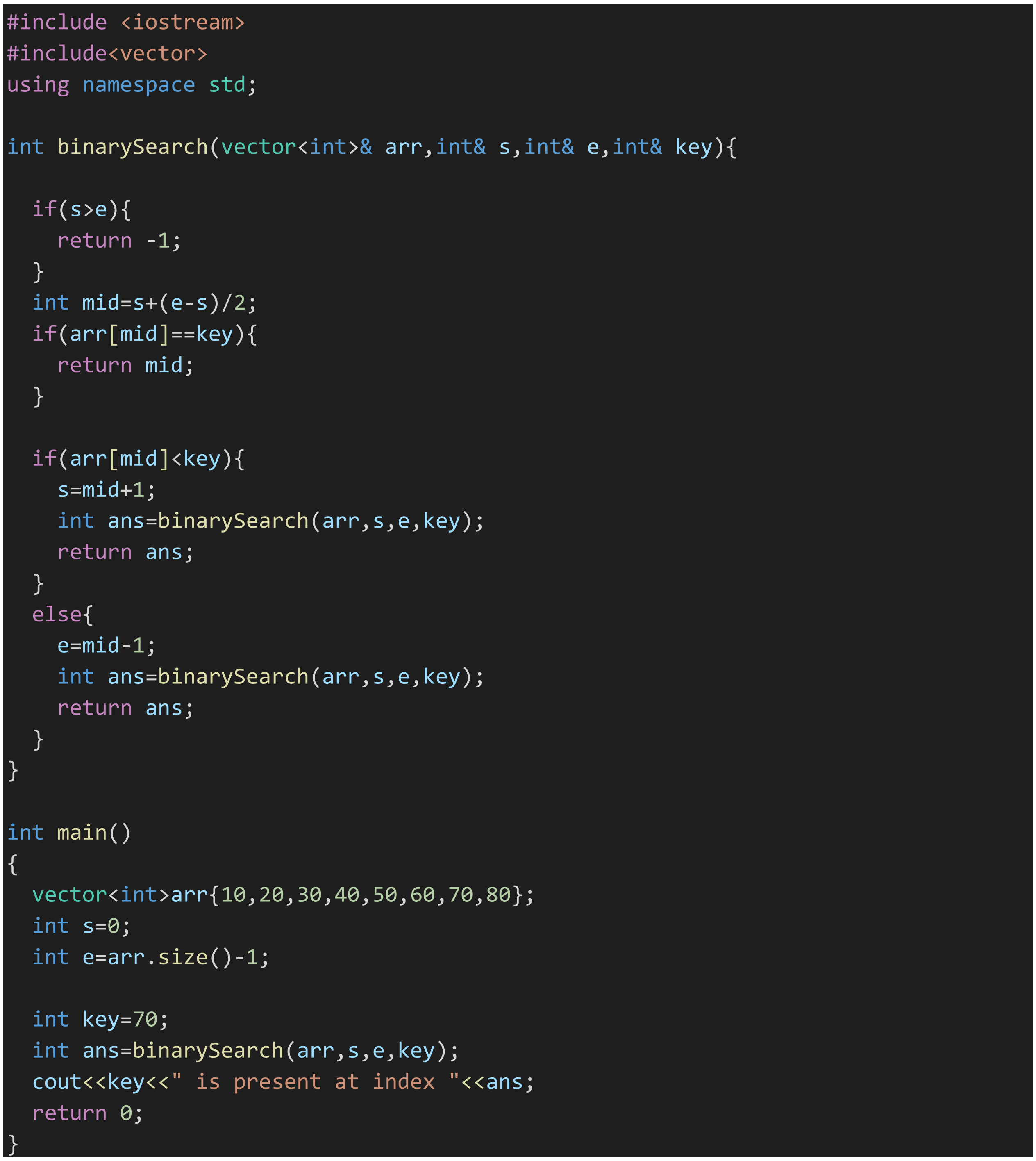
Q) using Binary Search with recursion -1 approach

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

int binarySearch(vector<int>arr,int s,int e,int key){

  if(s>e){
    return -1;
  }
  int mid=s+(e-s)/2;
  if(arr[mid]==key){
    return mid;
  }

  if(arr[mid]<key){
    int ans=binarySearch(arr,mid+1,e,key);
    return ans;
  }
  else{
    int ans=binarySearch(arr,s,mid-1,key);
    return ans;
  }
}

int main()
{
  vector<int>arr{10,20,30,40,50,60,70,80};
  int s=0;
  int e=arr.size()-1;

  int key=70;
  int ans=binarySearch(arr,s,e,key);
  cout<<key<<" is present at index "<<ans;
  return 0;
}
```

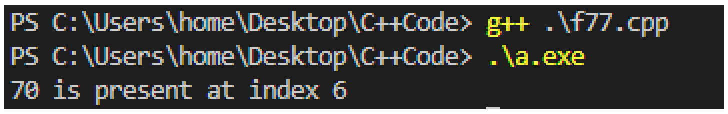


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
70 is present at index 6
```

Q) using Binary Search with recursion -2 approach



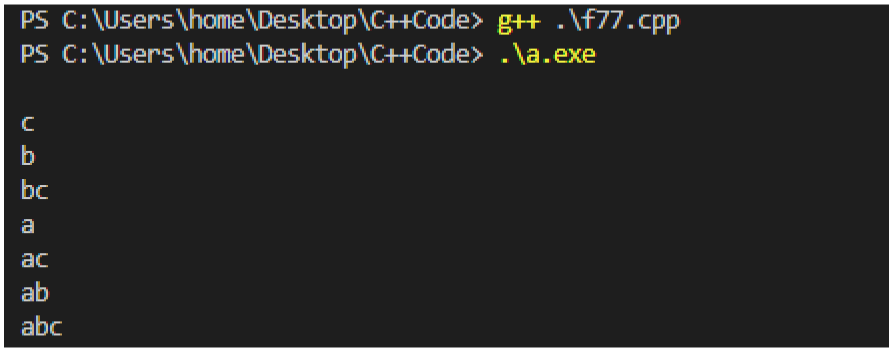
```cpp
#include <iostream>
#include<vector>
using namespace std;

int binarySearch(vector<int>& arr,int& s,int& e,int& key){

  if(s>e){
    return -1;
  }
  int mid=s+(e-s)/2;
  if(arr[mid]==key){
    return mid;
  }

  if(arr[mid]<key){
    s=mid+1;
    int ans=binarySearch(arr,s,e,key);
    return ans;
  }
  else{
    e=mid-1;
    int ans=binarySearch(arr,s,e,key);
    return ans;
  }
}

int main()
{
  vector<int>arr{10,20,30,40,50,60,70,80};
  int s=0;
  int e=arr.size()-1;

  int key=70;
  int ans=binarySearch(arr,s,e,key);
  cout<<key<<" is present at index "<<ans;
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
70 is present at index 6
```

Q) Print Subsequence(Adobe_include_Exclude)

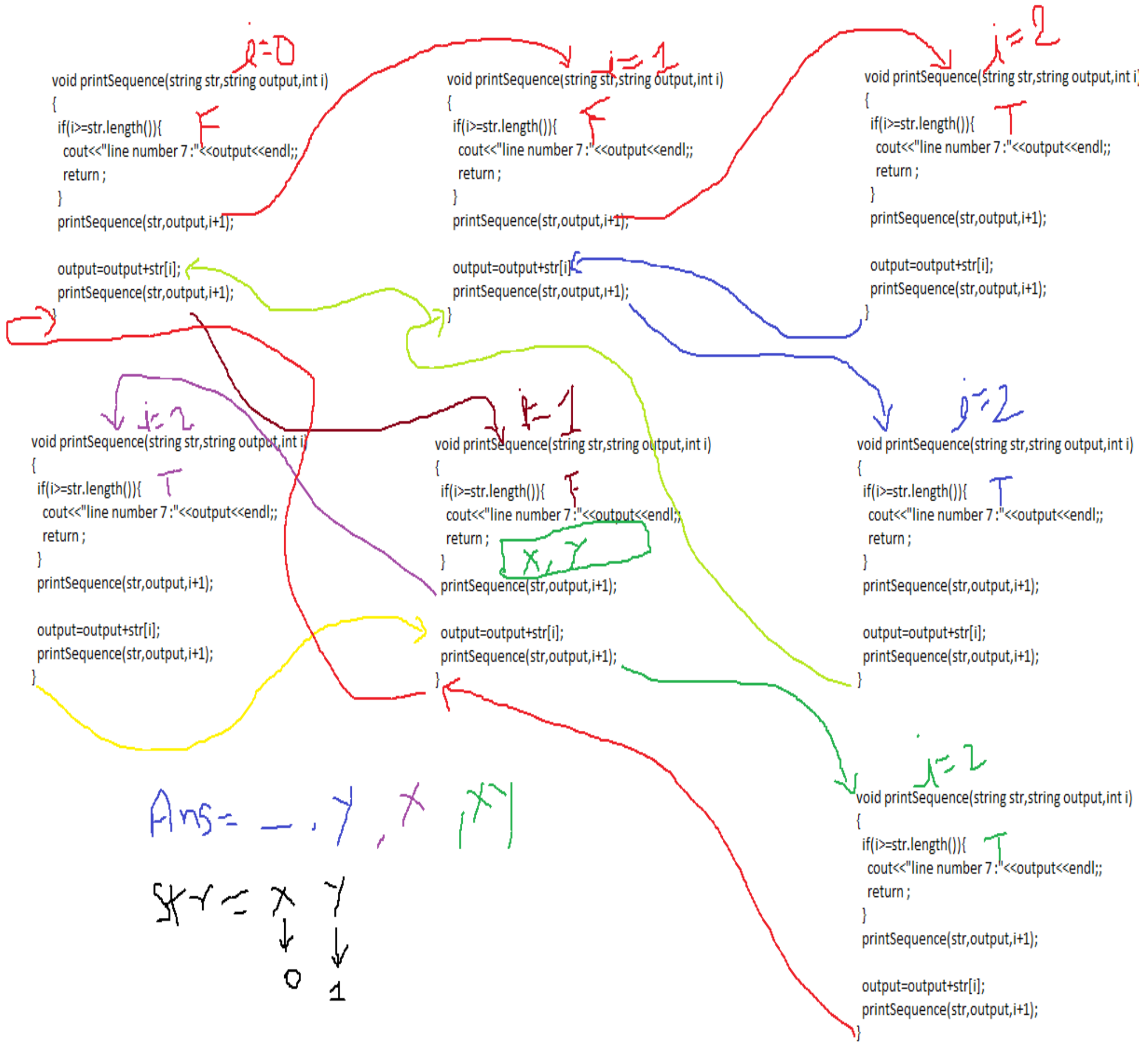
```cpp
#include <iostream>
#include<vector>
using namespace std;

void printSequence(string str,string output,int i){
  if(i>=str.length()){
    cout<<output<<endl;;
    return ;
  }
  printSequence(str,output,i+1);

  output=output+str[i];
  printSequence(str,output,i+1);
}

int main()
{
  string str="abc";
  string output="";
  int i=0;
  printSequence(str,output,i);
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe

c
b
bc
a
ac
ab
abc
```

Another approach :

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

void printSequence(string str,string output,int i,vector<string>& arr){
  if(i>=str.length()){
    cout<<"line number 7 :"<<output<<endl;;
    arr.push_back(output);
    return ;
  }
  printSequence(str,output,i+1,arr);

  cout<<"line number 13 :"<<str[i]<<endl;;
  output=output+str[i];
  cout<<"line number 14 :"<<output<<endl;;
  printSequence(str,output,i+1,arr);
}

int main()
{
  string str="xy";
  string output="";
  vector<string> arr;
  int i=0;
  printSequence(str,output,i,arr);
  //print all elements in vector
  cout<<"subSequence ";
  for(auto i :arr){
    cout<<i<<" ";
  }
  return 0;
}
```



```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

```
void printSequence(string str,string output,int i
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

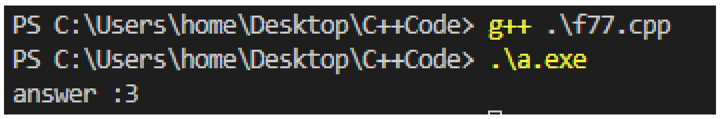
```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

```
void printSequence(string str,string output,int i)
{
 if(i>=str.length()){
  cout<<"line number 7 :"<<output<<endl;;
  return ;
 }
 printSequence(str,output,i+1);

 output=output+str[i];
 printSequence(str,output,i+1);
}
```

Recursion Level – 4

Q) Minimum numbers of elements required to reach sum

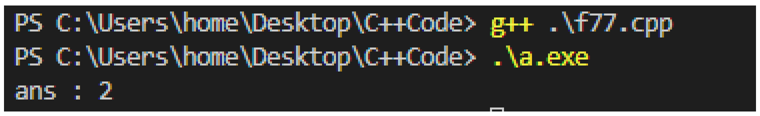
```
#include <iostream>
#include<vector>
#include<limits.h>

using namespace std;

int solve( vector<int>& arr,int target){
  if(target==0){
    return 0;
  }
  if(target<0){
    return INT_MAX;
  }

  int mini=INT_MAX;
  for(int i=0;i<arr.size();i++){
    int ans=solve(arr,target-arr[i]);

    if(ans!=INT_MAX){
      mini=min(mini,ans + 1);
    }
  }
    return mini;
}
int main()
{
  vector<int>arr{1,2};
  int target=5;
  int ans=solve(arr,target);
  cout<<"answer :"<<ans;
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
answer :3
```

Q) Rod cut into segments

```cpp
#include <iostream>
#include<vector>
#include<limits.h>

using namespace std;

int solve(int n,int x,int y,int z){
  if(n==0){
    return 0;
  }
  if(n<0){
    return INT_MIN;
  }
  int ans1=solve(n-x,  x,y,z)+1;
  int ans2=solve(n-y,x,y,z)+1;
  int ans3=solve(n-z,x,y,z)+1;

  int ans=max(ans1,max(ans2,ans3));
  return ans;
}
int main()
{
  int n=7;
  int x=5;
  int y=2;
  int z=2;
  int ans=solve(n,x,y,z);
  if(ans<0){
    ans=0;
  }
  cout<<"ans : "<<ans;
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
ans : 2
```

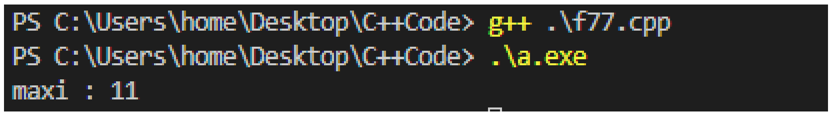
Q)Max sum of non-adjacent elements

Case:2,1,4,9 5 (output=11)

Case:1,2,5 (output= 5)

Case:1,2,3,5,4 (output= 8)

```cpp
#include <iostream>
#include<vector>
#include<limits.h>

using namespace std;
void solve(vector<int>& arr,int i,int sum,int& maxi){
  if(i>=arr.size()){
    maxi=max(sum,maxi);
    return ;
  }
  solve(arr,i+2,sum+arr[i],maxi);

   solve(arr,i+1,sum,maxi);
}
int main()
{
  vector<int>arr{2,1,4,9};
  int sum=0;
  int maxi=INT_MIN;
  int i=0;
  solve(arr,i,sum,maxi);
  cout<<"maxi : "<<maxi;
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
maxi : 11
```

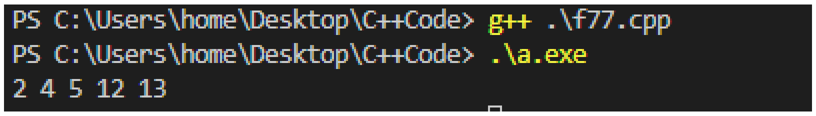
Divide & Conquer ,Time Complexity & Space Complexity

DnC Level -1

Q) MergeSort based on Divide & Conquer(Imortant Question)

```cpp
#include <iostream>
#include <vector>
#include <limits.h>

using namespace std;
void merge(int *arr, int s, int e)
{
  int mid = s + (e - s) / 2;

  int len1 = mid - s + 1;
  int len2 = e - mid;

  int *left = new int[len1];
  int *right = new int[len2];

  int k = s;
  for (int i = 0; i < len1; i++)
  {
    left[i] = arr[k];
    k++;
  }

  k = mid + 1;
  for (int i = 0; i < len2; i++)
  {
    right[i] = arr[k];
    k++;
  }
  int leftIndex = 0;
  int rightIndex = 0;
  int mainArrayIndex = s;

  while (leftIndex < len1 && rightIndex < len2)
  {
    if (left[leftIndex] < right[rightIndex])
    {
      arr[mainArrayIndex++] = left[leftIndex++];
    }
    else
    {
      arr[mainArrayIndex++] = right[rightIndex++];
    }
  }

  while (leftIndex < len1)
  {
    arr[mainArrayIndex++] = left[leftIndex++];
  }

  while (rightIndex < len2)
  {
    arr[mainArrayIndex++] = right[rightIndex++];
  }
}
void mergeSort(int *arr, int s, int e)
{

  if (s >= e)
  {
    return;
  }
  int mid = s + (e - s) / 2;
  mergeSort(arr, s, mid);
  mergeSort(arr, mid + 1, e);
  merge(arr, s, e);
}
```

```cpp
int main()
{
  int arr[] = {4, 5, 13, 2, 12};
  int n = 5;
  int s = 0;
  int e = n - 1;
  mergeSort(arr, s, e);
  for (int i = 0; i < n; i++)
  {
    cout << arr[i] << " ";
  }
  cout << endl;
  return 0;
}
```

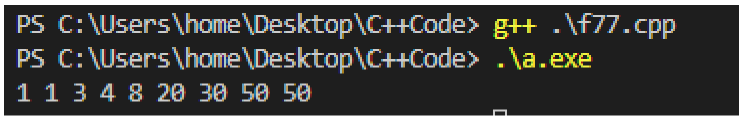


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
2 4 5 12 13
```

DnC Level -2

Q) Quick Sort (we will be given an array & we need to sort in the increasing order)

```
 #include <iostream>

#include <vector>
#include <limits.h>

using namespace std;

int partition(int arr[], int s, int e)
{
  int pivotIndex = s;
  int pivotElement = arr[s];

  int count = 0;
  for (int i = s + 1; i <= e; i++)
  {
    if (arr[i] <= pivotElement)
    {
      count++;
    }
  }

  int rightIndex = s + count;
  swap(arr[pivotIndex], arr[rightIndex]);
  pivotIndex = rightIndex;

  int i = s;
  int j = e;
  while (i < pivotIndex && j > pivotIndex)
  {
    while (arr[i] <= pivotElement)
    {
      i++;
    }
    while (arr[j] > pivotElement)
    {
      j--;
    }
  }
if (i < pivotIndex && j > pivotIndex)
  {
    swap(arr[i], arr[j]);
  }
  return pivotIndex;
}

void quickSort(int arr[], int s, int e)
{
  if (s >= e)
  {
    return;
  }

  int p = partition(arr, s, e);

  quickSort(arr, s, p - 1);
  quickSort(arr, p + 1, e);
}

int main()
{
  int arr[] = {8, 1,1, 3, 4, 20, 50,50, 30};
  int n = 9;
  int s = 0;
  int e = n - 1;
  quickSort(arr, s, e);

  for (int i = 0; i < n; i++)
  {
    cout << arr[i] << " ";
  }
  cout << endl;
  return 0;
}
```

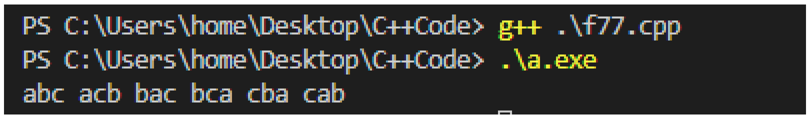


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
1 1 3 4 8 20 30 50 50
```

Backtracking

Q) permutation

```
#include <iostream>

using namespace std;
void printPermutation(string &str,int i){
  if(i>=str.length()){
    cout<<str<<" ";
    return ;
  }

  for(int j=i;j<str.length();j++){
    swap(str[i],str[j]);

    printPermutation(str,i+1);
    swap(str[i],str[j]);
  }
}
int main()
{
  string str="abc";
  int i=0;
  printPermutation(str,i);
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
abc acb bac bca cba cab
```

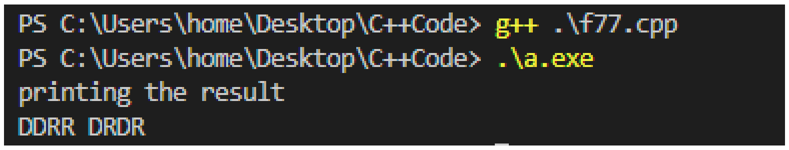
DnC Level – 3

Rat In Maze

```cpp
#include <iostream>
#include<vector>

using namespace std;

bool isSafe(int i,int j,int row,int col,int arr[][3],vector <vector<bool> >
&visited)
{
  if( ((i>=0 && i<row) && (j>=0 && j<col)) && (arr[i][j]==1) &&
(visited[i][j]==false)){
    return true;
  }
  else{
    return false;
  }
}

void solveMaze(int arr[3][3],int row,int col,int i,int j,
                vector <vector<bool> > &visited,vector<string> &path,string
output)
{

if(i==row-1 && j==col-1){
  path.push_back(output);
  return;
}

//down i+1,j
if(isSafe(i+1,j,row,col,arr,visited)){
  visited[i+1][j]=true;
  solveMaze(arr,row,col,i+1,j,visited,path,output + 'D');
  visited[i+1][j]=false;
}
//left i,j-1
if(isSafe(i,j-1,row,col,arr,visited)){
  visited[i][j-1]=true;
  solveMaze(arr,row,col,i,j-1,visited,path,output + 'L');
  visited[i][j-1]=false;
}
```

```
//right i,j+1
if(isSafe(i,j+1,row,col,arr,visited)){
  visited[i][j+1]=true;
  solveMaze(arr,row,col,i,j+1,visited,path,output + 'R');
  visited[i][j+1]=false;
}

//up i-1,j
if(isSafe(i-1,j,row,col,arr,visited)){
  visited[i-1][j]=true;
  solveMaze(arr,row,col,i-1,j,visited,path,output + 'U');
  visited[i-1][j]=false;
}

}
int main()
{
 int maze[3][3]={
                  {1,0,0},
                  {1,1,0},
                  {1,1,1}
                };
  //check first position
  if(maze[0][0]==0){
    cout<<"No path Exists "<<endl;
    return 0;
  }
  int row=3;
  int col=3;

  vector <vector<bool> >visited(row,vector<bool>(col,false));

  visited[0][0]=true;

  vector<string> path;
  string output="";

  solveMaze(maze,row,col,0,0,visited,path,output);
  cout<<"printing the result "<<endl;
  for(auto i:path){
    cout<<i<<" ";
  }
  cout<<endl;
```

```
if(path.size()==0){
    cout<<"No path exists "<<endl;
  }
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
printing the result
DDRR DRDR
```

