1. Find the summation of the boundary elements for the given array. Take input from user keyboard.

1	2	3	4	5
14	15	16	17	6
13	20	9	18	7
12	11	10	9	8

Hint: max row size = m = 4 and max column size = n = 5.

```
If(i==0||i==m-1||j==0||j==n-1){
        sum= arr[i][j] + sum;
}
```

For example,

Matrix_1:

1 2 3 4 5
14 15 16 17 6
13 1 9 18 7
12 11 10 9 8

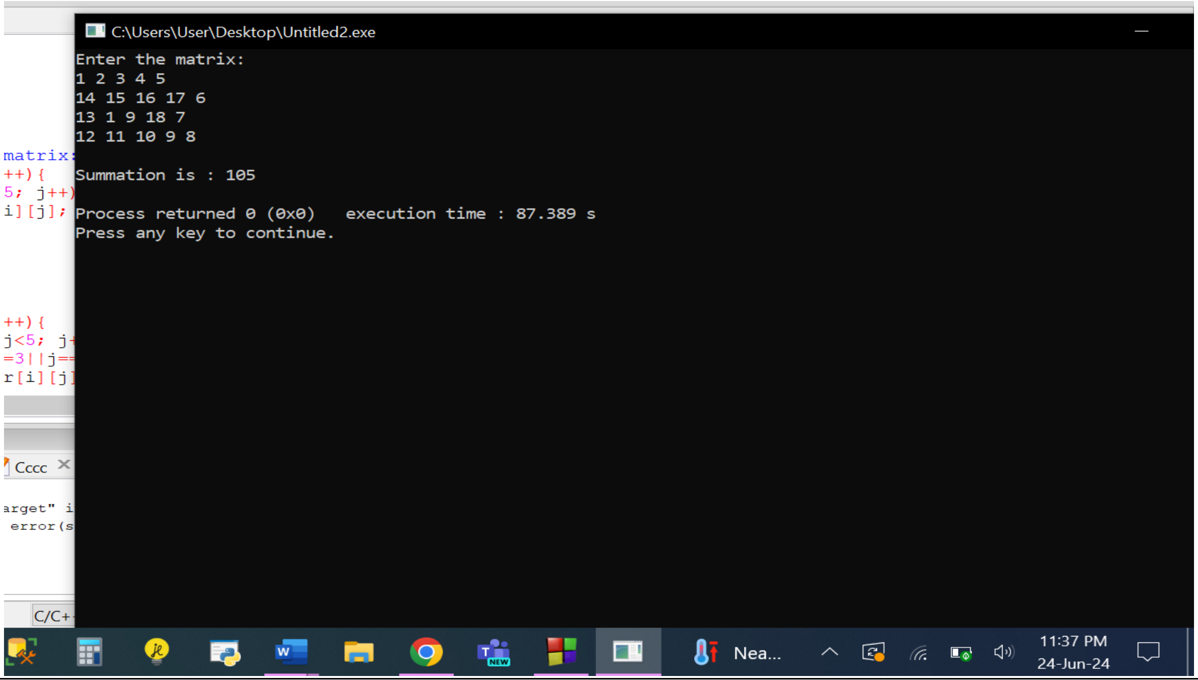
Output:

Summation is: 105

Your code here:

```
#include <iostream>
using namespace std;
int main(){
int arr[4][5];

 cout << "Enter the matrix: " << endl;
  for(int i=0; i<4; i++){
     for(int j=0; j<5; j++) {
        cin >> arr[i][j];
       }
    }
 cout << endl;

  int sum = 0;
    for(int i=0; i<4; i++){
        for(int j = 0; j<5; j++) {
            if(i==0||i==3||j==0||j==4){
                sum= arr[i][j] + sum;
      }
     }
    }
    cout << "Summation is : " << sum << endl;
}
```

Your whole Screenshot here: (Console Output):



```
Enter the matrix:
1 2 3 4 5
14 15 16 17 6
13 1 9 18 7
12 11 10 9 8

Summation is : 105

Process returned 0 (0x0)   execution time : 87.389 s
Press any key to continue.
```