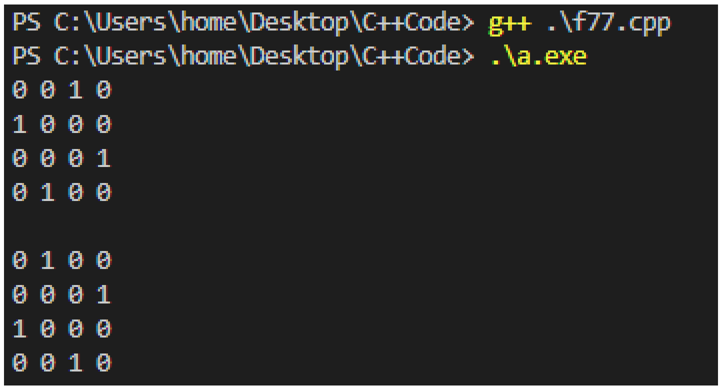
DnC Level – 4

Q) Place Queen (Important Question)

```
#include <iostream>
#include <vector>

using namespace std;

void printSolution(vector<vector<int>> &board, int n)
{
  for (int i = 0; i < n; i++)
  {
    for (int j = 0; j < n; j++)
    {
      cout << board[i][j] << " ";
    }
    cout << endl;
  }
  cout<<endl<<endl;
}

bool isSafe(int row, int col, vector<vector<int>> &board, int n)
{

  int i = row;
  int j = col;

  // check row
  while (j >= 0)
  {
    if (board[i][j] == 1)
    {
      return false;
    }
    j--;
  }
```

```
 // check upper left digonal
  i = row;
  j = col;

  while (i >= 0 && j >= 0)
  {
    if (board[i][j] == 1)
    {
      return false;
    }
    i--;
    j--;
  }

  // check bottom left digonal
  i = row;
  j = col;

  while (i < n && j >= 0)
  {
    if (board[i][j] == 1)
    {
      return false;
    }
    i++;
    j--;
  }
  return true;
}
```

```cpp
void solve(vector<vector<int>> &board, int col, int n)
{
  if (col >= n)
  {
    printSolution(board, n);
    return;
  }

  for (int row = 0; row < n; row++)
  {
    if (isSafe(row, col, board, n))
    {
      board[row][col] = 1;

      solve(board, col+1, n);

      board[row][col] = 0;
    }
  }
}

int main()
{
  int n = 4;
  vector<vector<int>> board(n, vector<int>(n, 0));
  int col = 0;
  solve(board, col, n);
  return 0;
}
```

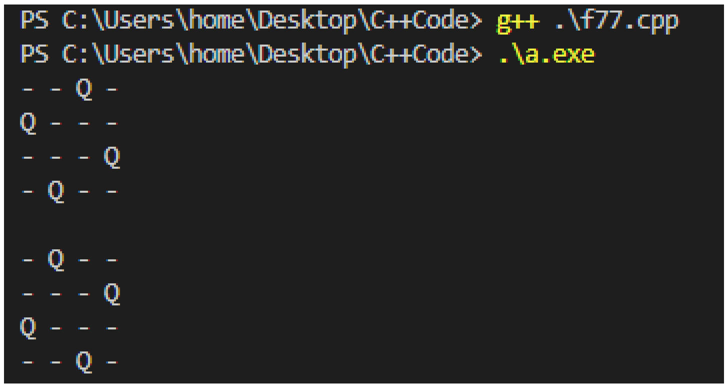


```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
0 0 1 0
1 0 0 0
0 0 0 1
0 1 0 0

0 1 0 0
0 0 0 1
1 0 0 0
0 0 1 0
```

2nd Apprach

```cpp
#include <iostream>
#include <vector>

using namespace std;

void printSolution(vector<vector<char>> &board, int n)
{
  for (int i = 0; i < n; i++)
  {
    for (int j = 0; j < n; j++)
    {
      cout << board[i][j] << " ";
    }
    cout << endl;
  }
  cout<<endl;
}

bool isSafe(int row, int col, vector<vector<char>> &board, int n)
{

  int i = row;
  int j = col;

  // check row
  while (j >= 0)
  {
    if (board[i][j] == 'Q')
    {
      return false;
    }
    j--;
  }
// check upper left digonal
  i = row;
  j = col;

 while (i >= 0 && j >= 0)
  {
    if (board[i][j] == 'Q')
    {
      return false;
    }
    i--;
    j--;
  }

  // check bottom left digonal
  i = row;
  j = col;

  while (i < n && j >= 0)
  {
    if (board[i][j] == 'Q')
    {
      return false;
    }
    i++;
    j--;
  }
  return true;
}
```

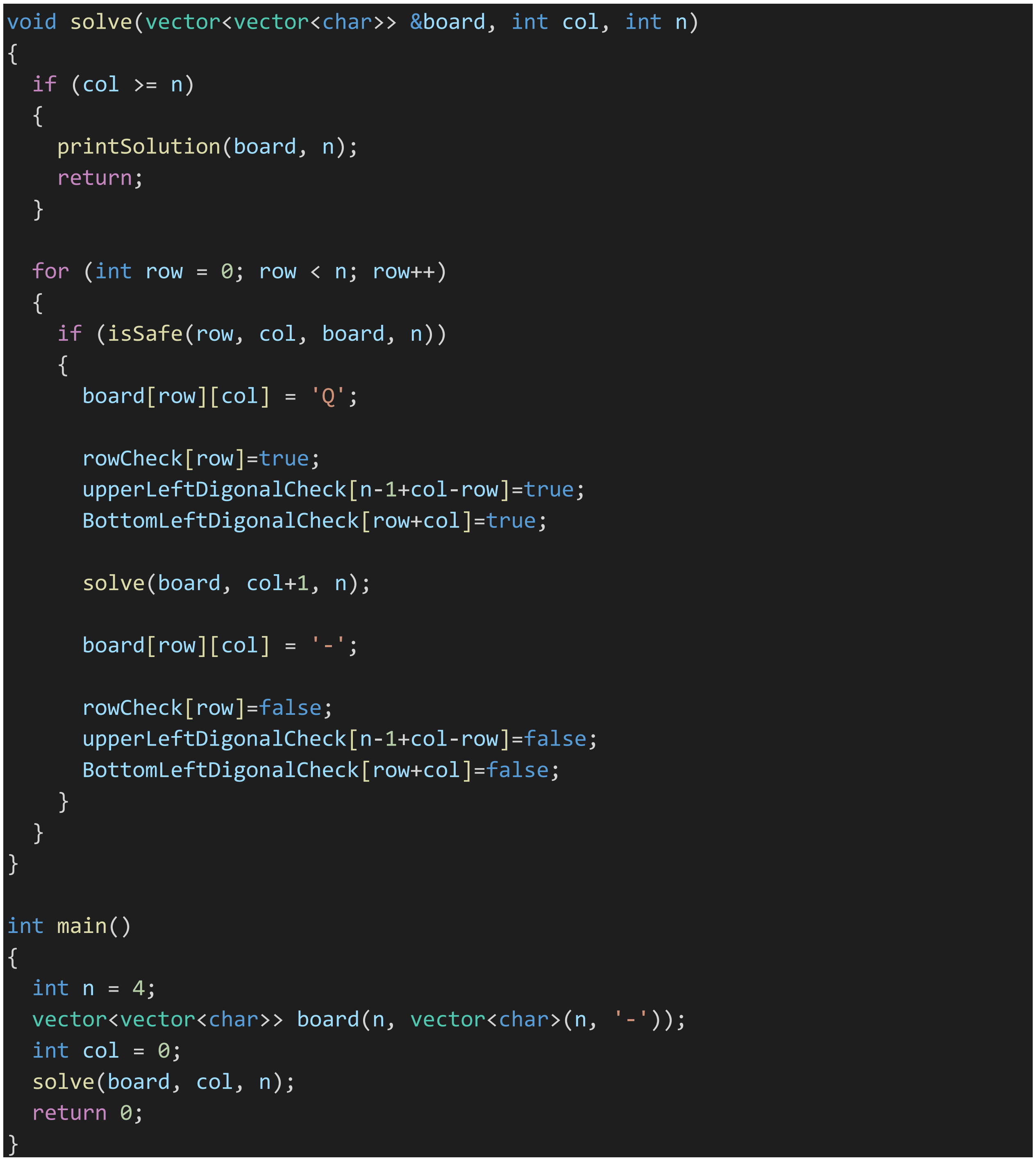
```
void solve(vector<vector<char>> &board, int col, int n)
{
  if (col >= n)
  {
    printSolution(board, n);
    return;
  }

  for (int row = 0; row < n; row++)
  {
    if (isSafe(row, col, board, n))
    {
      board[row][col] = 'Q';

      solve(board, col+1, n);

      board[row][col] = '-';
    }
  }
}

int main()
{
  int n = 4;
  vector<vector<char>> board(n, vector<char>(n, '-'));
  int col = 0;
  solve(board, col, n);
  return 0;
}
```



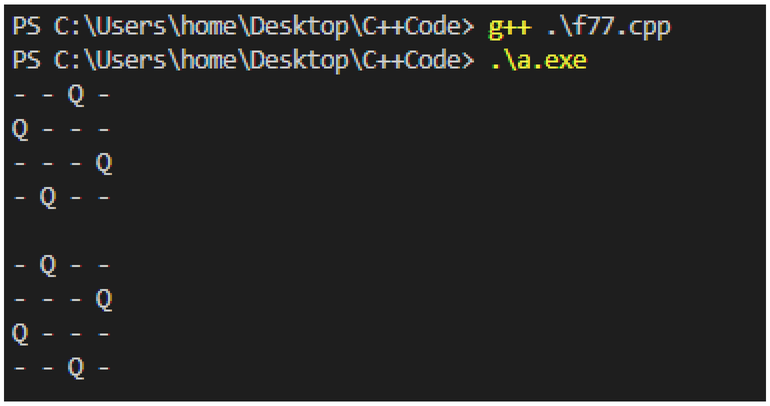
```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
- - Q -
Q - - -
- - - Q
- Q - -

- Q - -
- - - Q
Q - - -
- - Q -
```

3rd Approach with unordered_map

```cpp
#include <iostream>
#include <vector>
#include<unordered_map>

using namespace std;

unordered_map<int,bool>rowCheck;
unordered_map<int,bool>upperLeftDigonalCheck;
unordered_map<int,bool>BottomLeftDigonalCheck;

void printSolution(vector<vector<char>> &board, int n)
{
  for (int i = 0; i < n; i++)
  {
    for (int j = 0; j < n; j++)
    {
      cout << board[i][j] << " ";
    }
    cout << endl;
  }
  cout<<endl;
}

bool isSafe(int row, int col, vector<vector<char>> &board, int n)
{
  if(rowCheck[row]==true)
    return false;

  if(upperLeftDigonalCheck[n-1+col-row]==true)
    return false;

  if(BottomLeftDigonalCheck[row+col]==true)
    return false;

  return true;
}
```



```cpp
void solve(vector<vector<char>> &board, int col, int n)
{
  if (col >= n)
  {
    printSolution(board, n);
    return;
  }

  for (int row = 0; row < n; row++)
  {
    if (isSafe(row, col, board, n))
    {
      board[row][col] = 'Q';

      rowCheck[row]=true;
      upperLeftDigonalCheck[n-1+col-row]=true;
      BottomLeftDigonalCheck[row+col]=true;

      solve(board, col+1, n);

      board[row][col] = '-';

      rowCheck[row]=false;
      upperLeftDigonalCheck[n-1+col-row]=false;
      BottomLeftDigonalCheck[row+col]=false;
    }
  }
}

int main()
{
  int n = 4;
  vector<vector<char>> board(n, vector<char>(n, '-'));
  int col = 0;
  solve(board, col, n);
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
- - Q -
Q - - -
- - - Q
- Q - -

- Q - -
- - - Q
Q - - -
- - Q -
```

DnC Level – 5

22. Generate Parentheses(LeetCode)_ImportantQuestion

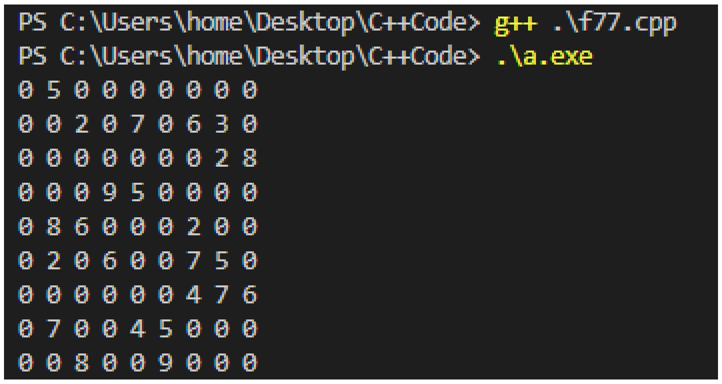
17. Letter Combinations of a Phone Number(LeetCode)_ImportantQuestion_deshaw company

DnC Level – 6

Q)Sudoku Solver(wrong answer aa raha hai)

```
#include <iostream>
#include <vector>
#include<unordered_map>

using namespace std;

bool isSafe(int current_row,int current_col,vector<vector<char>> board,int
value){
  int n=9;
  for(int i=0;i<n;i++){
    if(board[current_row][i]==value){
      return false;
    }

    if(board[i][current_col]==value){
      return false;
    }

    if(board[3*(current_row/3)+i/3][3*(current_col/3)+(i%3)]){
      return false;
    }
  }
  return true;
}
```

```cpp
bool solve(vector<vector<char>> &board,int n){
  for(int i=0;i<n;i++){
    for(int j=0;j<n;j++){
      if(board[i][j]='0'){
        for(char value='1';value<='9';value++){
          if(isSafe(i,j,board,value)){
            board[i][j]=value;
            bool remainingSol=solve(board,n);
            if(remainingSol==true){
              return true;
            }
          }
          else{
            board[i][j]='0';
          }
        }

      }
       return false;
    }
  }
  return true;
}
int main()
{
  vector<vector<char>> board(9);
  // int board[9][9]={
  //   {4,5,0,0,0,0,0,0,0},
  //   {0,0,2,0,7,0,6,3,0},
  //   {'0','0','0','0','0','0','0','2','8'};
  //   {0,0,0,9,5,0,0,0,0},
  //   {0,8,6,0,0,0,2,0,0},
  //   {0,2,0,6,0,0,7,5,0},
  //   {0,0,0,0,0,0,4,7,6},
  //   {0,7,0,0,4,5,0,0,0},
  //   {0,0,8,0,0,9,0,0,0}
  // };
  board[0]={'4','5','0','0','0','0','0','0','0'};
  board[1]={'0','0','2','0','7','0','6','3','0'};
  board[2]={'0','0','0','0','0','0','0','2','8'};
  board[3]={'0','0','0','9','5','0','0','0','0'};
  board[4]={'0','8','6','0','0','0','2','0','0'};
  board[5]={'0','2','0','6','0','0','7','5','0'};
  board[6]={'0','0','0','0','0','0','4','7','6'};
  board[7]={'0','7','0','0','4','5','0','0','0'};
  board[8]={'0','0','8','0','0','9','0','0','0'};
  int n=9;
  solve(board,n);
  //printing
  for(int i=0;i<n;i++){
    for(int j=0;j<n;j++){
      cout<<board[i][j]<<" ";
    }
    cout<<endl;
  }
  return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
0 5 0 0 0 0 0 0 0
0 0 2 0 7 0 6 3 0
0 0 0 0 0 0 0 2 8
0 0 0 9 5 0 0 0 0
0 8 6 0 0 0 2 0 0
0 2 0 6 0 0 7 5 0
0 0 0 0 0 0 4 7 6
0 7 0 0 4 5 0 0 0
0 0 8 0 0 9 0 0 0
```

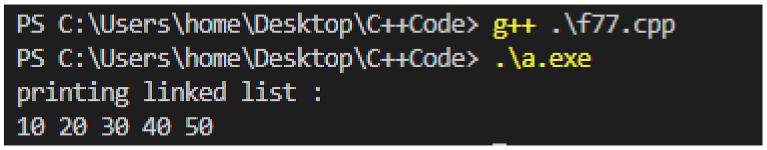
37. Sudoku Solver(LeetCode) _ImportantQuestion

Linked Lists

LL Class-1

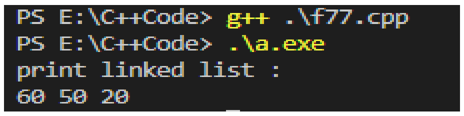
```
#include <iostream>

using namespace std;


class Node{
    public:
    int data ;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }
    Node(int data){
        this->data=data;
        this->next=NULL;
    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data <<" ";
       temp=temp->next;
    }

}
```

```
int main()
{
    Node* first=new Node(10);
    Node* second=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);

    first->next=second;
    second->next=third;
    third->next=fourth;
    fourth->next=fifth;

    cout<<"printing linked list :"<<endl;
    print(first);
    return 0;
}
```



```
PS C:\Users\home\Desktop\C++Code> g++ .\f77.cpp
PS C:\Users\home\Desktop\C++Code> .\a.exe
printing linked list :
10 20 30 40 50
```

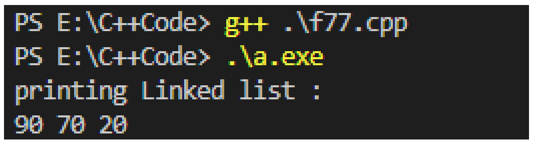
```cpp
#include<iostream>
using namespace std;

class Node{
    public:
    int data;
    Node* next;
    Node(){
        this->data=0;
        this->next=NULL;
    }
    Node(int data){
        this->data=data;
        this->next=NULL;
    }
};

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
}

void insertAtHead(Node* &head,int data){
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
}

int main (){

    Node* head=NULL;
    insertAtHead(head,20);
    insertAtHead(head,50);
    insertAtHead(head,60);
    cout<<"print linked list :"<<endl;
    print(head);
   return 0;
}
```



```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
print linked list :
60 50 20
```

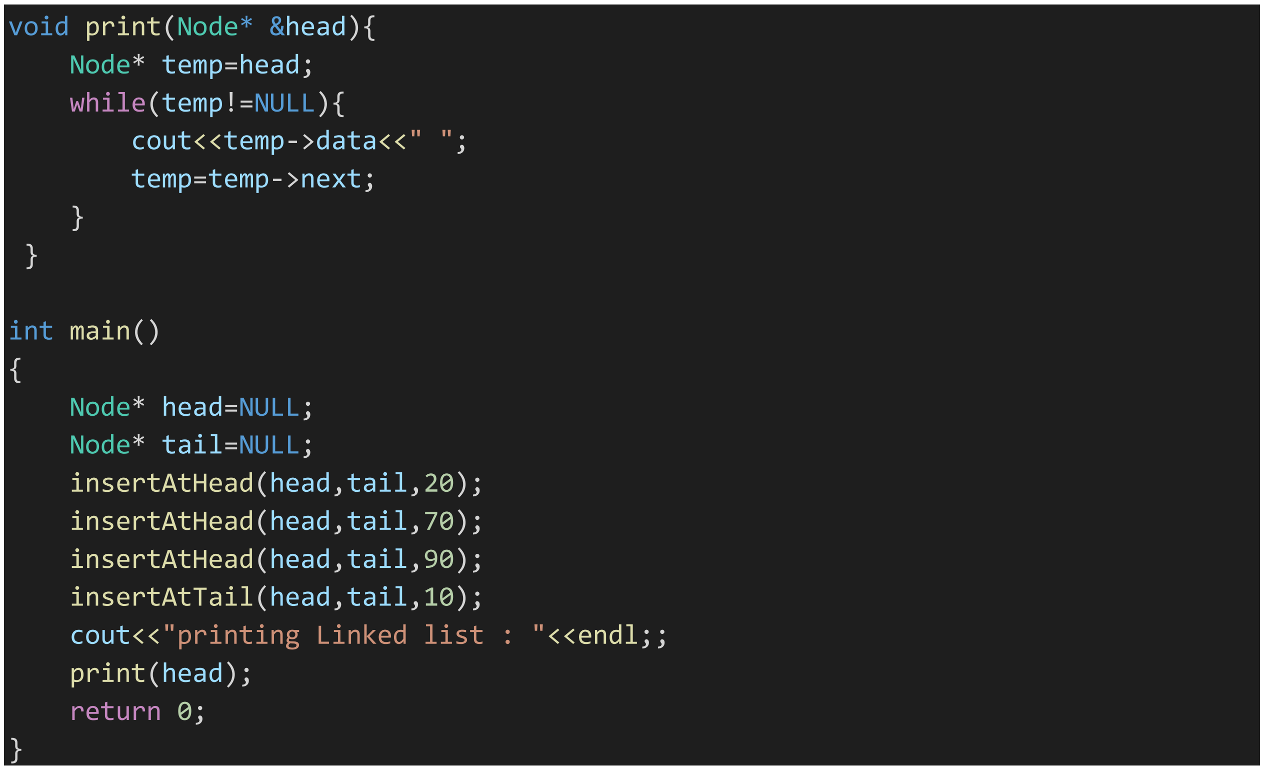
```cpp
 #include <iostream>


using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }

};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
 }
```

```cpp
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    return 0;
}
```

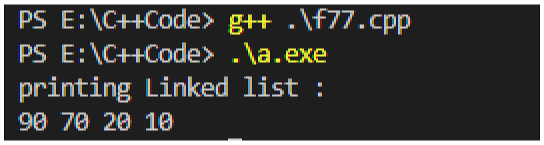


```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20
```

```cpp
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }

};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}
```



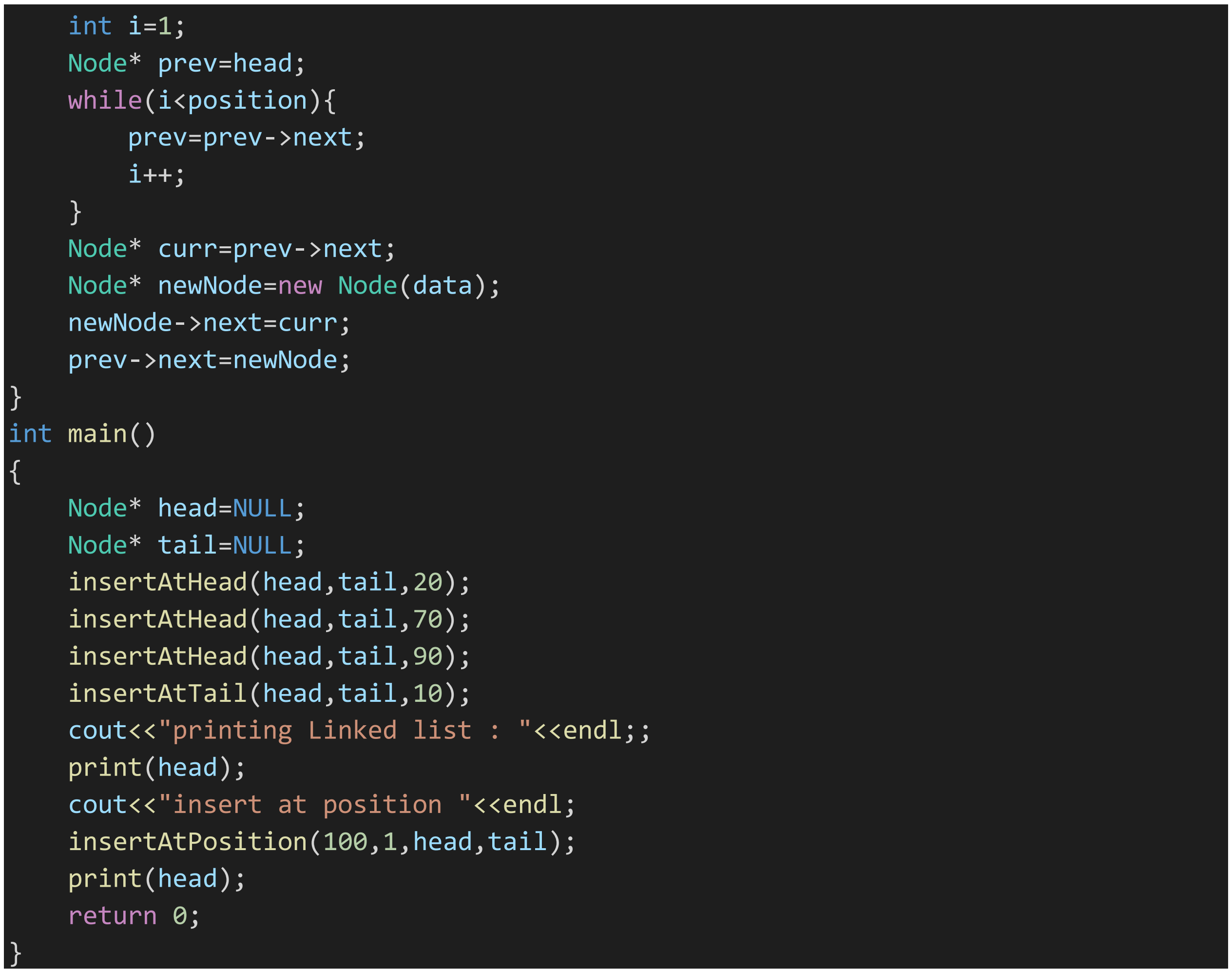
```
void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
 }

int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    return 0;
}
```



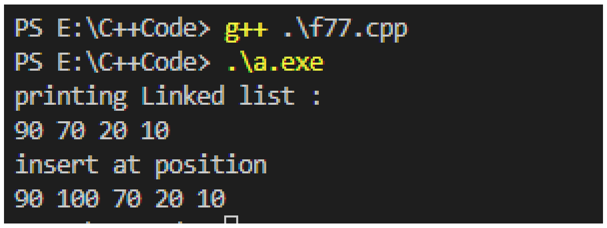
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20 10
```

Important

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }

};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}
void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    cout<<endl;
 }
int findLen(Node* &head){
     Node* temp=head;
     int i=0;
     while (temp!=NULL)
     {
       temp=temp->next;
       i++;
     }
     return i;

}
void insertAtPosition(int data,int position,Node* &head,Node* &tail){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==0){
        insertAtHead(head,tail,data);
        return;
    }

    int len=findLen(head);
    if(position>=len){
        insertAtTail(head,tail,data);
        return;
    }
```



```cpp
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    Node* newNode=new Node(data);
    newNode->next=curr;
    prev->next=newNode;
}
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    cout<<"insert at position "<<endl;
    insertAtPosition(100,1,head,tail);
    print(head);
    return 0;
}
```



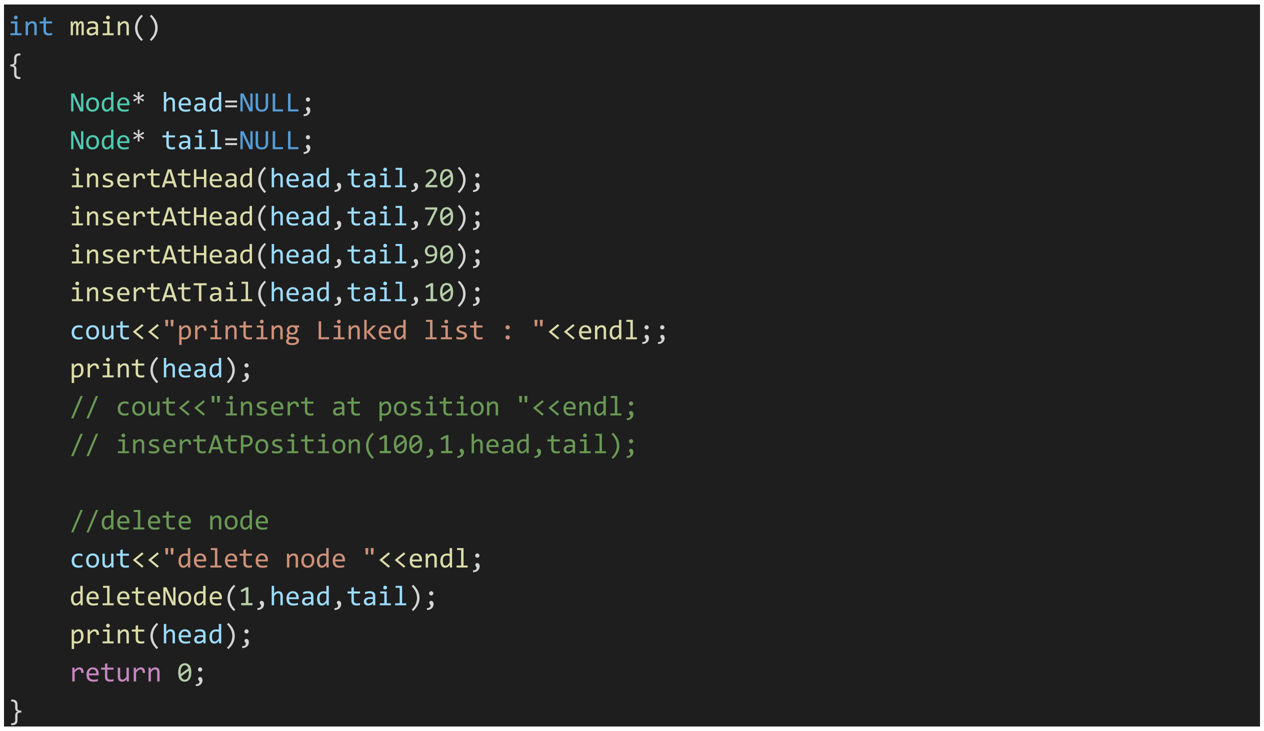
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20 10
insert at position
90 100 70 20 10
```

Delete Node

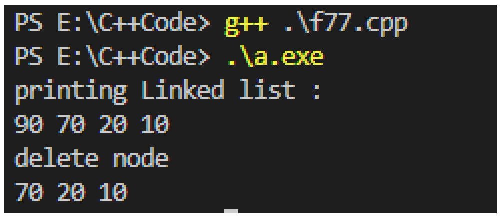
```cpp
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
```

```cpp
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    cout<<endl;
 }

int findLen(Node* &head){
     Node* temp=head;
     int i=0;
     while (temp!=NULL)
     {
       temp=temp->next;
       i++;
     }
     return i;

}
```

```cpp
void insertAtPosition(int data,int position,Node* &head,Node* &tail){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==0){
        insertAtHead(head,tail,data);
        return;
    }

    int len=findLen(head);
    if(position>=len){
        insertAtTail(head,tail,data);
        return;
    }
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    Node* newNode=new Node(data);
    newNode->next=curr;
    prev->next=newNode;
}

void deleteNode(int position,Node* &head,Node* &tail){
    if(head==NULL){
        cout<<"can not delete empty linked list "<<endl;
    }
    if(position==1){
        Node* temp=head;
        head=head->next;
        temp->next=NULL;
        delete temp;
    }
}
```



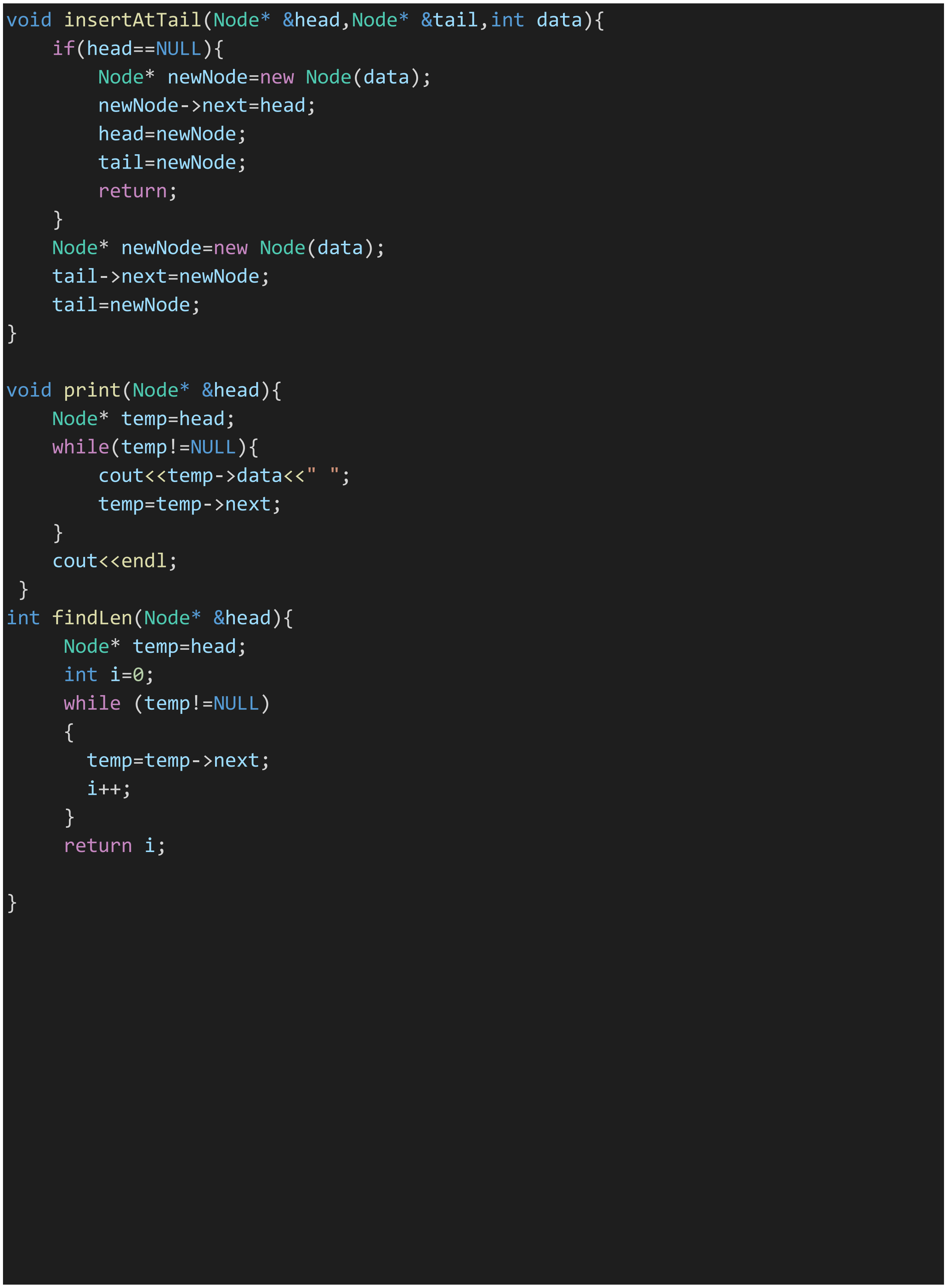
```cpp
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    // cout<<"insert at position "<<endl;
    // insertAtPosition(100,1,head,tail);

    //delete node
    cout<<"delete node "<<endl;
    deleteNode(1,head,tail);
    print(head);
    return 0;
}
```



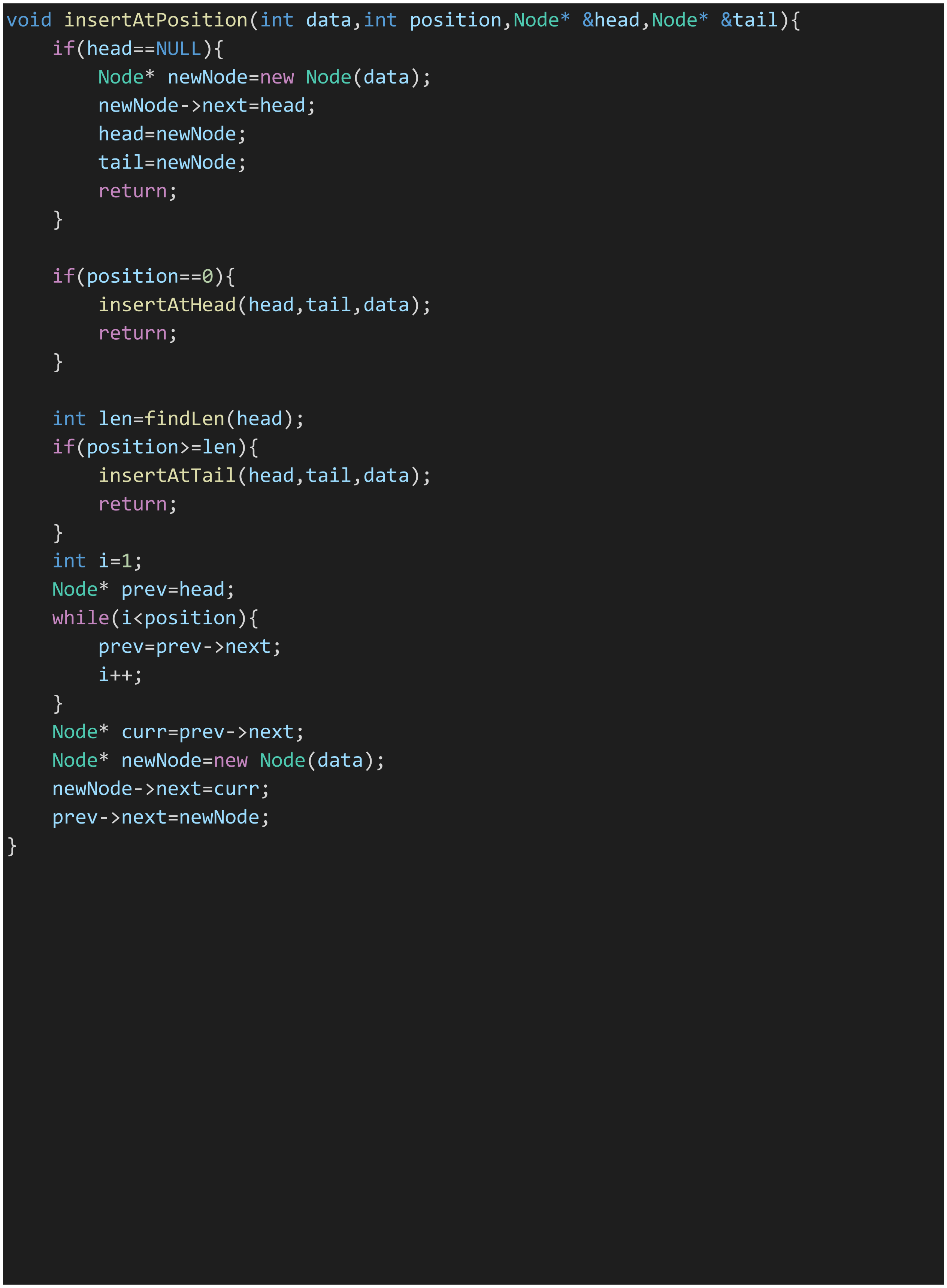
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20 10
delete node
70 20 10
```

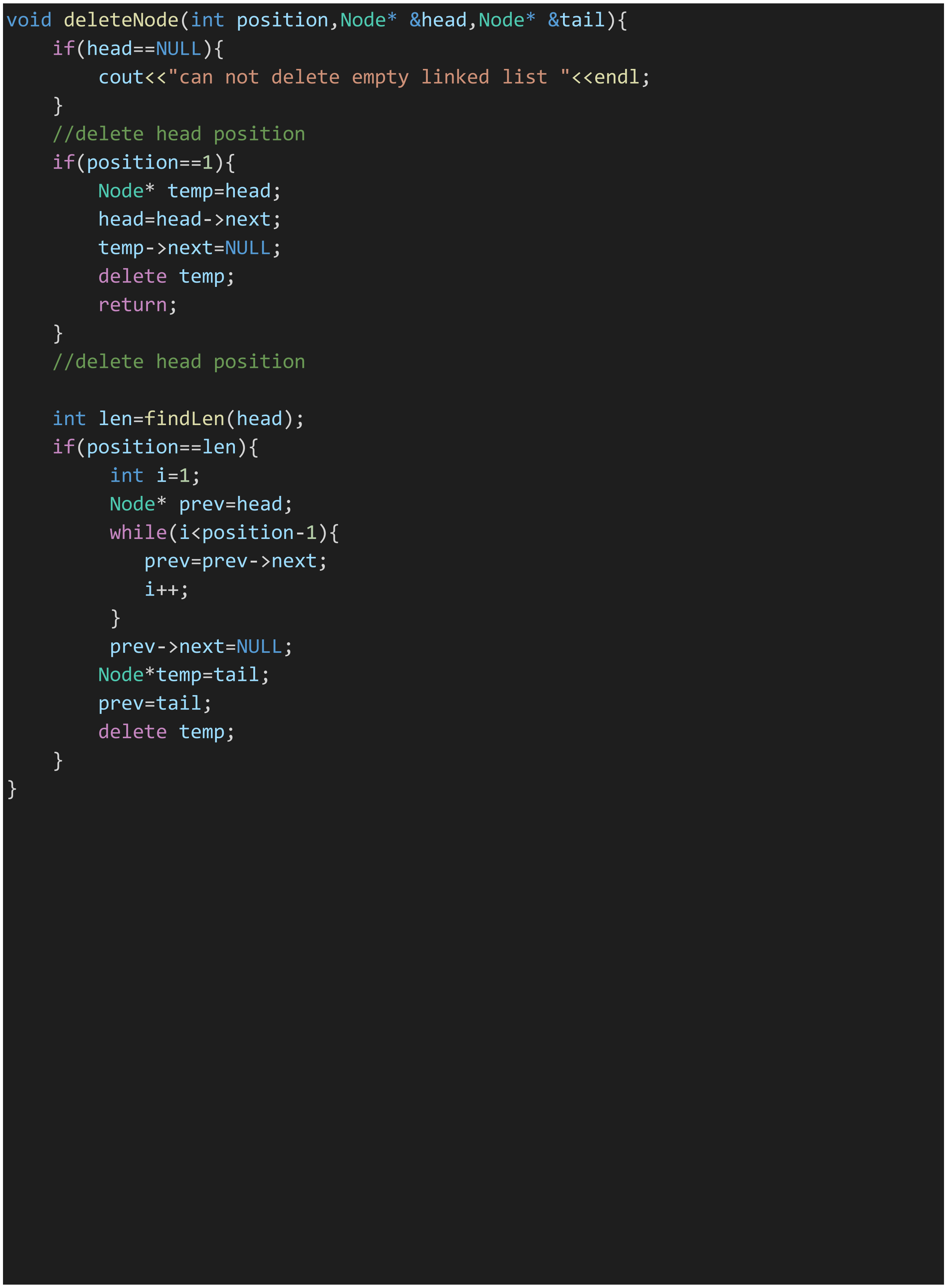
Delete tail Node

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
```



```cpp
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    cout<<endl;
 }
int findLen(Node* &head){
     Node* temp=head;
     int i=0;
     while (temp!=NULL)
     {
       temp=temp->next;
       i++;
     }
     return i;