2. Find the summation of the diagonal and anti-diagonal elements for the given array. Take input from user keyboard.

1	2	3	4	5
14	15	16	17	6
13	20	19	18	7
12	11	10	9	8
21	22	23	24	25

If(i==j || i+j == n+1)

For example,

Matrix_1:

1 2 3 4 5
14 15 16 17 6
13 1 19 18 7
12 11 10 9 8
21 22 23 24 25

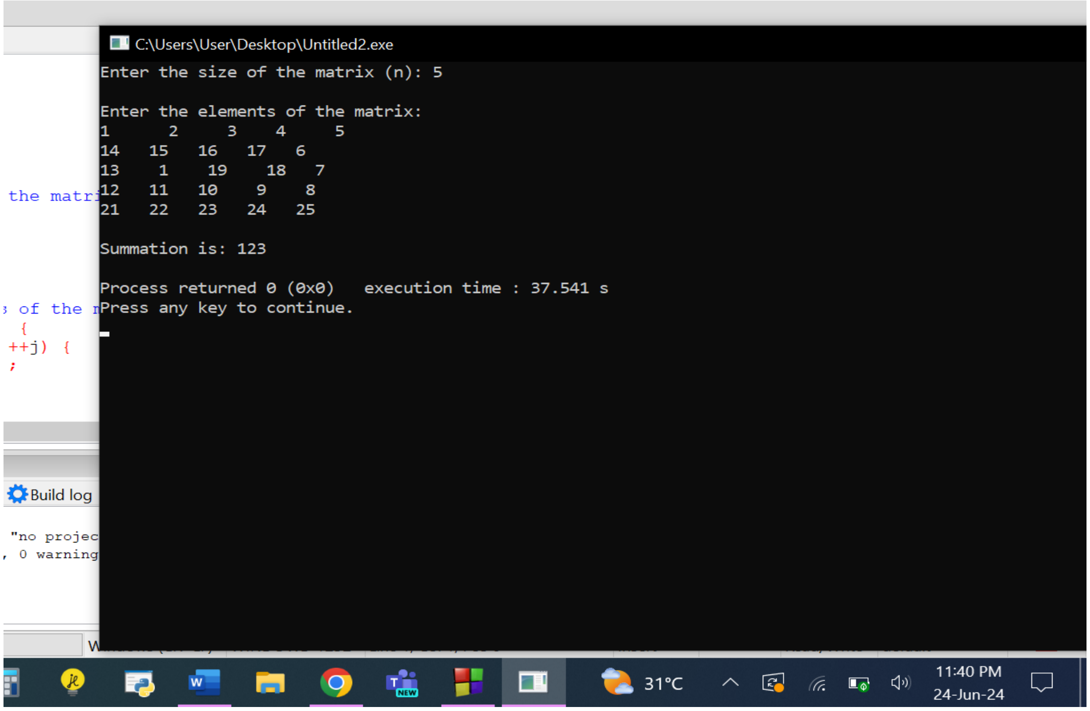
Output:

Summation is: 123

Your code here:

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

int main() {
   int n;
   cout << "Enter the size of the matrix (n): ";
   cin >> n;
   cout << endl;
   int matrix[n][n];
cout << "Enter the elements of the matrix:" << endl;;
   for (int i = 0; i < n; ++i) {
       for (int j = 0; j < n; ++j) {
           cin >> matrix[i][j];
       }
   }
cout << endl;

   int sum = 0;
   for (int i = 0; i < n; ++i) {
sum += matrix[i][i];
sum += matrix[i][n - i - 1];
   }
if (n % 2 == 1) {
       sum -= matrix[n / 2][n / 2];
   }
cout << "Summation is: " << sum << endl;
}
```

Your whole Screenshot here: (Console Output):



```
Enter the size of the matrix (n): 5

Enter the elements of the matrix:
1      2     3    4     5
14   15   16   17   6
13    1    19    18   7
12   11   10    9    8
21   22   23   24   25