}
```


```
void insertAtPosition(int data,int position,Node* &head,Node* &tail){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==0){
        insertAtHead(head,tail,data);
        return;
    }

    int len=findLen(head);
    if(position>=len){
        insertAtTail(head,tail,data);
        return;
    }
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    Node* newNode=new Node(data);
    newNode->next=curr;
    prev->next=newNode;
}
```



```cpp
void deleteNode(int position,Node* &head,Node* &tail){
    if(head==NULL){
        cout<<"can not delete empty linked list "<<endl;
    }
    //delete head position
    if(position==1){
        Node* temp=head;
        head=head->next;
        temp->next=NULL;
        delete temp;
        return;
    }
    //delete head position

    int len=findLen(head);
    if(position==len){
         int i=1;
         Node* prev=head;
         while(i<position-1){
            prev=prev->next;
            i++;
         }
         prev->next=NULL;
        Node*temp=tail;
        prev=tail;
        delete temp;
    }
}
```

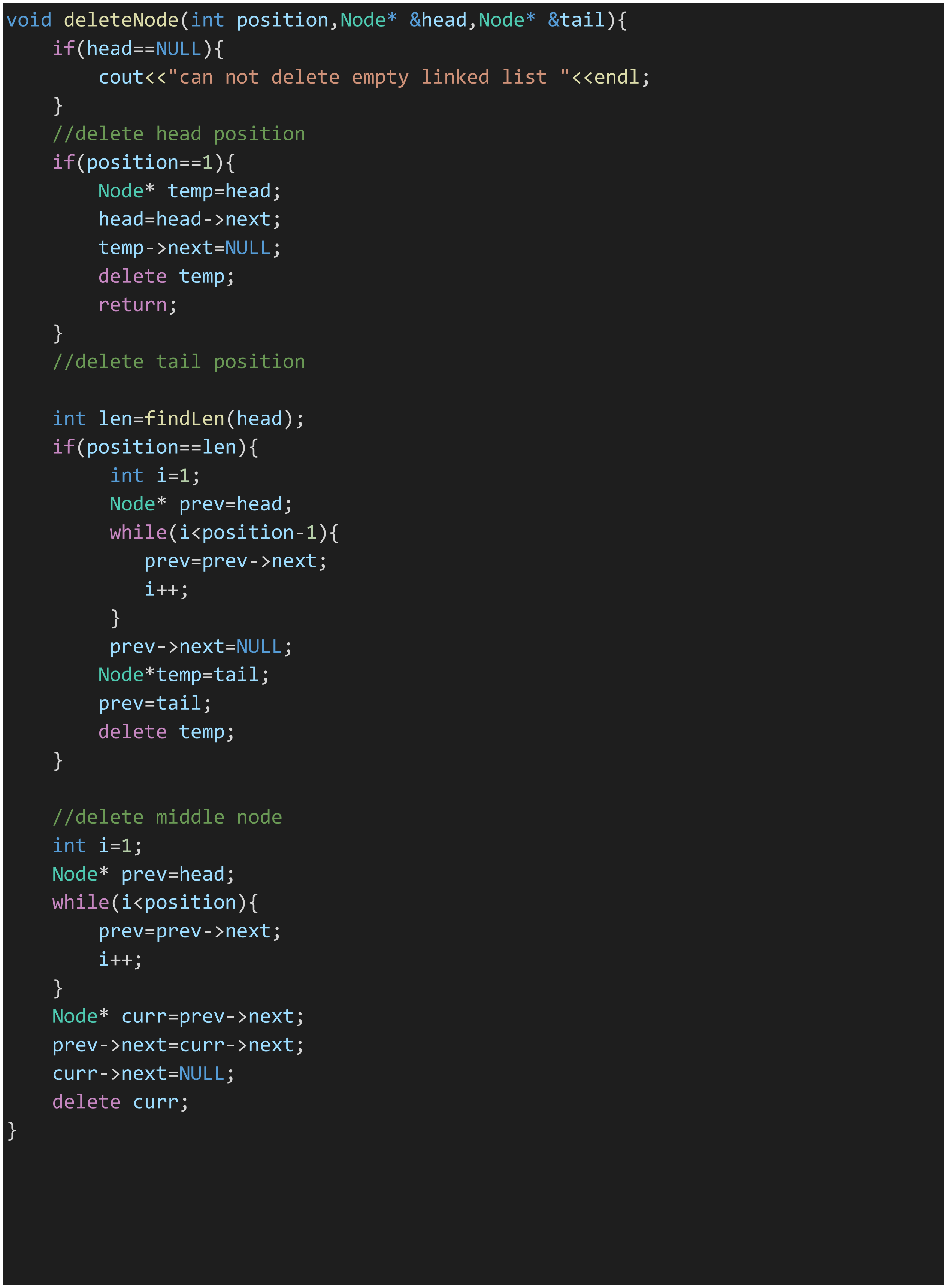
```cpp
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    // cout<<"insert at position "<<endl;
    // insertAtPosition(100,1,head,tail);

    //delete node
    cout<<"delete node "<<endl;
    deleteNode(4,head,tail);
    print(head);
    return 0;
}
```

Deleting middle node

```cpp
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
```

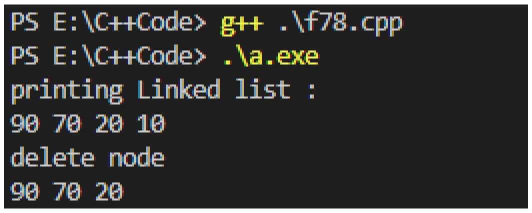
```
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    cout<<endl;
 }

int findLen(Node* &head){
     Node* temp=head;
     int i=0;
     while (temp!=NULL)
     {
       temp=temp->next;
       i++;
     }
     return i;

}
```

```
void insertAtPosition(int data,int position,Node* &head,Node* &tail){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==0){
        insertAtHead(head,tail,data);
        return;
    }

    int len=findLen(head);
    if(position>=len){
        insertAtTail(head,tail,data);
        return;
    }
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    Node* newNode=new Node(data);
    newNode->next=curr;
    prev->next=newNode;
}
```



```cpp
void deleteNode(int position,Node* &head,Node* &tail){
    if(head==NULL){
        cout<<"can not delete empty linked list "<<endl;
    }
    //delete head position
    if(position==1){
        Node* temp=head;
        head=head->next;
        temp->next=NULL;
        delete temp;
        return;
    }
    //delete tail position

    int len=findLen(head);
    if(position==len){
         int i=1;
         Node* prev=head;
         while(i<position-1){
            prev=prev->next;
            i++;
         }
         prev->next=NULL;
        Node*temp=tail;
        prev=tail;
        delete temp;
    }

    //delete middle node
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    prev->next=curr->next;
    curr->next=NULL;
    delete curr;
}
```

```cpp
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);
    cout<<"printing Linked list : "<<endl;;
    print(head);
    // cout<<"insert at position "<<endl;n-
    // insertAtPosition(100,1,head,tail);

    //delete node
    cout<<"delete node "<<endl;
    deleteNode(3,head,tail);
    print(head);
    return 0;
}
```



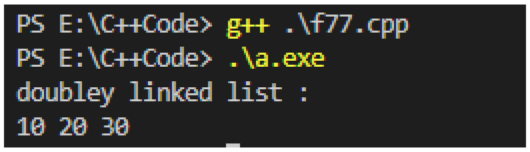
```
PS E:\C++Code> g++ .\f78.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20 10
delete node
90 70 20
```

LL Class-2

Double linked list

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;
    Node* prev;

    Node(){
        this->data=0;
        this->next=NULL;
        this->prev=NULL;
    }
     Node(int data){
        this->data=data;
        this->next=NULL;
        this->prev=NULL;
    }
};
void print (Node* &head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data<<" ";
       temp=temp->next;
    }

}
int getLength(Node* head){
    int length=0;
    Node* temp=head;
    while(temp!=NULL){
        temp=temp->next;
        length++;
    }
    return length;
}
```

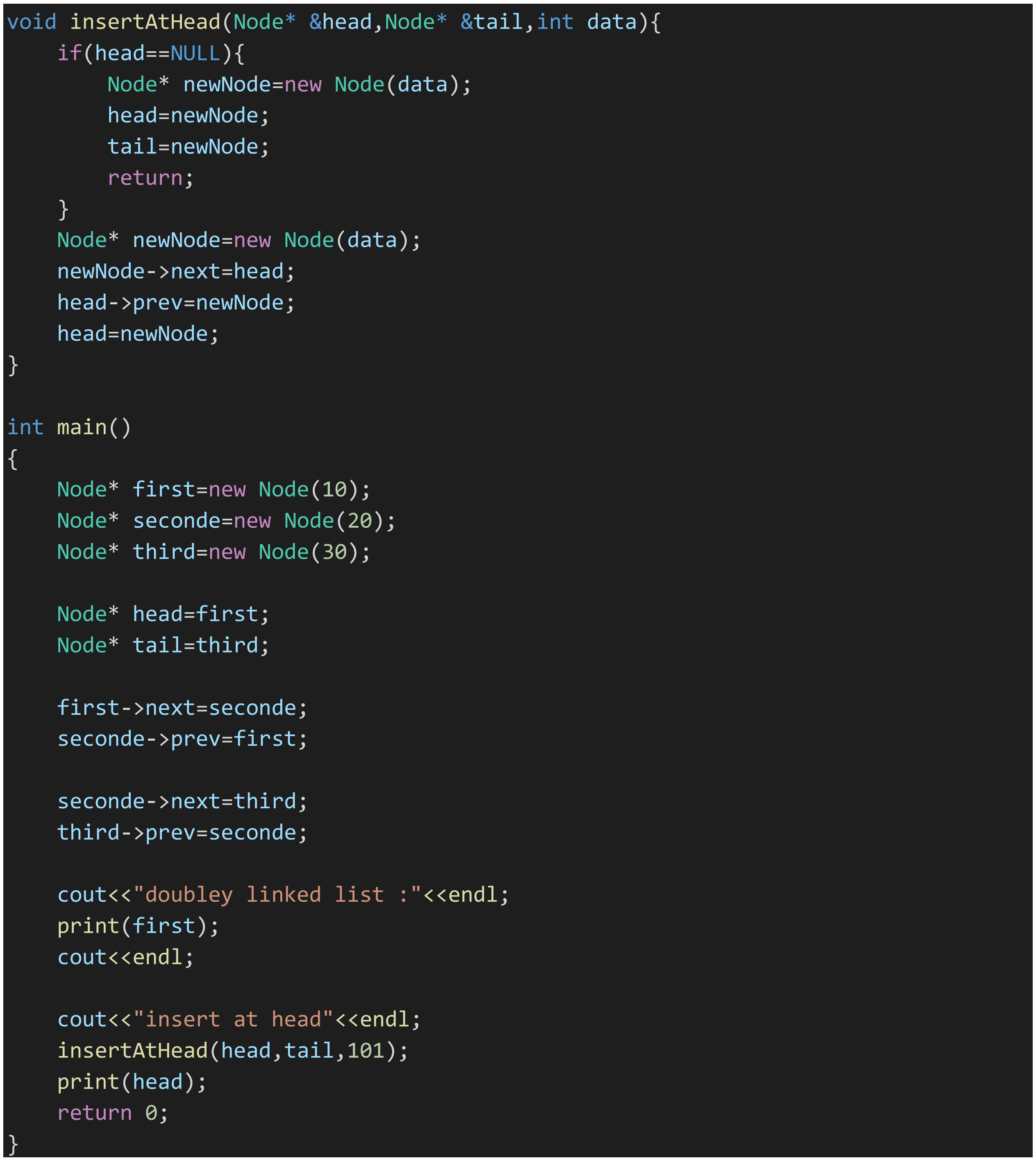
```cpp
int main()
{
    Node* first=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);

    first->next=seconde;
    seconde->prev=first;

    seconde->next=third;
    third->prev=seconde;

    cout<<"doubley linked list :"<<endl;
    print(first);
    return 0;
}
```



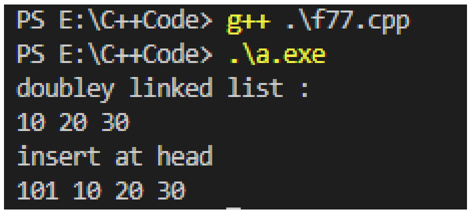
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
doubley linked list :
10 20 30
```

Insert at head

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;
    Node* prev;

    Node(){
        this->data=0;
        this->next=NULL;
        this->prev=NULL;
    }
     Node(int data){
        this->data=data;
        this->next=NULL;
        this->prev=NULL;
    }
};

void print (Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data<<" ";
       temp=temp->next;
    }

}

int getLength(Node* head){
    int length=0;
    Node* temp=head;
    while(temp!=NULL){
        temp=temp->next;
        length++;
    }
    return length;
}
```



```cpp
void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head->prev=newNode;
    head=newNode;
}

int main()
{
    Node* first=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);

    Node* head=first;
    Node* tail=third;

    first->next=seconde;
    seconde->prev=first;

    seconde->next=third;
    third->prev=seconde;

    cout<<"doubley linked list :"<<endl;
    print(first);
    cout<<endl;

    cout<<"insert at head"<<endl;
    insertAtHead(head,tail,101);
    print(head);
    return 0;
}
```



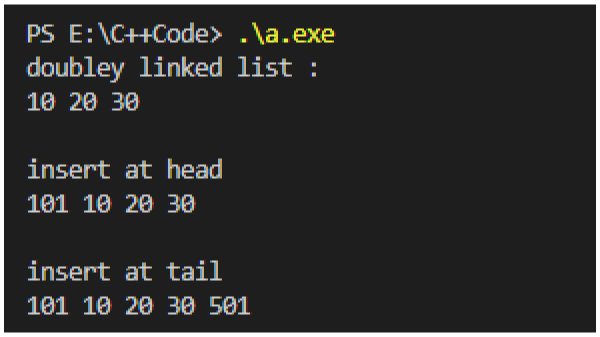
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
doubley linked list :
10 20 30
insert at head
101 10 20 30
```

Insert Node at tail

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;
    Node* prev;

    Node(){
        this->data=0;
        this->next=NULL;
        this->prev=NULL;
    }
     Node(int data){
        this->data=data;
        this->next=NULL;
        this->prev=NULL;
    }
};

void print (Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data<<" ";
       temp=temp->next;
    }

}

int getLength(Node* head){
    int length=0;
    Node* temp=head;
    while(temp!=NULL){
        temp=temp->next;
        length++;
    }
    return length;
}
```

```cpp
void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head->prev=newNode;
    head=newNode;
}

void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
     Node* newNode=new Node(data);
     tail->next=newNode;
     newNode->prev=tail;
     tail=newNode;
}
```

```cpp
int main()
{
    Node* first=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);

    Node* head=first;
    Node* tail=third;

    first->next=seconde;
    seconde->prev=first;

    seconde->next=third;
    third->prev=seconde;

    cout<<"doubley linked list :"<<endl;
    print(first);
    cout<<endl<<endl;

    cout<<"insert at head"<<endl;
    insertAtHead(head,tail,101);
    print(head);
    cout<<endl<<endl;

    cout<<"insert at tail "<<endl;
    insertAtTail(head,tail,501);
    print(head);
    cout<<endl<<endl;

    return 0;
}
```

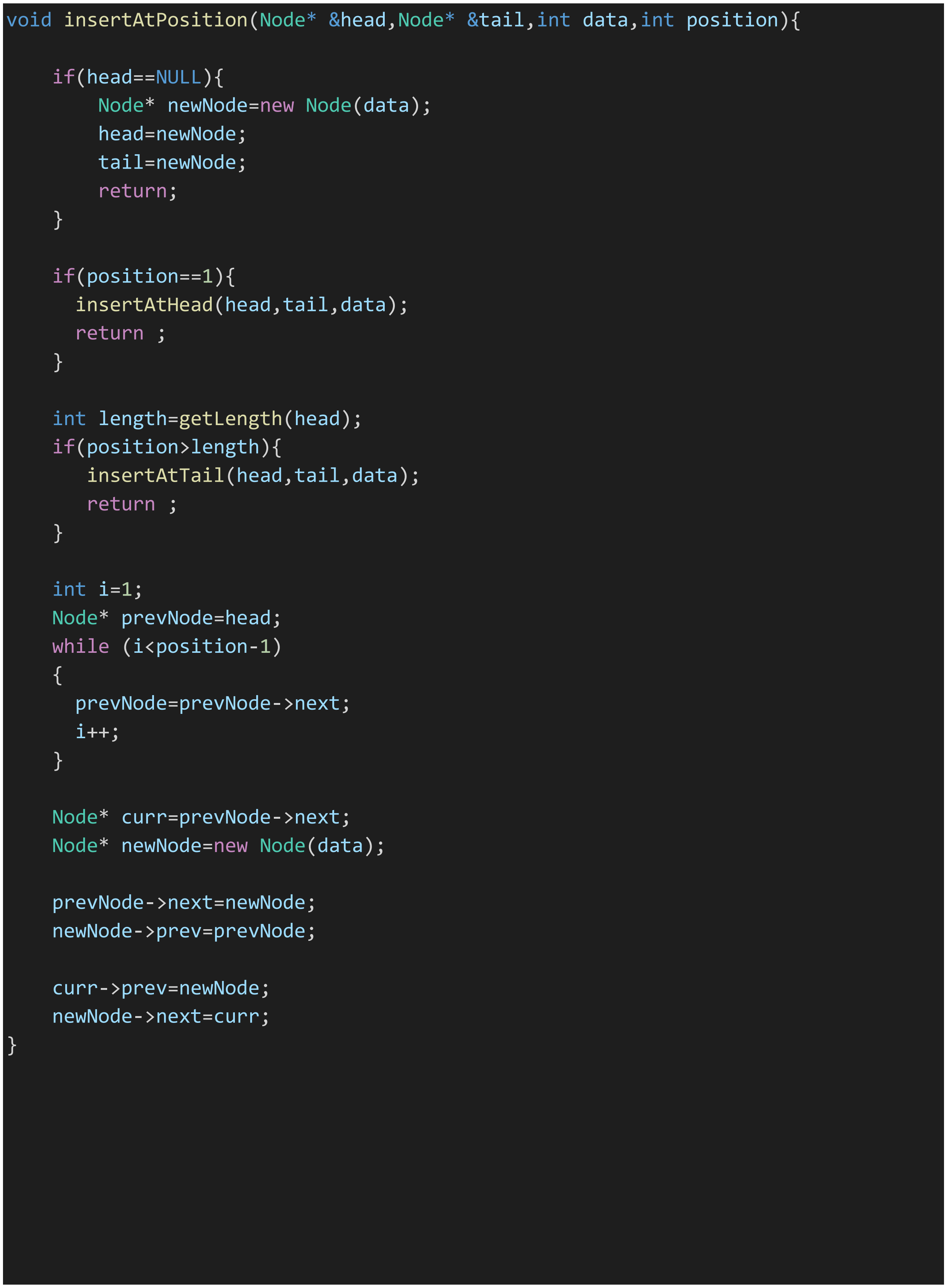


```
PS E:\C++Code> .\a.exe
doubley linked list :
10 20 30

insert at head
101 10 20 30

insert at tail
101 10 20 30 501
```

Insert Node at Middle

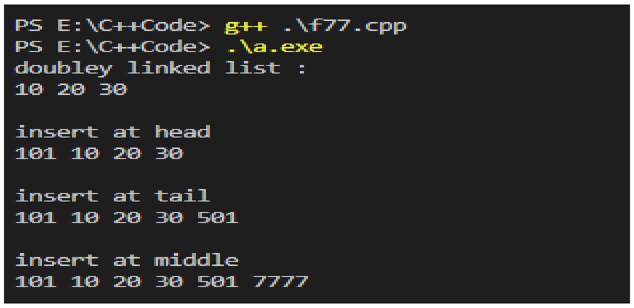
```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;
    Node* prev;

    Node(){
        this->data=0;
        this->next=NULL;
        this->prev=NULL;
    }
     Node(int data){
        this->data=data;
        this->next=NULL;
        this->prev=NULL;
    }
};

void print (Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data<<" ";
       temp=temp->next;
    }

}

int getLength(Node* head){
    int length=0;
    Node* temp=head;
    while(temp!=NULL){
        temp=temp->next;
        length++;
    }
    return length;
}
```

```
void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head->prev=newNode;
    head=newNode;
}

void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
     Node* newNode=new Node(data);
     tail->next=newNode;
     newNode->prev=tail;
     tail=newNode;
}
```



```
void insertAtPosition(Node* &head,Node* &tail,int data,int position){

    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==1){
      insertAtHead(head,tail,data);
      return ;
    }

    int length=getLength(head);
    if(position>length){
       insertAtTail(head,tail,data);
       return ;
    }

    int i=1;
    Node* prevNode=head;
    while (i<position-1)
    {
      prevNode=prevNode->next;
      i++;
    }

    Node* curr=prevNode->next;
    Node* newNode=new Node(data);

    prevNode->next=newNode;
    newNode->prev=prevNode;

    curr->prev=newNode;
    newNode->next=curr;
}
```

```cpp
int main()
{
    Node* first=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);

    Node* head=first;
    Node* tail=third;

    first->next=seconde;
    seconde->prev=first;

    seconde->next=third;
    third->prev=seconde;

    cout<<"doubley linked list :"<<endl;
    print(first);
    cout<<endl<<endl;

    cout<<"insert at head"<<endl;
    insertAtHead(head,tail,101);
    print(head);
    cout<<endl<<endl;

    cout<<"insert at tail "<<endl;
    insertAtTail(head,tail,501);
    print(head);
    cout<<endl<<endl;

    cout<<"insert at middle "<<endl;
    insertAtPosition(head,tail,7777,6);
    print(head);
    cout<<endl<<endl;

    return 0;
}
```

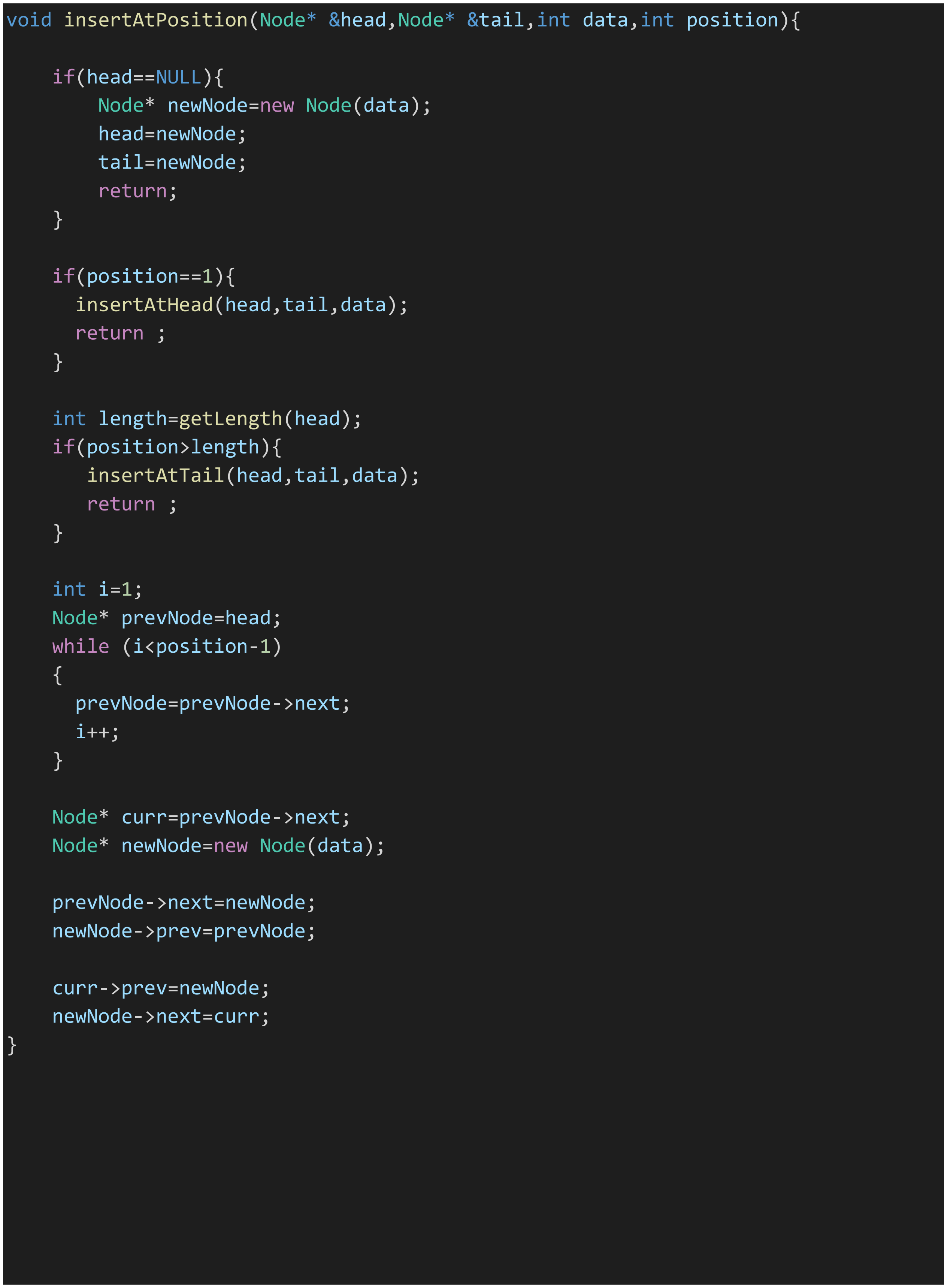


```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
doubley linked list :
10 20 30

insert at head
101 10 20 30

insert at tail
101 10 20 30 501

insert at middle
101 10 20 30 501 7777
```

Deleting Node

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

class Node{
    public:
    int data;
    Node* next;
    Node* prev;

    Node(){
        this->data=0;
        this->next=NULL;
        this->prev=NULL;
    }
     Node(int data){
        this->data=data;
        this->next=NULL;
        this->prev=NULL;
    }
    ~Node(){
      cout<<"Node with value :"<<this->data<<" deleted"<<endl;
    }
};

void print (Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
       cout<<temp->data<<" ";
       temp=temp->next;
    }

}

int getLength(Node* head){
    int length=0;
    Node* temp=head;
    while(temp!=NULL){
        temp=temp->next;
        length++;
    }
    return length;
}
```

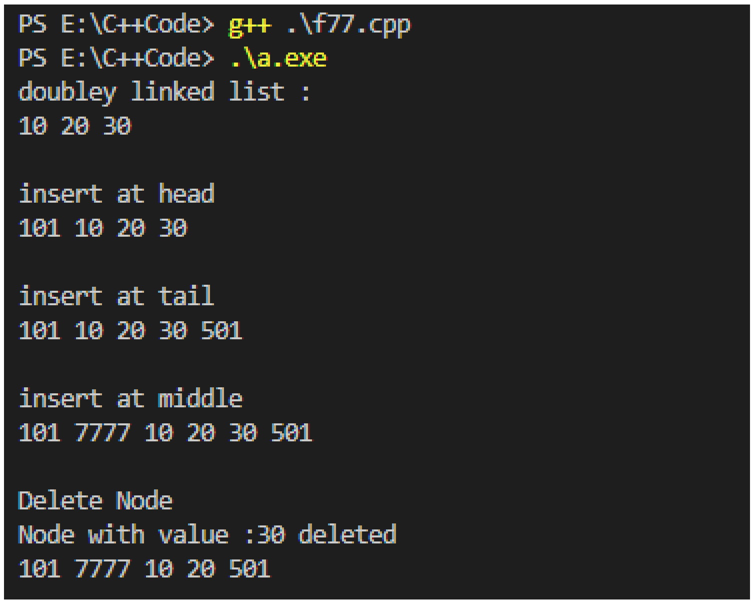
```
void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head->prev=newNode;
    head=newNode;
}

void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }
     Node* newNode=new Node(data);
     tail->next=newNode;
     newNode->prev=tail;
     tail=newNode;
}
```



```
void insertAtPosition(Node* &head,Node* &tail,int data,int position){

    if(head==NULL){
        Node* newNode=new Node(data);
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==1){
      insertAtHead(head,tail,data);
      return ;
    }

    int length=getLength(head);
    if(position>length){
       insertAtTail(head,tail,data);
       return ;
    }

    int i=1;
    Node* prevNode=head;
    while (i<position-1)
    {
      prevNode=prevNode->next;
      i++;
    }

    Node* curr=prevNode->next;
    Node* newNode=new Node(data);

    prevNode->next=newNode;
    newNode->prev=prevNode;

    curr->prev=newNode;
    newNode->next=curr;
}
```

```
void deleteFromPos(Node* &head,Node* &tail,int position){

   if(head==NULL){
      cout<<"linked list is empty";
      return;

   }

   if(head->next==NULL){
      Node*temp =head;
      head=NULL;
      tail=NULL;
      delete temp;
      return;
   }

   int len=getLength(head);
   if(position>len){
      cout<<"please enter a valid linked list ";
   }

   if(position==1){
      Node* temp=head;
      head=head->next;
      head->prev=NULL;
      temp->next=NULL;
      delete temp;
      return;
   }

   //int len=getLength(head);
   if(position==len){
      Node*temp=tail;
      tail=tail->prev;
      temp->prev=NULL;
      tail->next=NULL;
      delete temp;
      return;
   }
   int i=1;
   Node* left=head;
   while(i<position-1){
      left=left->next;
      i++;
   }
   Node* curr=left->next;
   Node* right=curr->next;

   left->next=right;
   right->prev=left;

   curr->next=NULL;
   curr->prev=NULL;
   delete curr;

}
```

```cpp
int main()
{
    Node* first=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);

    Node* head=first;
    Node* tail=third;

    first->next=seconde;
    seconde->prev=first;

    seconde->next=third;
    third->prev=seconde;

    cout<<"doubley linked list :"<<endl;
    print(first);
    cout<<endl<<endl;

    cout<<"insert at head"<<endl;
    insertAtHead(head,tail,101);
    print(head);
    cout<<endl<<endl;

    cout<<"insert at tail "<<endl;
    insertAtTail(head,tail,501);
    print(head);
    cout<<endl<<endl;

    cout<<"insert at middle "<<endl;
    insertAtPosition(head,tail,7777,2);
    print(head);
    cout<<endl<<endl;

    cout<<"Delete Node "<<endl;
    deleteFromPos(head,tail,5);
    print(head);
    cout<<endl<<endl;

    return 0;
}
```



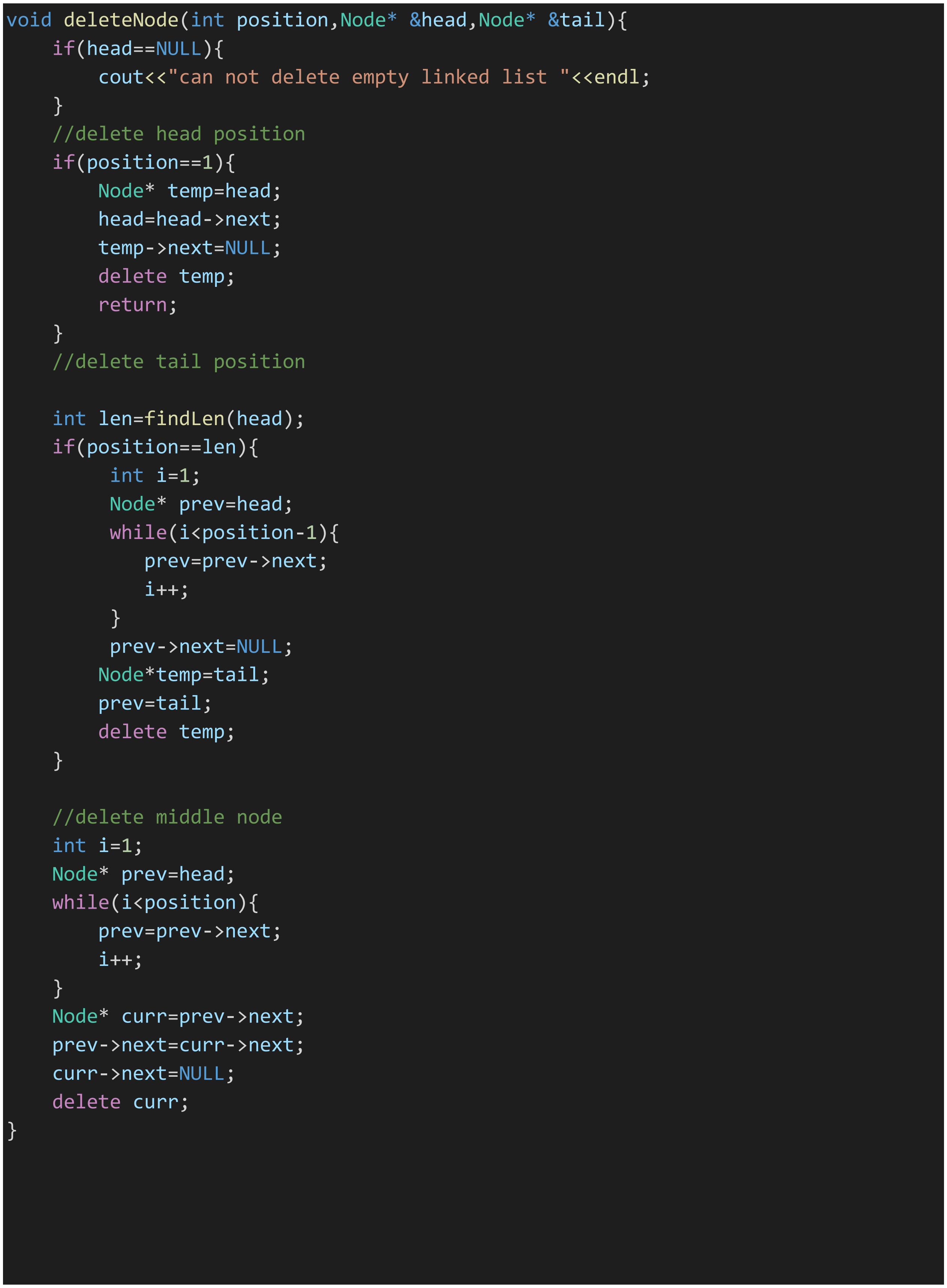
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
doubley linked list :
10 20 30

insert at head
101 10 20 30

insert at tail
101 10 20 30 501

insert at middle
101 7777 10 20 30 501

Delete Node
Node with value :30 deleted
101 7777 10 20 501
```

Reverse linked list using Loop and Recursion

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void insertAtHead(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    newNode->next=head;
    head=newNode;
 }
```

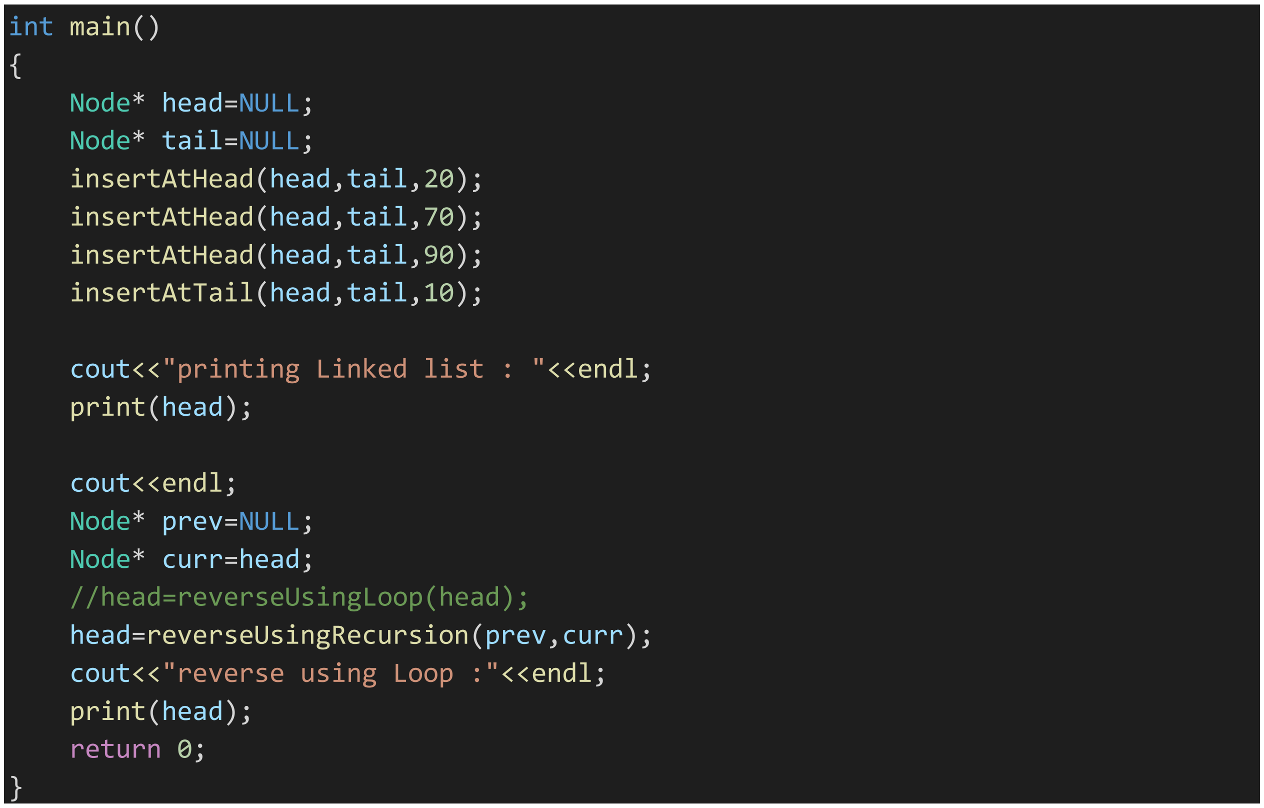
```cpp
void insertAtTail(Node* &head,Node* &tail,int data){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }
    Node* newNode=new Node(data);
    tail->next=newNode;
    tail=newNode;
}

void print(Node* &head){
    Node* temp=head;
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    cout<<endl;
 }
int findLen(Node* &head){
     Node* temp=head;
     int i=0;
     while (temp!=NULL)
     {
       temp=temp->next;
       i++;
     }
     return i;

}
```

```cpp
void insertAtPosition(int data,int position,Node* &head,Node* &tail){
    if(head==NULL){
        Node* newNode=new Node(data);
        newNode->next=head;
        head=newNode;
        tail=newNode;
        return;
    }

    if(position==0){
        insertAtHead(head,tail,data);
        return;
    }

    int len=findLen(head);
    if(position>=len){
        insertAtTail(head,tail,data);
        return;
    }
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    Node* newNode=new Node(data);
    newNode->next=curr;
    prev->next=newNode;
}
```

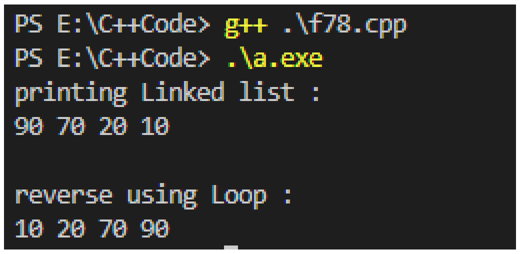


```cpp
void deleteNode(int position,Node* &head,Node* &tail){
    if(head==NULL){
        cout<<"can not delete empty linked list "<<endl;
    }
    //delete head position
    if(position==1){
        Node* temp=head;
        head=head->next;
        temp->next=NULL;
        delete temp;
        return;
    }
    //delete tail position

    int len=findLen(head);
    if(position==len){
         int i=1;
         Node* prev=head;
         while(i<position-1){
            prev=prev->next;
            i++;
         }
         prev->next=NULL;
        Node*temp=tail;
        prev=tail;
        delete temp;
    }

    //delete middle node
    int i=1;
    Node* prev=head;
    while(i<position){
        prev=prev->next;
        i++;
    }
    Node* curr=prev->next;
    prev->next=curr->next;
    curr->next=NULL;
    delete curr;
}
```

```cpp
//reverse linked list using loop
Node* reverseUsingLoop(Node* head){
    Node* prev=NULL;
    Node* curr=head;

    while (curr!=NULL)
    {
        Node* temp=curr->next;
        curr->next=prev;
        prev=curr;
        curr=temp;
    }
    return prev;

}

//reverse linked list using recursion
Node* reverseUsingRecursion(Node* prev,Node* curr){
    if(curr==NULL){
        return prev;
    }
    Node* temp=curr->next;
    curr->next=prev;
    prev=curr;
    curr=temp;

    reverseUsingRecursion(prev,curr);
}
```



```cpp
int main()
{
    Node* head=NULL;
    Node* tail=NULL;
    insertAtHead(head,tail,20);
    insertAtHead(head,tail,70);
    insertAtHead(head,tail,90);
    insertAtTail(head,tail,10);

    cout<<"printing Linked list : "<<endl;
    print(head);

    cout<<endl;
    Node* prev=NULL;
    Node* curr=head;
    //head=reverseUsingLoop(head);
    head=reverseUsingRecursion(prev,curr);
    cout<<"reverse using Loop :"<<endl;
    print(head);
    return 0;
}
```

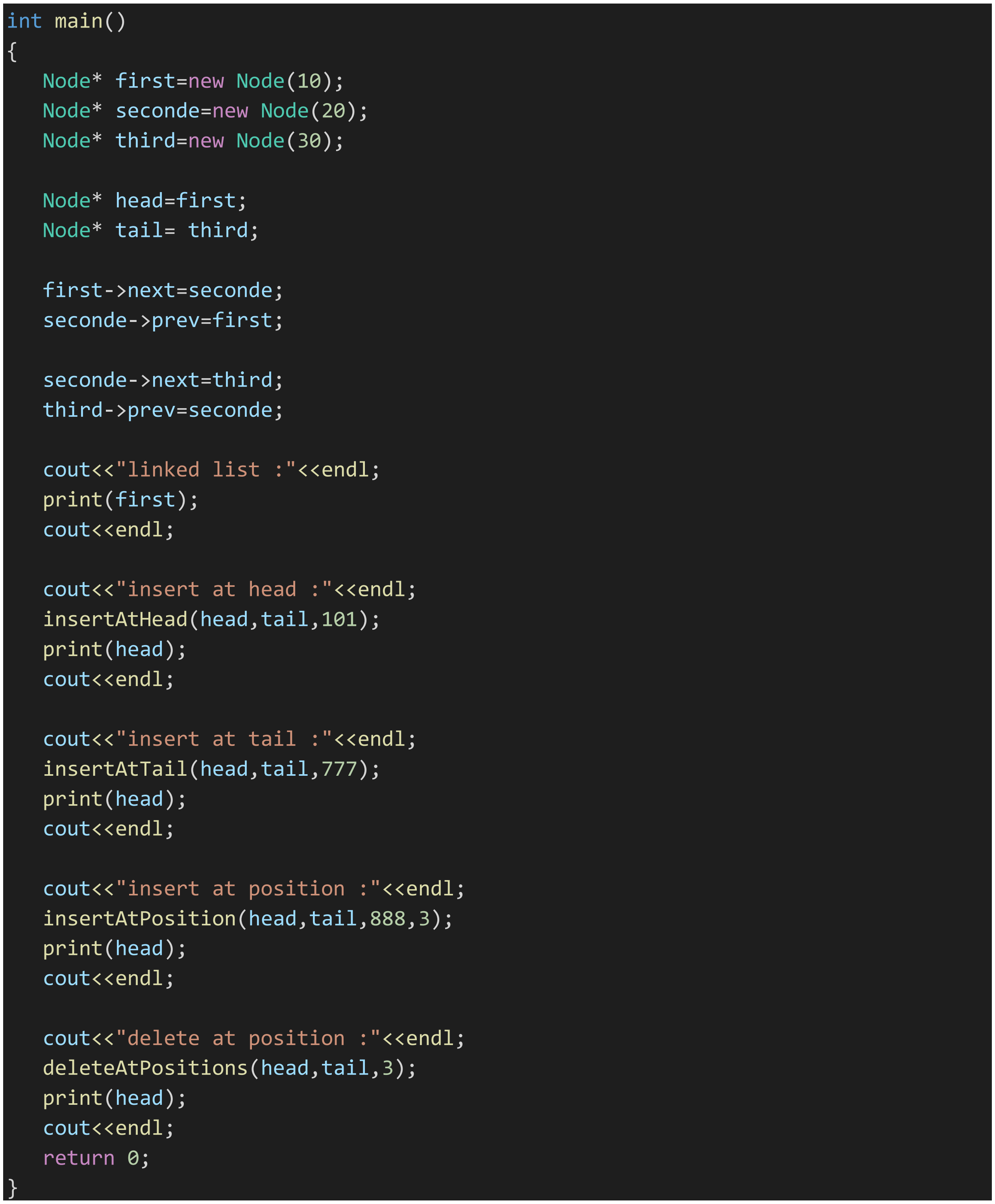