Summation is: 123

Process returned 0 (0x0)   execution time : 37.541 s
Press any key to continue.
```

3. Write a code that will create custom ciphers (encoded words) on strings. Follow this procedure:
    1. Write a function named ***encode*** that takes TWO parameters, a string **s** and an integer **j**.
    2. Increase the ASCII value of the next character by 2 (leave white spaces).
    3. Perform step **(2)** throughout the string.
    4. Return the converted string from **encode** function.

For example,

Sample String (s): I am a student

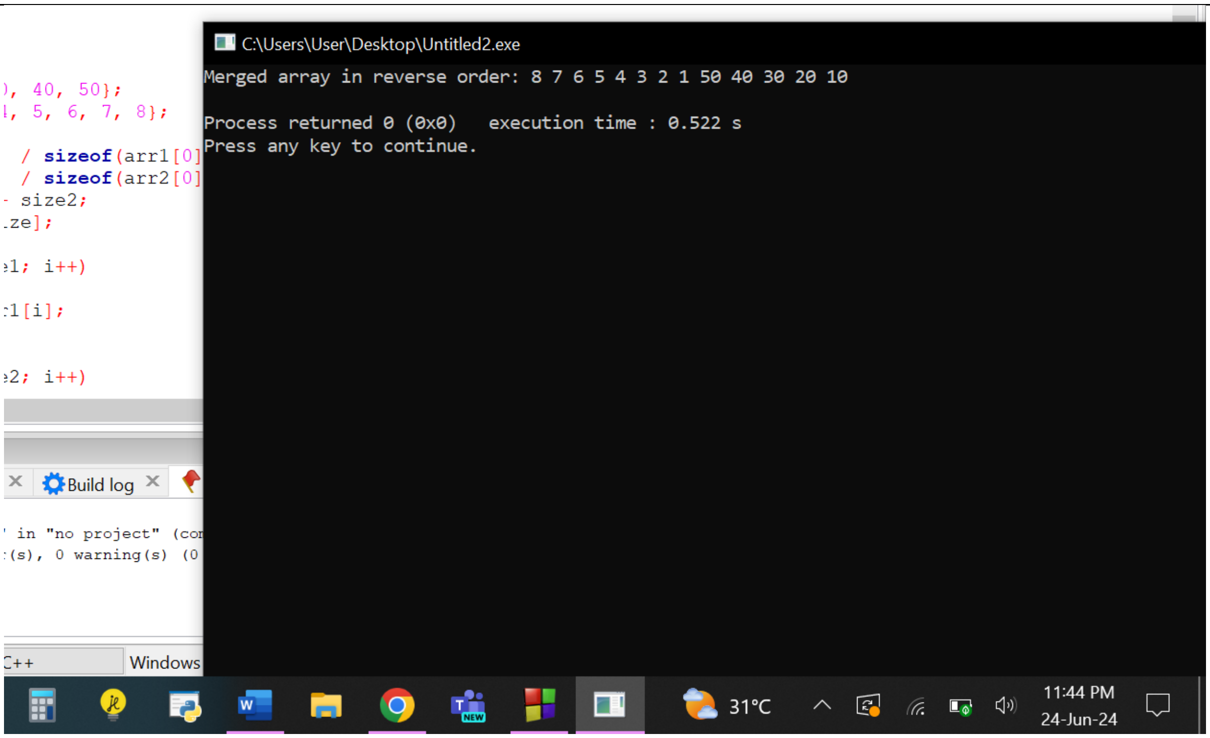
Sample Integer (j): 2

Converted String: K co c uvwfgpv

Your code here:

```
#include <iostream>
#include <string>

using namespace std;
string encode(string s, int j)
{
   for (size_t i = 0; i < s.length(); i++)
   {
      if (s[i] != ' ')
      {
         s[i] += j;
      }
   }
   return s;
}

int main()
{
   string s;
   int j;
   cout << "Enter a string: ";
   getline(cin, s);
   cout << "Enter an integer: ";
   cin >> j;

   string encodedString = encode(s, j);
   cout << "Converted String: " << encodedString << endl;
}
```

Your whole Screenshot here: (Console Output):



```
Merged array in reverse order: 8 7 6 5 4 3 2 1 50 40 30 20 10

Process returned 0 (0x0)   execution time : 0.522 s
Press any key to continue.
```

```
), 40, 50};
l, 5, 6, 7, 8};

  / sizeof(arr1[0]
  / sizeof(arr2[0]
- size2;
.ze];