```
PS E:\C++Code> g++ .\f78.cpp
PS E:\C++Code> .\a.exe
printing Linked list :
90 70 20 10

reverse using Loop :
10 20 70 90
```

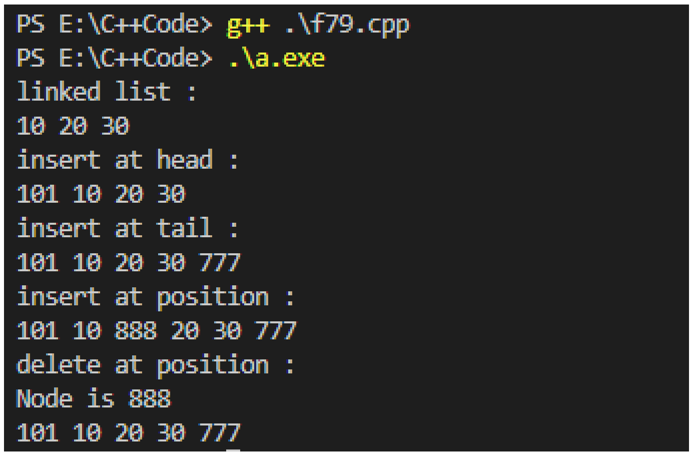
Q) insert ,delete operation on linked list

```
#include <iostream>

using namespace std;

class Node{
   public:
   int data;
   Node* next;
   Node* prev;

   Node(){
      this->data=0;
      this->next=NULL;
      this->prev=NULL;
   }

   Node(int data){
      this->data=data;
      this->next=NULL;
      this->prev=NULL;
   }
   ~Node(){
      cout<<"Node is "<<this->data<<endl;
   }
};


int  getLength(Node* head){
   Node* temp =head;
   int i=0;
   while (temp!=NULL)
   {
      temp=temp->next;
      i++;
   }
   return i;
}
```

```
void insertAtHead(Node* &head,Node* &tail,int data){
   if(head==NULL){
      Node* newNode=new Node(data);
      head=newNode;
      tail=newNode;
      return;
   }
   Node* newNode=new Node(data);
   newNode->next=head;
   head->prev=newNode;
   head=newNode;

}

void insertAtTail(Node* &head,Node* &tail,int data){
   if(head==NULL){
      Node* newNode=new Node(data);
      head=newNode;
      tail=newNode;
      return;
   }
   Node* newNode=new Node(data);
   tail->next=newNode;
   newNode->prev=tail;
   tail=newNode;
}

void insertAtPosition(Node* &head,Node* &tail,int data,int position){
   if(head==NULL){
      Node* newNode=new Node(data);
      head=newNode;
      tail=newNode;
      return;
   }
   if(position==1){
      insertAtHead(head,tail,data);
      return;
   }

   int len=getLength(head);
   if(position>len){
      insertAtTail(head,tail,data);
      return;
   }
   int i=1;
   Node* prevNode=head;
   while(i<position-1){
      prevNode=prevNode->next;
      i++;
   }
   Node* curr=prevNode->next;
   Node* newNode=new Node(data);

   prevNode->next=newNode;
   newNode->prev=prevNode;

   curr->prev=newNode;
   newNode->next=curr;
}

void deleteAtPositions(Node* &head,Node* &tail,int position){
   if(head==NULL){
      cout<<"linked list is empty "<<endl;
      return;
   }

   if(head->next==NULL){
      Node* temp=head;
      head=NULL;
      tail=NULL;
      delete temp;
      return;
   }

   int len=getLength(head);

   if(position>len){
      cout<<"enter valid linked list number "<<endl;
      return;
   }
   if(position==1){
      Node* temp=head;
      head=head->next;
      head->prev=NULL;
      temp->next=NULL;
      delete temp;
      return;
   }
   if(position==len){
      Node* temp=tail;
      tail=tail->prev;
      temp->prev=NULL;
      tail->next=NULL;
      delete temp;
      return;
   }
   int i=1;
   Node* left=head;
   while (i<position-1)
   {
      left=left->next;
      i++;
   }

   Node* curr=left->next;
   Node* right=curr->next;

   left->next=right;
   right->prev=left;

   curr->next=NULL;
   curr->prev=NULL;

   delete curr;

}

void print(Node* &head){
   Node* temp =head;
    while (temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->next;

   }
}
```



```
int main()
{
   Node* first=new Node(10);
   Node* seconde=new Node(20);
   Node* third=new Node(30);

   Node* head=first;
   Node* tail= third;

   first->next=seconde;
   seconde->prev=first;

   seconde->next=third;
   third->prev=seconde;

   cout<<"linked list :"<<endl;
   print(first);
   cout<<endl;

   cout<<"insert at head :"<<endl;
   insertAtHead(head,tail,101);
   print(head);
   cout<<endl;

   cout<<"insert at tail :"<<endl;
   insertAtTail(head,tail,777);
   print(head);
   cout<<endl;

   cout<<"insert at position :"<<endl;
   insertAtPosition(head,tail,888,3);
   print(head);
   cout<<endl;

   cout<<"delete at position :"<<endl;
   deleteAtPositions(head,tail,3);
   print(head);
   cout<<endl;
   return 0;
}
```

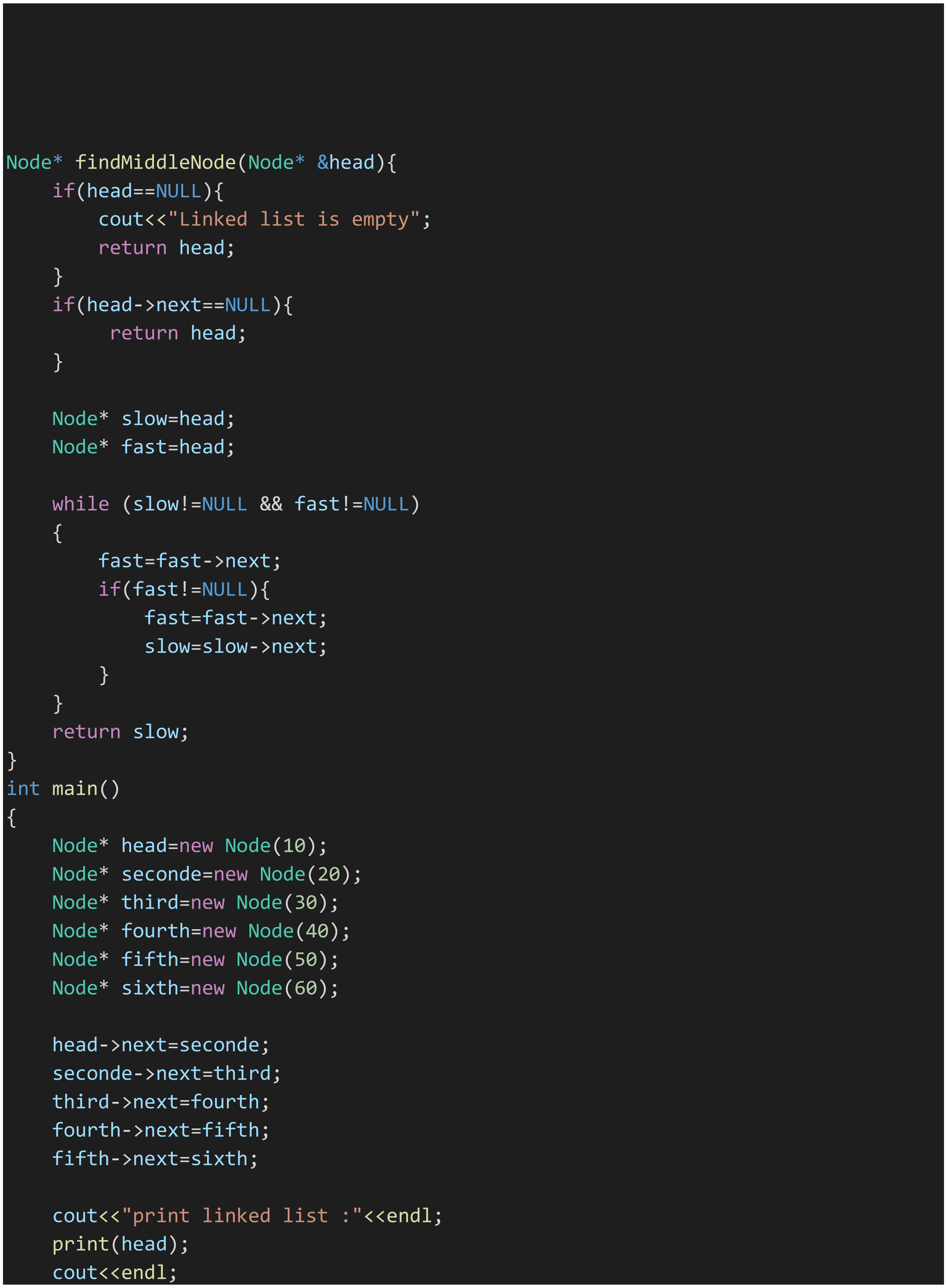


```
PS E:\C++Code> g++ .\f79.cpp
PS E:\C++Code> .\a.exe
linked list :
10 20 30
insert at head :
101 10 20 30
insert at tail :
101 10 20 30 777
insert at position :
101 10 888 20 30 777
delete at position :
Node is 888
101 10 20 30 777
```

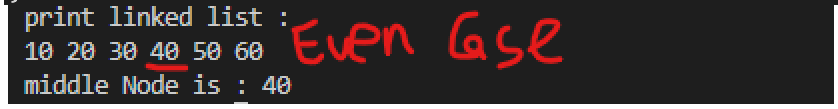
LL Class-3

Q)Find Middle Node (Impotant Question)

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }

}
```



```
Node* findMiddleNode(Node* &head){
    if(head==NULL){
        cout<<"Linked list is empty";
        return head;
    }
    if(head->next==NULL){
         return head;
    }

    Node* slow=head;
    Node* fast=head;

    while (slow!=NULL && fast!=NULL)
    {
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
        }
    }
    return slow;
}
int main()
{
    Node* head=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);
    Node* sixth=new Node(60);

    head->next=seconde;
    seconde->next=third;
    third->next=fourth;
    fourth->next=fifth;
    fifth->next=sixth;

    cout<<"print linked list :"<<endl;
    print(head);
    cout<<endl;
    cout<<"middle Node is : "<<findMiddleNode(head)->data;
    cout<<endl;
    return 0;
}
```

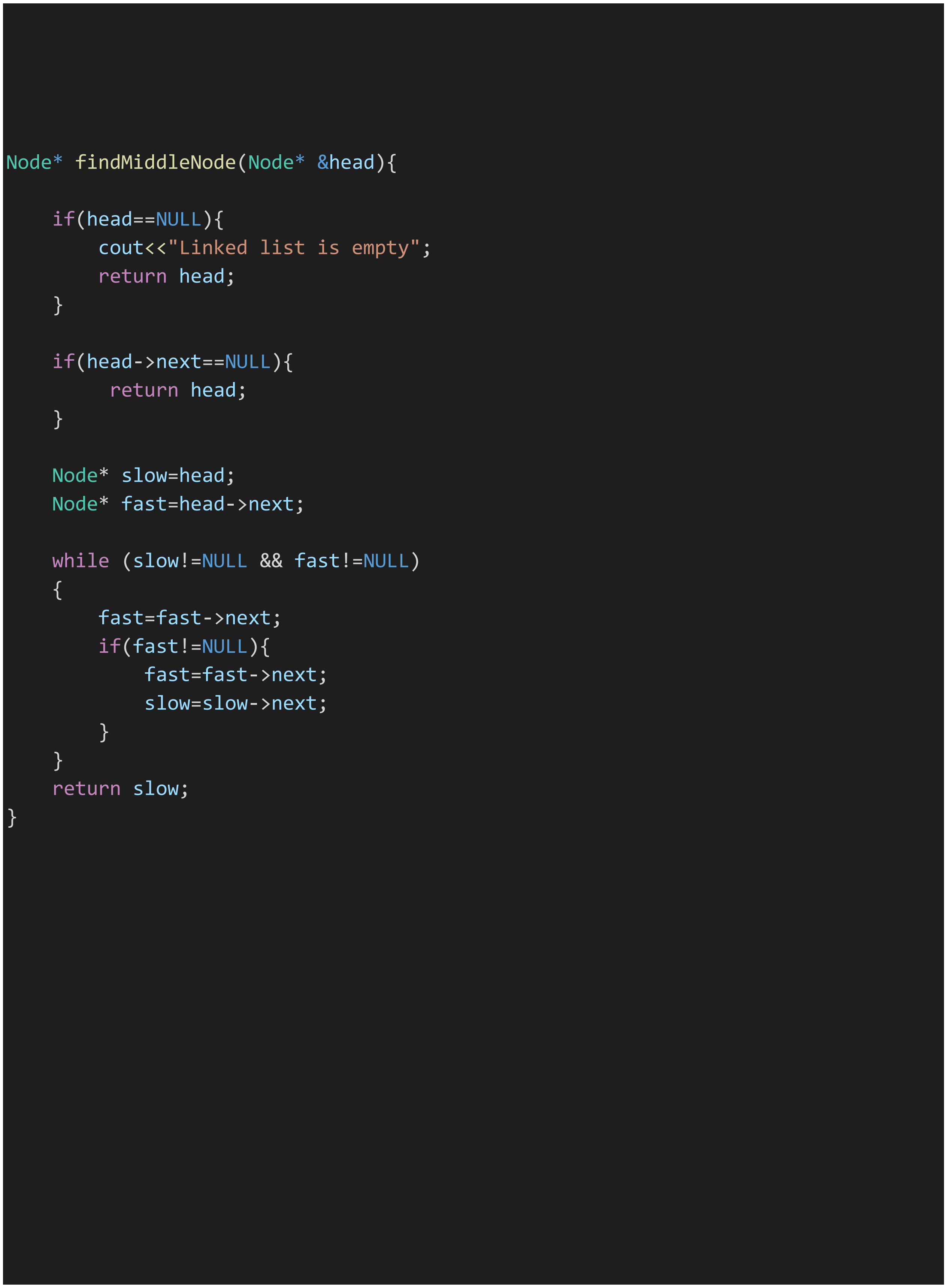


```
print linked list :
10 20 30 40 50 60
middle Node is : 40
```

Even Case

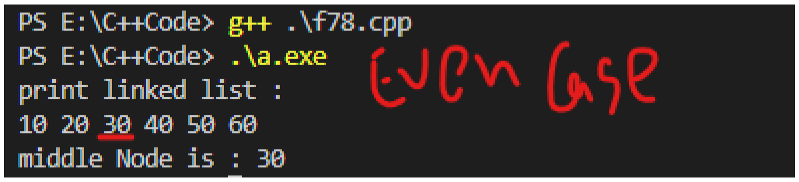
2nd Approach

```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }

}
```



```cpp
Node* findMiddleNode(Node* &head){

    if(head==NULL){
        cout<<"Linked list is empty";
        return head;
    }

    if(head->next==NULL){
         return head;
    }

    Node* slow=head;
    Node* fast=head->next;

    while (slow!=NULL && fast!=NULL)
    {
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
        }
    }
    return slow;
}
```

```cpp
int main()
{
    Node* head=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);
    Node* sixth=new Node(60);

    head->next=seconde;
    seconde->next=third;
    third->next=fourth;
    fourth->next=fifth;
    fifth->next=sixth;

    cout<<"print linked list :"<<endl;
    print(head);
    cout<<endl;

    cout<<"middle Node is : "<<findMiddleNode(head)->data;
    cout<<endl;
    return 0;
}
```



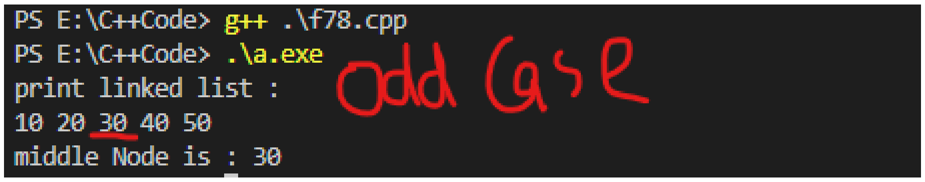
```
PS E:\C++Code> g++ .\f78.cpp
PS E:\C++Code> .\a.exe
print linked list :
10 20 30 40 50 60
middle Node is : 30
```

Even Case

3rd Approach

```cpp
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }

}
```

```cpp
Node* findMiddleNode(Node* &head){

    if(head==NULL){
        cout<<"Linked list is empty";
        return head;
    }

    if(head->next==NULL){
         return head;
    }

    Node* slow=head;
    Node* fast=head;

    while (slow!=NULL && fast!=NULL)
    {
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
        }
    }
    return slow;
}
int main()
{
    Node* head=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);


    head->next=seconde;
    seconde->next=third;
    third->next=fourth;
    fourth->next=fifth;
    cout<<"print linked list :"<<endl;
    print(head);
    cout<<endl;

    cout<<"middle Node is : "<<findMiddleNode(head)->data;
    cout<<endl;
    return 0;
}
```

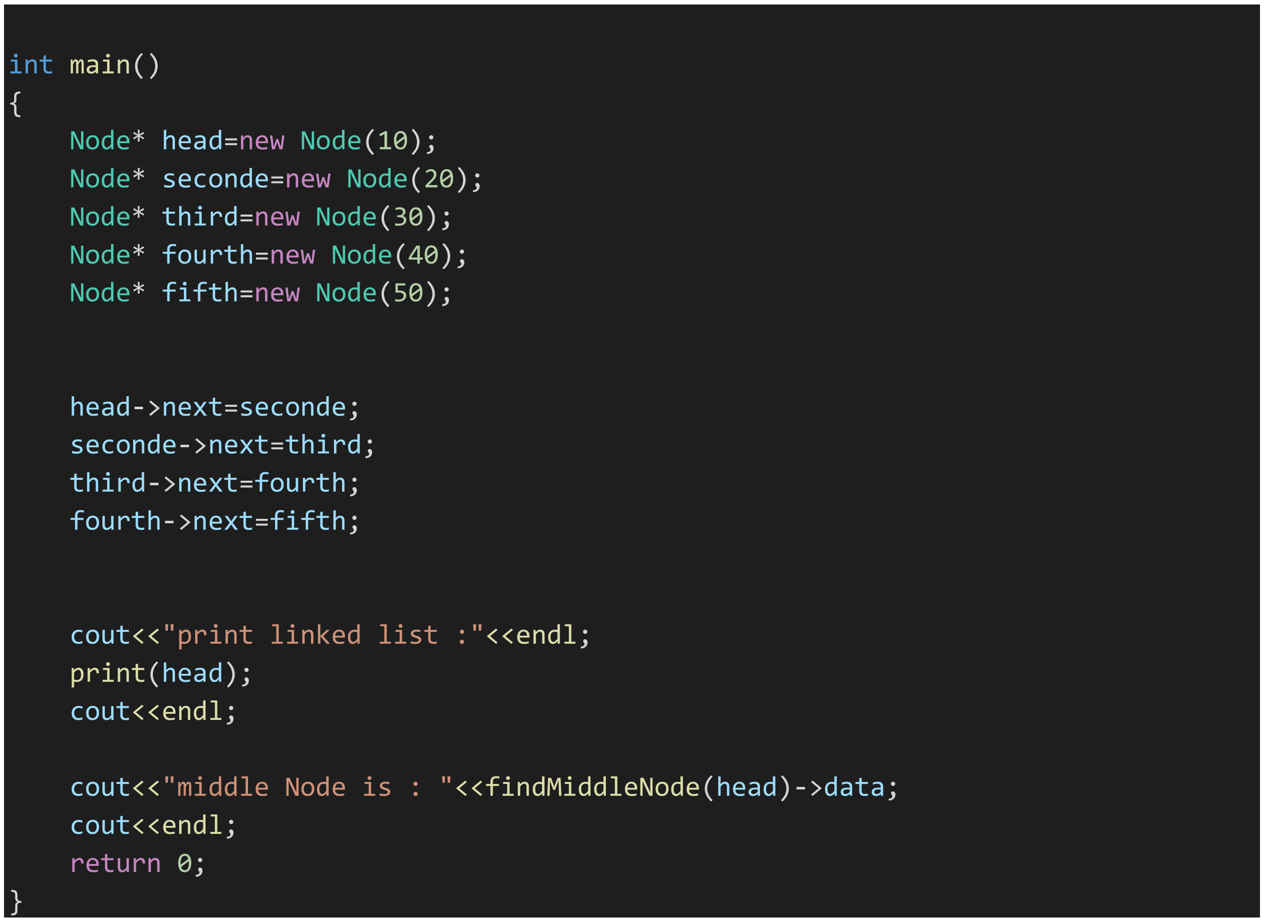


```
PS E:\C++Code> g++ .\f78.cpp
PS E:\C++Code> .\a.exe
print linked list :
10 20 30 40 50
middle Node is : 30
```