ɜ1; i++)

:1[i];


ɜ2; i++)
```

```
' in "no project" (co
:(s), 0 warning(s) (0
```

4. Initialize TWO integer arrays of different sizes. Merge the input arrays and create a new array. Then print the new array in reverse order.

For example,

Array_1 = **{10,20,30,40,50}**

Array_2 = **{1,2,3,4,5,6,7,8}**

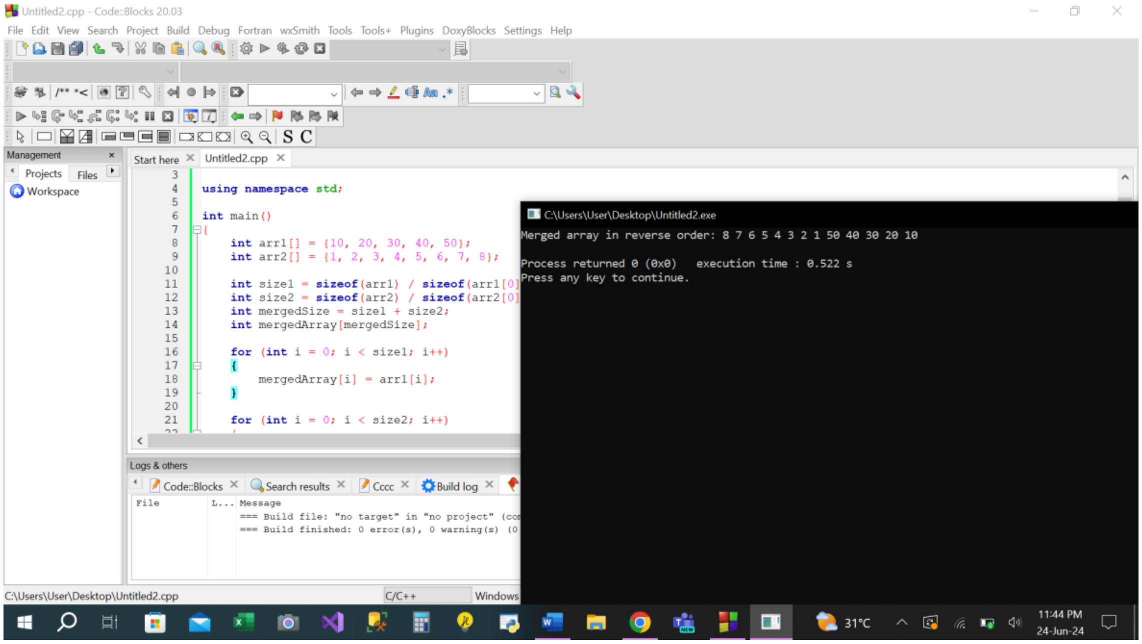
Output: **8 7 6 5 4 3 2 1 50 40 30 20 10**

Your code here:

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

int main()
{
   int arr1[] = {10, 20, 30, 40, 50};
   int arr2[] = {1, 2, 3, 4, 5, 6, 7, 8};

   int size1 = sizeof(arr1) / sizeof(arr1[0]);
   int size2 = sizeof(arr2) / sizeof(arr2[0]);
   int mergedSize = size1 + size2;
   int mergedArray[mergedSize];

   for (int i = 0; i < size1; i++)
   {
      mergedArray[i] = arr1[i];
   }

   for (int i = 0; i < size2; i++)
   {
      mergedArray[size1 + i] = arr2[i];
```

```
    }

    cout << "Merged array in reverse order: ";
    for (int i = mergedSize - 1; i >= 0; i--)
    {
        cout << mergedArray[i] << " ";
    }
    cout << endl;
}
```

Your whole Screenshot here: (Console Output):



Untitled2.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Management
Projects Files
Workspace
Start here
Untitled2.cpp
C:\Users\User\Desktop\Untitled2.exe
Logs & others
Code::Blocks
Search results
Cccc
Build log
C:\Users\User\Desktop\Untitled2.cpp
C/C++
Windows
31°C
11:44 PM
24-Jun-24

```

using namespace std;

int main()
{
    int arr1[] = {10, 20, 30, 40, 50};
    int arr2[] = {1, 2, 3, 4, 5, 6, 7, 8};

    int size1 = sizeof(arr1) / sizeof(arr1[0]
    int size2 = sizeof(arr2) / sizeof(arr2[0]
    int mergedSize = size1 + size2;
    int mergedArray[mergedSize];

    for (int i = 0; i < size1; i++)
    {
        mergedArray[i] = arr1[i];
    }

    for (int i = 0; i < size2; i++)
```

```
Merged array in reverse order: 8 7 6 5 4 3 2 1 50 40 30 20 10

Process returned 0 (0x0)   execution time : 0.522 s
Press any key to continue.
```

```
File         L... Message
                  === Build file: "no target" in "no project" (co
                  === Build finished: 0 error(s), 0 warning(s) (0
```

5. Initialize TWO integer arrays **A** and **B** of different sizes. Make a new array with the common elements between **A** and **B**. Print the new array element(s). If there is no common element, output "No common element!".