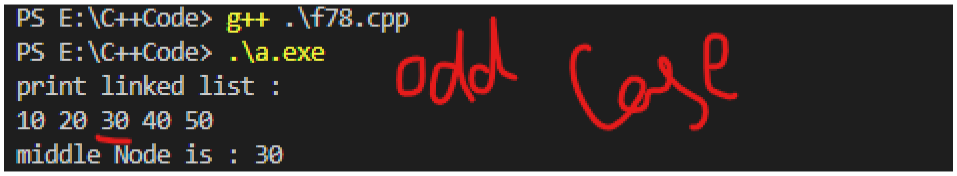
4th Approach

```cpp
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};
```

```cpp
void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }
}

Node* findMiddleNode(Node* &head){

    if(head==NULL){
        cout<<"Linked list is empty";
        return head;
    }

    if(head->next==NULL){
         return head;
    }

    Node* slow=head;
    Node* fast=head->next;

    while (slow!=NULL && fast!=NULL)
    {
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
        }
    }
    return slow;
}
```



```cpp
int main()
{
    Node* head=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);


    head->next=seconde;
    seconde->next=third;
    third->next=fourth;
    fourth->next=fifth;


    cout<<"print linked list :"<<endl;
    print(head);
    cout<<endl;

    cout<<"middle Node is : "<<findMiddleNode(head)->data;
    cout<<endl;
    return 0;
}
```

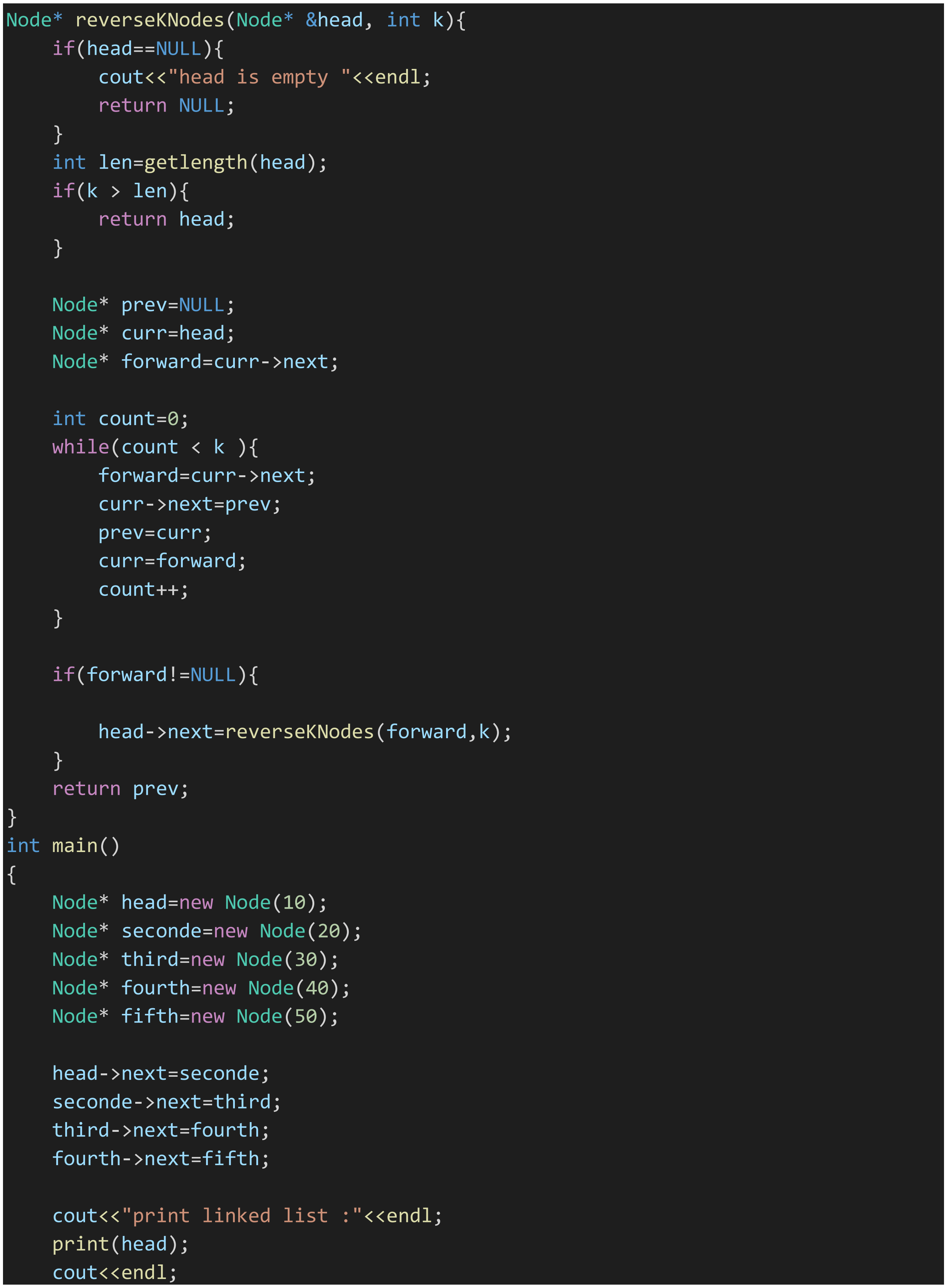


```
PS E:\C++Code> g++ .\f78.cpp
PS E:\C++Code> .\a.exe
print linked list :
10 20 30 40 50
middle Node is : 30
```

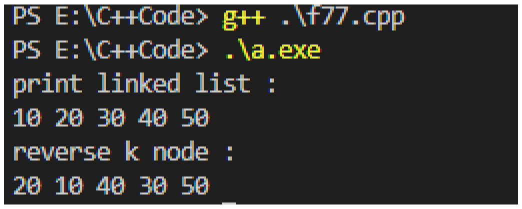
odd Case

Q)K groups –reverse Linked list (Important question)

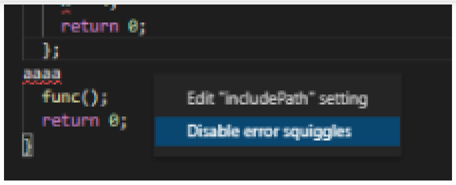
```
#include <iostream>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }

}
int getlength(Node* head){
    Node* temp=head;
    int len=0;
    while(temp!=NULL){
        temp=temp->next;
        len++;
    }
    return len++;
}
```



```
Node* reverseKNodes(Node* &head, int k){
    if(head==NULL){
        cout<<"head is empty "<<endl;
        return NULL;
    }
    int len=getlength(head);
    if(k > len){
        return head;
    }

    Node* prev=NULL;
    Node* curr=head;
    Node* forward=curr->next;

    int count=0;
    while(count < k ){
        forward=curr->next;
        curr->next=prev;
        prev=curr;
        curr=forward;
        count++;
    }

    if(forward!=NULL){

        head->next=reverseKNodes(forward,k);
    }
    return prev;
}
int main()
{
    Node* head=new Node(10);
    Node* seconde=new Node(20);
    Node* third=new Node(30);
    Node* fourth=new Node(40);
    Node* fifth=new Node(50);

    head->next=seconde;
    seconde->next=third;
    third->next=fourth;
    fourth->next=fifth;

    cout<<"print linked list :"<<endl;
    print(head);
    cout<<endl;
    cout<<"reverse k node :"<<endl;
    head=reverseKNodes(head, 2);
    print(head);
    cout<<endl;
    return 0;
}
```

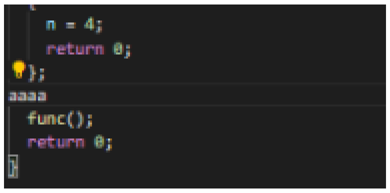


```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
print linked list :
10 20 30 40 50
reverse k node :
20 10 40 30 50
```



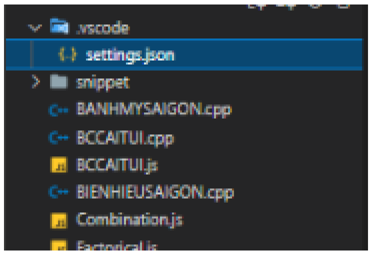
```
    return 0;
  };
aaaa
  func();
  return 0;
}
```

After ignoring, I got your problem:

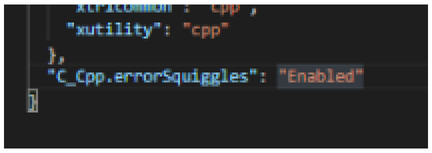


```
    n = 4;
    return 0;
  };
aaaa
  func();
  return 0;
}
```

To fix this, open settings.json file:



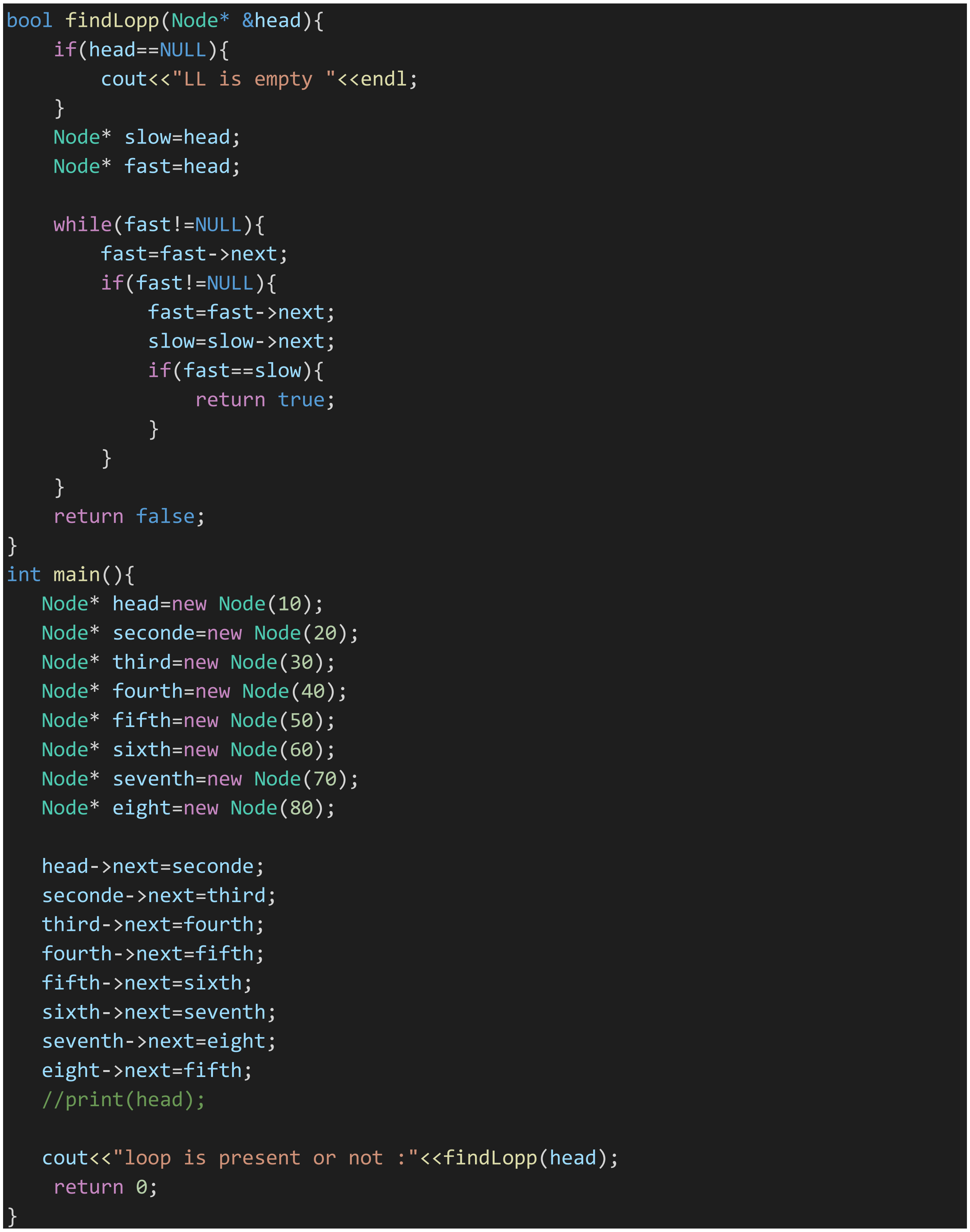
Scroll to the end, then set "C_Cpp.errorSquiggles": from **Disabled** to **Enabled**.



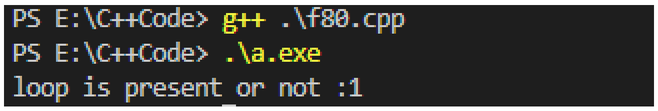
```
        "xutility": "cpp"
    },
    "C_Cpp.errorSquiggles": "Enabled"
}
```

Q) find loop is present ot not (Important Question)

```
#include<iostream>
#include<vector>

using namespace std;

class Node{
    public:
   int data;
   Node* next;
   Node* prev;

   Node(){
      this->data=0;
      this->next=NULL;
      this->prev=NULL;
   }

   Node(int data){
      this->data=data;
      this->next=NULL;
      this->prev=NULL;
   }

};

int  getLength(Node* head){
   Node* temp =head;
   int i=0;
   while (temp!=NULL)
   {
      temp=temp->next;
      i++;
   }
   return i;
}
void print(Node* &head){
   Node* temp =head;
    while (temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->next;
   }
}
```



```
bool findLopp(Node* &head){
    if(head==NULL){
        cout<<"LL is empty "<<endl;
    }
    Node* slow=head;
    Node* fast=head;

    while(fast!=NULL){
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
            if(fast==slow){
                return true;
            }
        }
    }
    return false;
}
int main(){
   Node* head=new Node(10);
   Node* seconde=new Node(20);
   Node* third=new Node(30);
   Node* fourth=new Node(40);
   Node* fifth=new Node(50);
   Node* sixth=new Node(60);
   Node* seventh=new Node(70);
   Node* eight=new Node(80);

   head->next=seconde;
   seconde->next=third;
   third->next=fourth;
   fourth->next=fifth;
   fifth->next=sixth;
   sixth->next=seventh;
   seventh->next=eight;
   eight->next=fifth;
   //print(head);

   cout<<"loop is present or not :"<<findLopp(head);
    return 0;
}
```



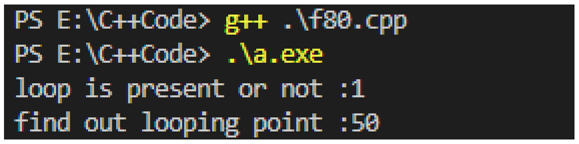
```
PS E:\C++Code> g++ .\f80.cpp
PS E:\C++Code> .\a.exe
loop is present or not :1
```

Q) find starting loop point using Linked List (Important question)

```
#include<iostream>
#include<vector>

using namespace std;

class Node{
    public:
   int data;
   Node* next;
   Node* prev;

   Node(){
      this->data=0;
      this->next=NULL;
      this->prev=NULL;
   }

   Node(int data){
      this->data=data;
      this->next=NULL;
      this->prev=NULL;
   }

};

int  getLength(Node* head){
   Node* temp =head;
   int i=0;
   while (temp!=NULL)
   {
      temp=temp->next;
      i++;
   }
   return i;
}
```

```
void print(Node* &head){
   Node* temp =head;
    while (temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->next;
   }
}

bool findLopp(Node* &head){
    if(head==NULL){
        cout<<"LL is empty "<<endl;
    }
    Node* slow=head;
    Node* fast=head;

    while(fast!=NULL){
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
            if(fast==slow){
                return true;
            }
        }
    }
    return false;
}
```

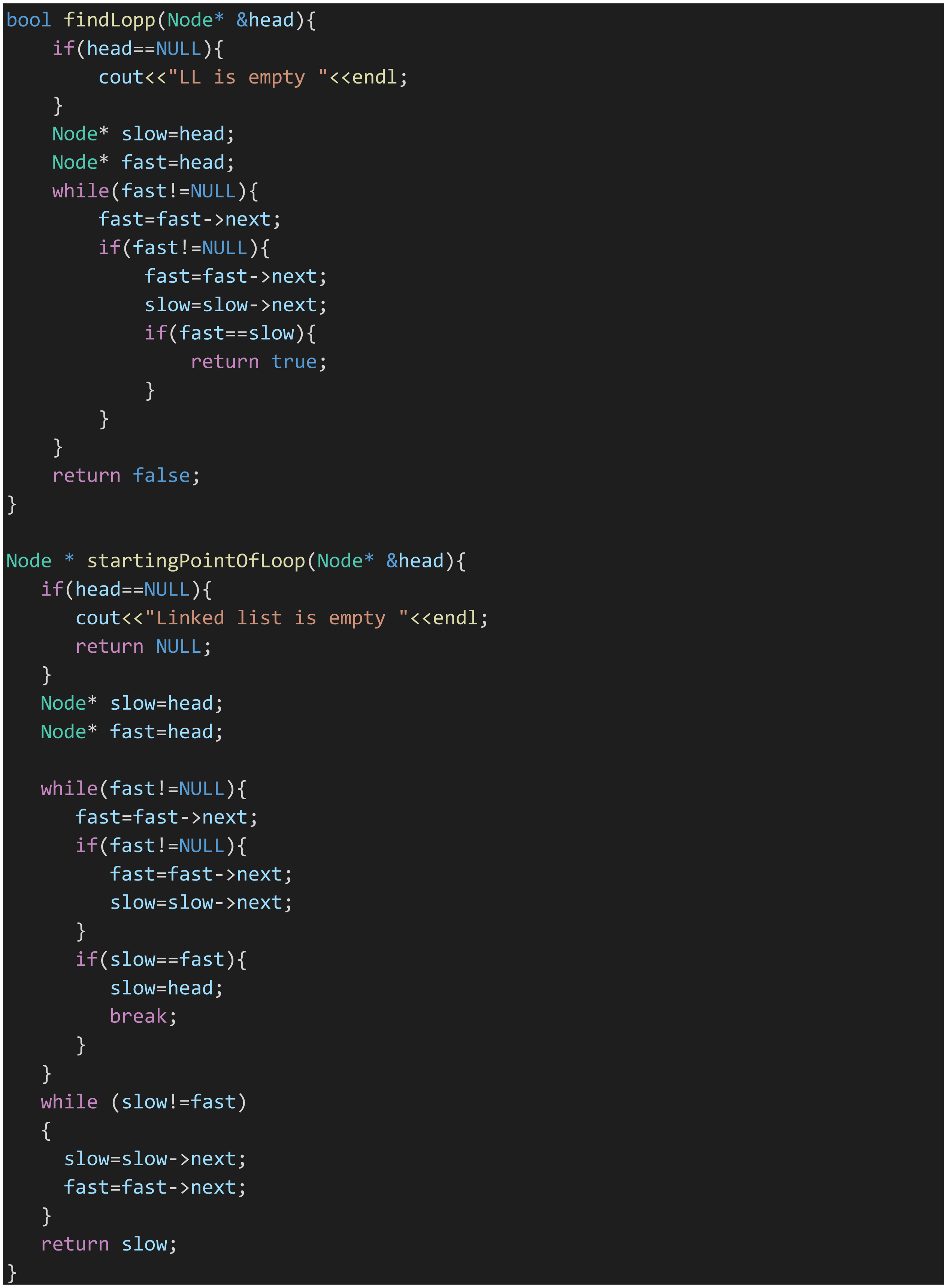
```
Node * startingPointOfLoop(Node* &head){
   if(head==NULL){
      cout<<"Linked list is empty "<<endl;
      return NULL;
   }
   Node* slow=head;
   Node* fast=head;

   while(fast!=NULL){
      fast=fast->next;
      if(fast!=NULL){
         fast=fast->next;
         slow=slow->next;
      }
      if(slow==fast){
         slow=head;
         break;
      }
   }
   while (slow!=fast)
   {
     slow=slow->next;
     fast=fast->next;
   }
   return slow;
}

int main(){
   Node* head=new Node(10);
   Node* seconde=new Node(20);
   Node* third=new Node(30);
   Node* fourth=new Node(40);
   Node* fifth=new Node(50);
   Node* sixth=new Node(60);
   Node* seventh=new Node(70);
   Node* eight=new Node(80);

   head->next=seconde;
   seconde->next=third;
   third->next=fourth;
   fourth->next=fifth;
   fifth->next=sixth;
   sixth->next=seventh;
   seventh->next=eight;
   eight->next=fifth;
   //print(head);
   cout<<"loop is present or not :"<<findLopp(head)<<endl;
   cout<<"find out looping point :"<<startingPointOfLoop(head)->data;
    return 0;
}
```



```
PS E:\C++Code> g++ .\f80.cpp
PS E:\C++Code> .\a.exe
loop is present or not :1
find out looping point :50
```

Q) remove starting loop point using Linked List (Important question)

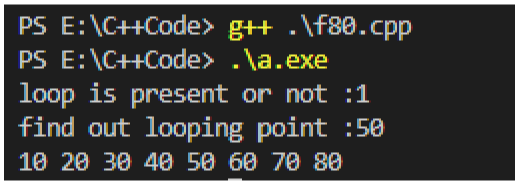
```
#include<iostream>
#include<vector>

using namespace std;

class Node{
    public:
   int data;
   Node* next;
   Node* prev;

   Node(){
      this->data=0;
      this->next=NULL;
      this->prev=NULL;
   }

   Node(int data){
      this->data=data;
      this->next=NULL;
      this->prev=NULL;
   }
};

int  getLength(Node* head){
   Node* temp =head;
   int i=0;
   while (temp!=NULL)
   {
      temp=temp->next;
      i++;
   }
   return i;
}

void print(Node* &head){
   Node* temp =head;
    while (temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->next;
   }
}
```



```
bool findLopp(Node* &head){
    if(head==NULL){
        cout<<"LL is empty "<<endl;
    }
    Node* slow=head;
    Node* fast=head;
    while(fast!=NULL){
        fast=fast->next;
        if(fast!=NULL){
            fast=fast->next;
            slow=slow->next;
            if(fast==slow){
                return true;
            }
        }
    }
    return false;
}

Node * startingPointOfLoop(Node* &head){
   if(head==NULL){
      cout<<"Linked list is empty "<<endl;
      return NULL;
   }
   Node* slow=head;
   Node* fast=head;

   while(fast!=NULL){
      fast=fast->next;
      if(fast!=NULL){
         fast=fast->next;
         slow=slow->next;
      }
      if(slow==fast){
         slow=head;
         break;
      }
   }
   while (slow!=fast)
   {
     slow=slow->next;
     fast=fast->next;
   }
   return slow;
}
```

```
Node * removeLoop(Node* &head){
   if(head==NULL){
      cout<<"Linked list is empty "<<endl;
      return NULL;
   }
   Node* slow=head;
   Node* fast=head;

   while(fast!=NULL){
      fast=fast->next;
      if(fast!=NULL){
         fast=fast->next;
         slow=slow->next;
      }
      if(slow==fast){
         slow=head;
         break;
      }
   }
   Node* prev=fast;
   while (slow!=fast)
   {
     prev=fast;
     slow=slow->next;
     fast=fast->next;
   }
   prev->next=NULL;
   return slow;
}
```

```cpp
int main(){
   Node* head=new Node(10);
   Node* seconde=new Node(20);
   Node* third=new Node(30);
   Node* fourth=new Node(40);
   Node* fifth=new Node(50);
   Node* sixth=new Node(60);
   Node* seventh=new Node(70);
   Node* eight=new Node(80);

   head->next=seconde;
   seconde->next=third;
   third->next=fourth;
   fourth->next=fifth;
   fifth->next=sixth;
   sixth->next=seventh;
   seventh->next=eight;
   eight->next=fifth;

   //print(head);
   cout<<"loop is present or not :"<<findLopp(head)<<endl;
   cout<<"find out looping point :"<<startingPointOfLoop(head)->data<<endl;
   removeLoop(head);
   print(head);
   return 0;
}
```

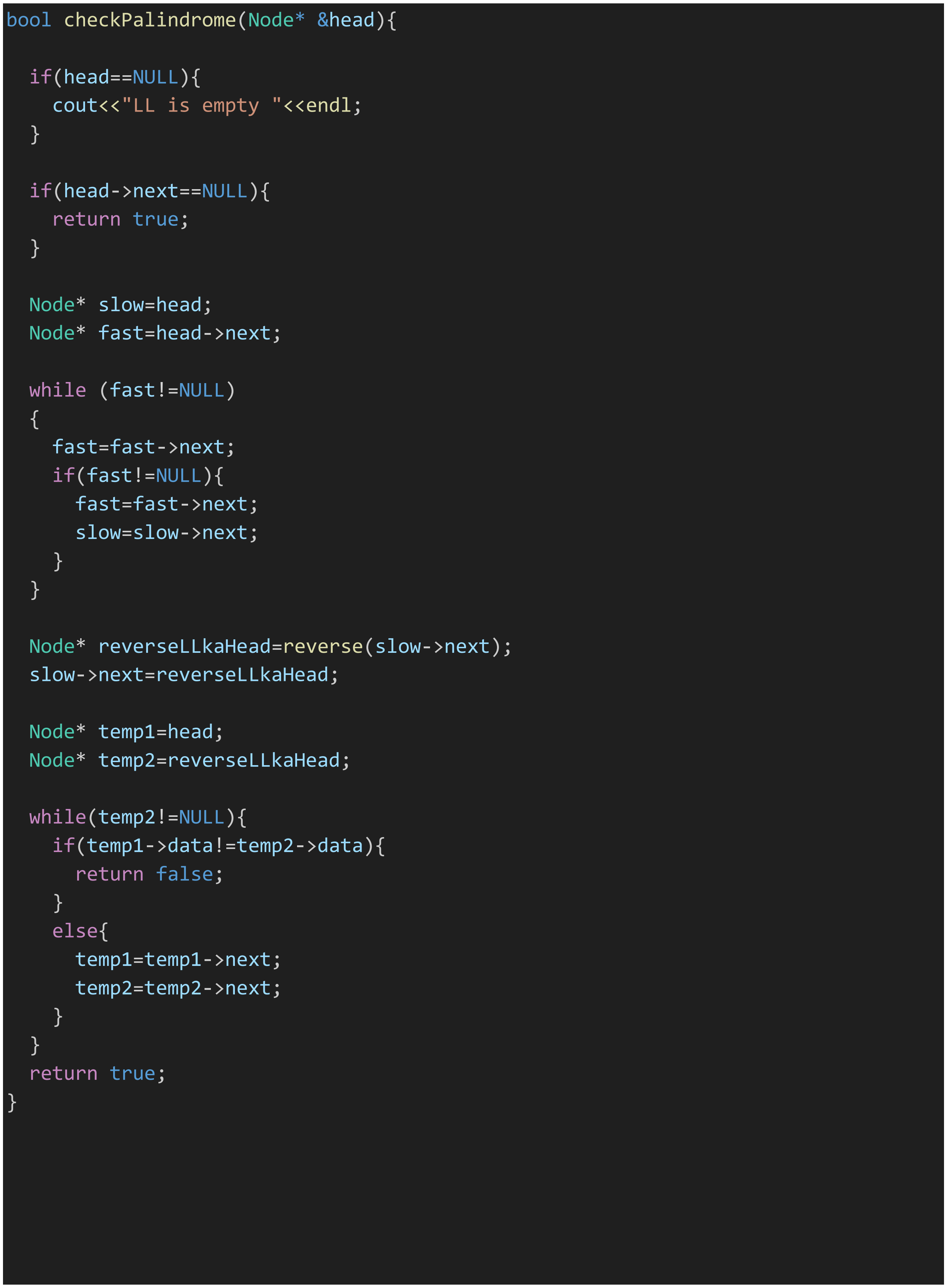


```
PS E:\C++Code> g++ .\f80.cpp
PS E:\C++Code> .\a.exe
loop is present or not :1
find out looping point :50
10 20 30 40 50 60 70 80
```

LL Class- 4

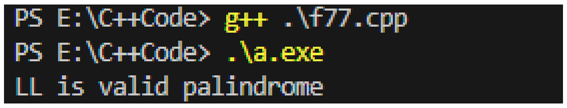
Q) check Linked list is palindrome or not

```
#include <iostream>

using namespace std;

class Node{
  public:
  int data;
  Node* next;

  Node(){
    this->data=0;
    this->next=NULL;
  }
  Node(int data){
    this->data=data;
    this->next=NULL;
  }

};

Node* reverse(Node* & head){
  Node* prev=NULL;
  Node* curr=head;
  Node * next=curr->next;

  while(curr!=NULL){
    next=curr->next;
    curr->next=prev;
    prev=curr;
    curr=next;
  }
  return prev;
}
```



```cpp
bool checkPalindrome(Node* &head){

  if(head==NULL){
    cout<<"LL is empty "<<endl;
  }

  if(head->next==NULL){
    return true;
  }

  Node* slow=head;
  Node* fast=head->next;

  while (fast!=NULL)
  {
    fast=fast->next;
    if(fast!=NULL){
      fast=fast->next;
      slow=slow->next;
    }
  }

  Node* reverseLLkaHead=reverse(slow->next);
  slow->next=reverseLLkaHead;

  Node* temp1=head;
  Node* temp2=reverseLLkaHead;

  while(temp2!=NULL){
    if(temp1->data!=temp2->data){
      return false;
    }
    else{
      temp1=temp1->next;
      temp2=temp2->next;
    }
  }
  return true;
}
```

```cpp
int main()
{
  Node* head=new Node(10);
  Node* seconde=new Node(20);
  Node* third=new Node(30);
  Node* fourth=new Node(30);
  Node* fifth=new Node(20);
  Node* sixth=new Node(10);

  head->next=seconde;
  seconde->next=third;
  third->next=fourth;
  fourth->next=fifth;
  fifth->next=sixth;

  bool isPalindrome=checkPalindrome(head);

  if(isPalindrome){
    cout<<"LL is valid palindrome "<<endl;
  }
  else{
     cout<<"LL is NOT valid palindrome "<<endl;
  }
  return 0;
}
```



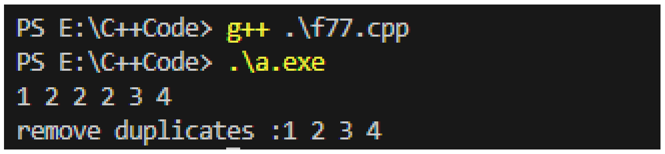
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
LL is valid palindrome
```

Q) remove duplicates using linked list

```
#include <iostream>

using namespace std;

class Node{

  public:
  int data;
  Node* next;

  Node(){
    this->data=0;
    this->next=NULL;
  }
  Node(int data){
    this->data=data;
    this->next=NULL;
  }

};

void print(Node* &head){
  if(head==NULL){
    cout<<"empty linked list "<<endl;
    return;
  }
  if(head->next==NULL){
    cout<<"single linked list "<<endl;
    return;
  }
  Node* temp=head;
  while (temp!=NULL)
  {
    cout<<temp->data<<" ";
    temp=temp->next;
  }

}
```

```
void removeDuplicate(Node* &head){

   if(head==NULL){
    cout<<"empty linked list "<<endl;
    return;
  }

  if(head->next==NULL){
    cout<<"single linked list "<<endl;
    return;
  }

  Node* curr=head;

  while(curr!=NULL){

    if((curr->next!=NULL) && (curr->data==curr->next->data)){
      Node* temp=curr->next;

      curr->next=curr->next->next;

      temp->next=NULL;
      delete temp;
    }
    else{
      curr=curr->next;
    }
  }
}
```

```cpp
int main()
{
  Node* head=new Node(1);
  Node* seconde=new Node(2);
  Node* third=new Node(2);
  Node* fourth=new Node(2);
  Node* fifth=new Node(3);
  Node* sixth=new Node(4);

  head->next=seconde;
  seconde->next=third;
  third->next=fourth;
  fourth->next=fifth;
  fifth->next=sixth;

  print(head);
  cout<<endl;
  removeDuplicate(head);
  cout<<"remove duplicates :";
  print(head);
  return 0;
}
```



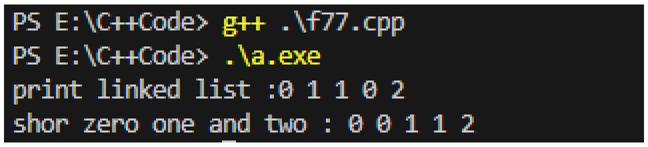
```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
1 2 2 2 3 4
remove duplicates :1 2 3 4
```

Q) sort zero ,one and two using linked list

```
#include <iostream>
#include <vector>
#include<algorithm>

using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(){
        this->data=0;
        this->next=NULL;
    }

      Node(int data){
        this->data=data;
        this->next=NULL;
    }
    ~Node(){

    }
};

void print(Node* head){
    Node* temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }

}
```

```
void sortZeroOneTwo(Node* &head){

  int zero=0;
  int one=0;
  int two=0;

  Node* temp=head;
  while(temp!=NULL){
    if(temp->data==0){
      zero++;
    }
    else if(temp->data==1){
      one++;
    }
    else if(temp->data==2){
      two++;
    }
    temp=temp->next;
  }

  temp=head;
  while (zero--)
  {
    temp->data=0;
    temp=temp->next;
  }

  while (one--)
  {
    temp->data=1;
    temp=temp->next;
  }

   while (two--)
  {
    temp->data=2;
    temp=temp->next;
  }

}
```

```
int main()
{
  Node* head=new Node(0);
  Node* seconde=new Node(1);
  Node* third=new Node(1);
  Node* fourth=new Node(0);
  Node* fifth=new Node(2);


  head->next=seconde;
  seconde->next=third;
  third->next=fourth;
  fourth->next=fifth;
  cout<<"print linked list :";
  print(head);

  sortZeroOneTwo(head);
  cout<<endl;
  cout<<"shor zero one and two : ";
  print(head);
  return 0;
}
```



```
PS E:\C++Code> g++ .\f77.cpp
PS E:\C++Code> .\a.exe
print linked list :0 1 1 0 2
shor zero one and two : 0 0 1 1 2
```

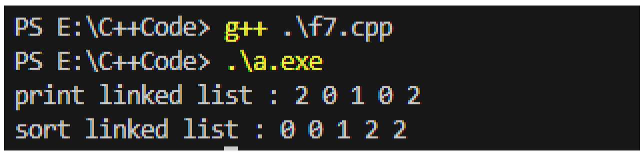
Q) sort 0 ,1 ,2 using linked list (Important question)

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

class Node
{
public:
    int data;
    Node *next;

    Node()
    {
        this->data = 0;
        this->next = NULL;
    }

    Node(int data)
    {
        this->data = data;
        this->next = NULL;
    }
    ~Node()
    {
    }
};

void print(Node *head)
{
    Node *temp = head;
    while (temp != NULL)
    {
        cout << temp->data << " ";
        temp = temp->next;
    }
}
```

```cpp
Node *sortFunction(Node * &head)
{
    if(head==NULL){

        cout<<"linked list is empty "<<endl;
        return NULL;
    }

    if(head->next==NULL){
        cout<<"one data is present "<<endl;
        return head;
    }

    Node *zeroHead = new Node(-1);
    Node *zeroTail = zeroHead;

    Node *oneHead = new Node(-1);
    Node *oneTail = oneHead;

    Node *twoHead = new Node(-1);
    Node *twoTail = twoHead;

    Node *curr = head;

    while (curr != NULL)
    {
        if (curr->data == 0)
        {
            Node *temp = curr;
            curr = curr->next;
            temp->next = NULL;

            zeroTail->next = temp;
            zeroTail = temp;
        }
        else if (curr->data == 1)
        {
            Node *temp = curr;
            curr = curr->next;
            temp->next = NULL;

            oneTail->next = temp;
            oneTail = temp;
        }
```

```
      else if (curr->data == 2)
        {
            Node *temp = curr;
            curr = curr->next;
            temp->next = NULL;

            twoTail->next = temp;
            twoTail = temp;
        }
    }

    Node *temp = oneHead;
    oneHead = oneHead->next;
    temp->next = NULL;
    delete temp;

    temp = twoHead;
    twoHead = twoHead->next;
    temp->next = NULL;
    delete temp;

    if (oneHead != NULL)
    {
        zeroTail->next = oneHead;
        if (twoHead != NULL)
        {
            oneTail->next = twoHead;
        }
    }
    else
    {
        if (twoHead != NULL)
        {
            zeroTail->next = twoHead;
        }
    }
    temp = zeroHead;
    zeroHead = zeroHead->next;
    temp->next = NULL;
    delete temp;

    return zeroHead;
}
```

```cpp
int main()
{

    Node *head = new Node(2);
    Node *seconde = new Node(0);
    Node *third = new Node(1);
    Node *fourth = new Node(0);
    Node *fifth = new Node(2);

    head->next = seconde;

    seconde->next = third;
    third->next = fourth;
    fourth->next = fifth;

    cout << "print linked list : ";
    print(head);
    cout << endl;

    cout<<"sort linked list : ";
    Node* temp=NULL;
    head=sortFunction(head);
    print(head);
    return 0;
}
```

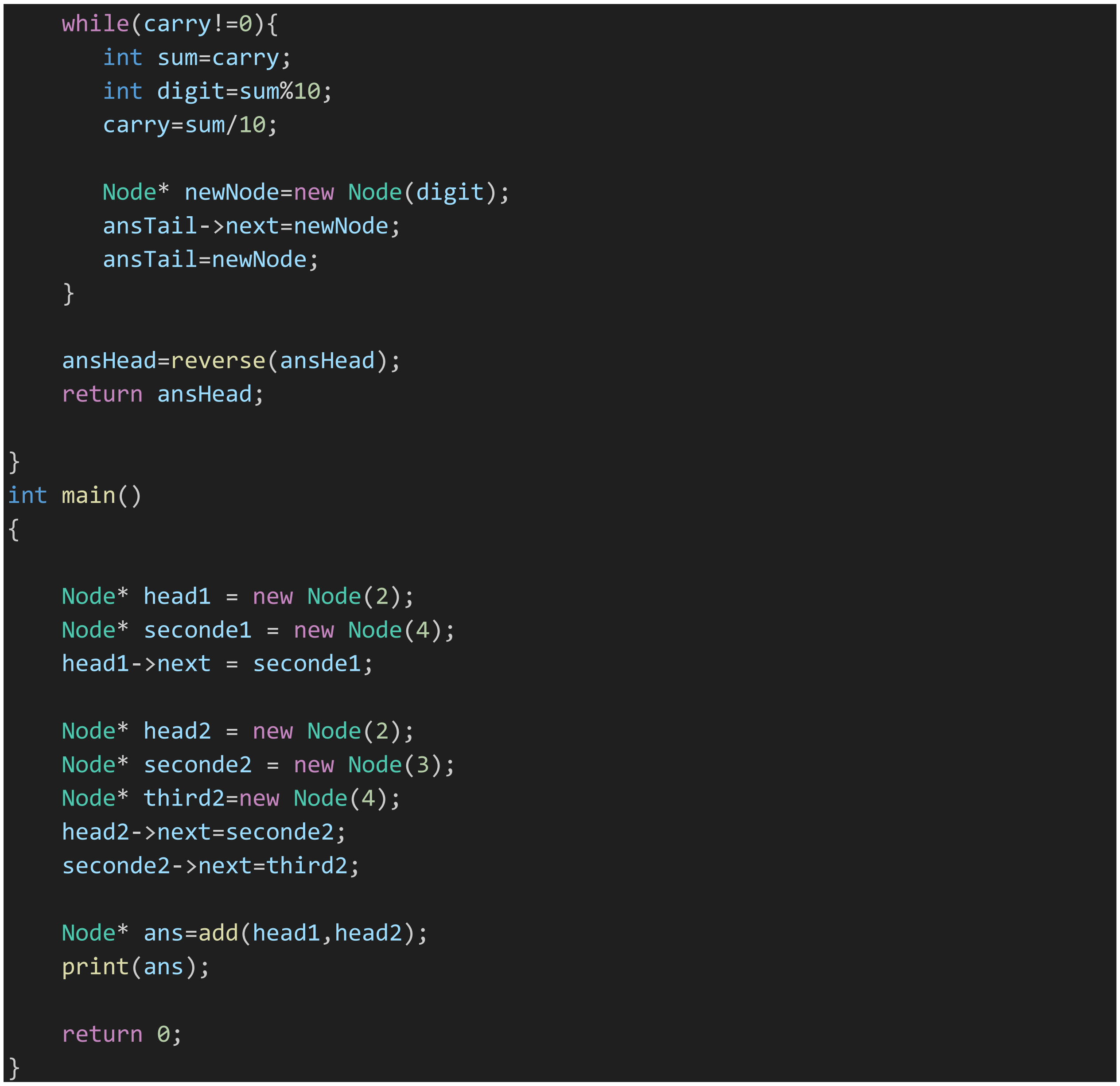


```
PS E:\C++Code> g++ .\f7.cpp
PS E:\C++Code> .\a.exe
print linked list : 2 0 1 0 2
sort linked list : 0 0 1 2 2
```

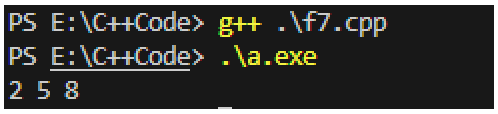
Q) Add two linked list

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;
class Node
{
public:
    int data;
    Node *next;

    Node()
    {
        this->data = 0;
        this->next = NULL;
    }

    Node(int data)
    {
        this->data = data;
        this->next = NULL;
    }
    ~Node()
    {
    }
};

void print(Node *head)
{
    Node *temp = head;
    while (temp != NULL)
    {
        cout << temp->data << " ";
        temp = temp->next;
    }
}
```

```
Node* reverse(Node* &head){
    Node* prev=NULL;
    Node* curr=head;
    Node* next=curr->next;

    while(curr!=NULL){
        next=curr->next;
        curr->next=prev;
        prev=curr;
        curr=next;
    }
    return prev;
}

Node* add(Node* &head1,Node* &head2){
    if(head1==NULL){
        return head2;
    }
    if(head2==NULL){
        return head1;
    }

    head1=reverse(head1);
    head2=reverse(head2);

    Node* ansHead=NULL;
    Node* ansTail=NULL;
    int carry=0;
    while (head1!=NULL && head2!=NULL )
    {
       int sum=carry + head1->data + head2->data;
       int digit=sum%10;
       carry=sum/10;

       Node* newNode=new Node(digit);

       if(ansHead==NULL){
        ansHead=newNode;
        ansTail=newNode;
       }
       else{
        ansTail->next=newNode;
        ansTail=newNode;
       }
       head1=head1->next;
       head2=head2->next;
    }

    while(head1!=NULL){
       int sum=carry + head1->data;
       int digit=sum%10;
       carry=sum/10;

       Node* newNode=new Node(digit);
       ansTail->next=newNode;
       ansTail=newNode;
       head1=head1->next;
    }

    while(head2!=NULL){
       int sum=carry + head2->data;
       int digit=sum%10;
       carry=sum/10;

       Node* newNode=new Node(digit);
       ansTail->next=newNode;
       ansTail=newNode;
       head2=head2->next;
    }
```



```
    while(carry!=0){
       int sum=carry;
       int digit=sum%10;
       carry=sum/10;

       Node* newNode=new Node(digit);
       ansTail->next=newNode;
       ansTail=newNode;
    }

    ansHead=reverse(ansHead);
    return ansHead;

}
int main()
{

    Node* head1 = new Node(2);
    Node* seconde1 = new Node(4);
    head1->next = seconde1;

    Node* head2 = new Node(2);
    Node* seconde2 = new Node(3);
    Node* third2=new Node(4);
    head2->next=seconde2;
    seconde2->next=third2;

    Node* ans=add(head1,head2);
    print(ans);

    return 0;
}
```



```
PS E:\C++Code> g++ .\f7.cpp
PS E:\C++Code> .\a.exe
2 5 8
```