For example,

<u>Scenario 1:</u>

Array_1 = **{1,4,6,3,6,9}**

Array_2 = **{5,3,7,1,2,6}**

Output: **1 6 3**

<u>Scenario 2:</u>

Array_1 = **{1,4,6,3,6,9}**

Array_2 = **{5,8,7,12,21,63}**

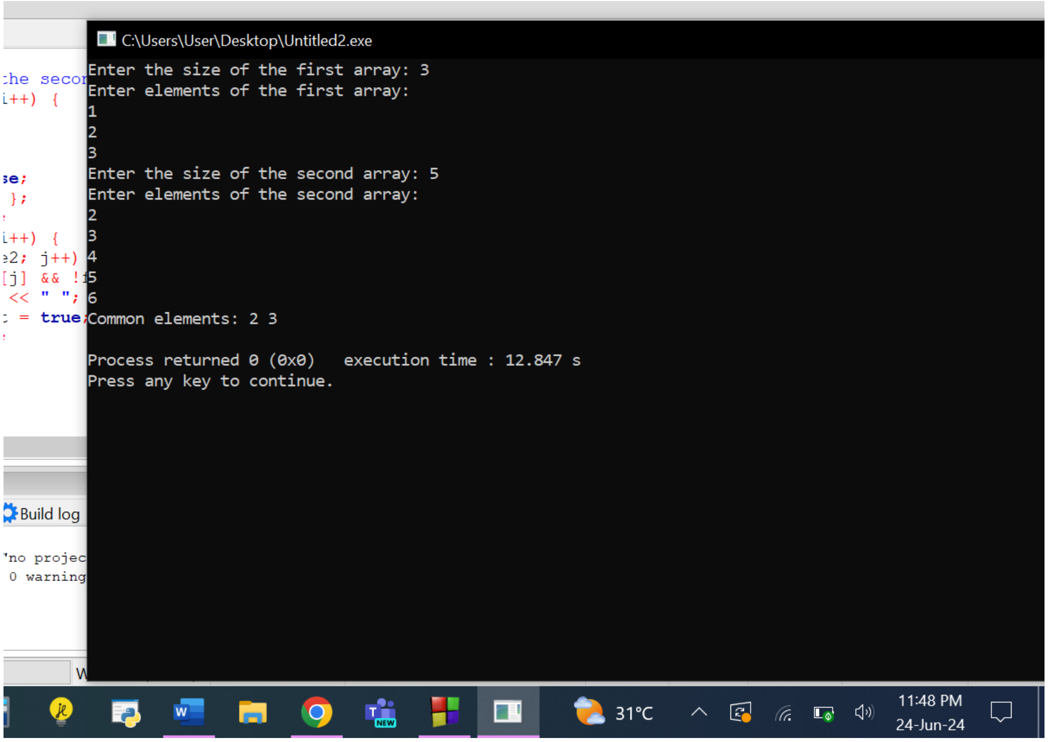
Output: **No common element!**

Your code here:

```
#include <iostream>
using namespace std;
int main() {
    int size1, size2;

    cout << "Enter the size of the first array: ";
    cin >> size1;
    int arr1[size1];
    cout << "Enter elements of the first array:\n";
    for (int i = 0; i < size1; i++) {
        cin >> arr1[i];
    }

    cout << "Enter the size of the second array: ";
    cin >> size2;
    int arr2[size2];
    cout << "Enter elements of the second array:\n";
    for (int i = 0; i < size2; i++) {
        cin >> arr2[i];
    }

    bool hasCommonElement = false;
    bool found[size1] = { false };
    cout << "Common elements: ";
    for (int i = 0; i < size1; i++) {
    for (int j = 0; j < size2; j++) {
     if (arr1[i] == arr2[j] && !found[i]) {
     cout << arr1[i] << " ";
    hasCommonElement = true;
     found[i] = true;
     break;
   }
   }
    }

    if (!hasCommonElement) {
        cout << "No common element!";
    }

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

Your whole Screenshot here: (Console Output):

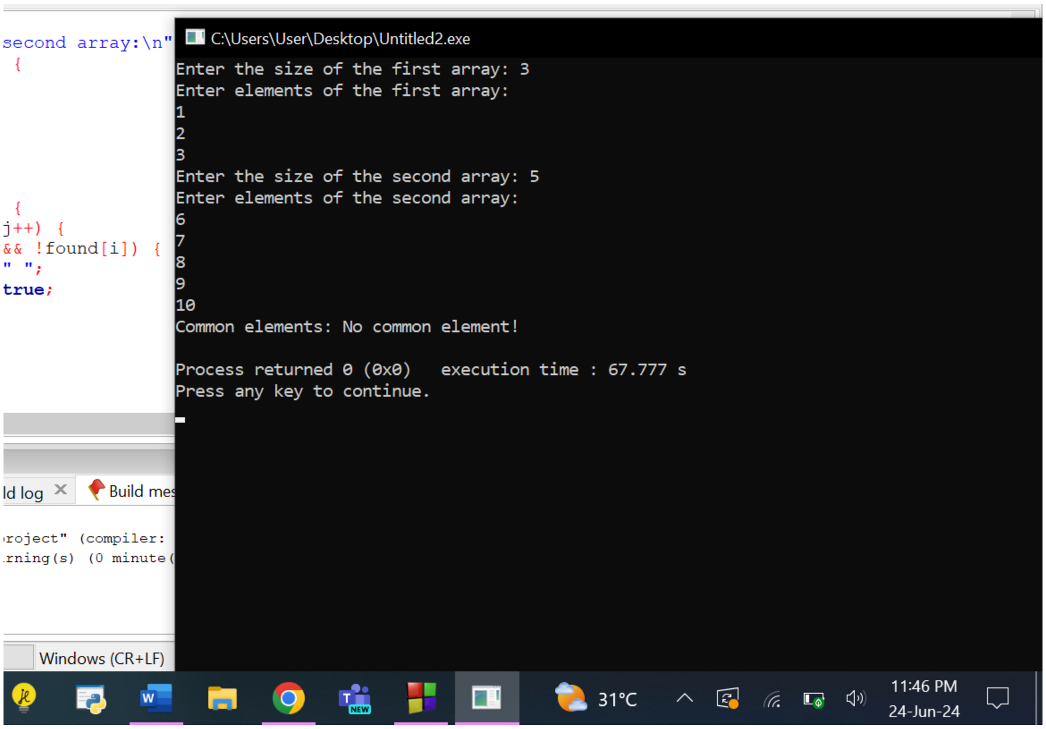
Scenerio 1



C:\Users\User\Desktop\Untitled2.exe

```
Enter the size of the first array: 3
Enter elements of the first array:
1
2
3
Enter the size of the second array: 5
Enter elements of the second array:
2
3
4
5
6
Common elements: 2 3

Process returned 0 (0x0)   execution time : 12.847 s
Press any key to continue.
```

Scenerio 2



C:\Users\User\Desktop\Untitled2.exe

```
Enter the size of the first array: 3
Enter elements of the first array:
1
2
3
Enter the size of the second array: 5
Enter elements of the second array:
6
7
8
9
10
Common elements: No common element!

Process returned 0 (0x0)   execution time : 67.777 s
Press any key to continue.
```

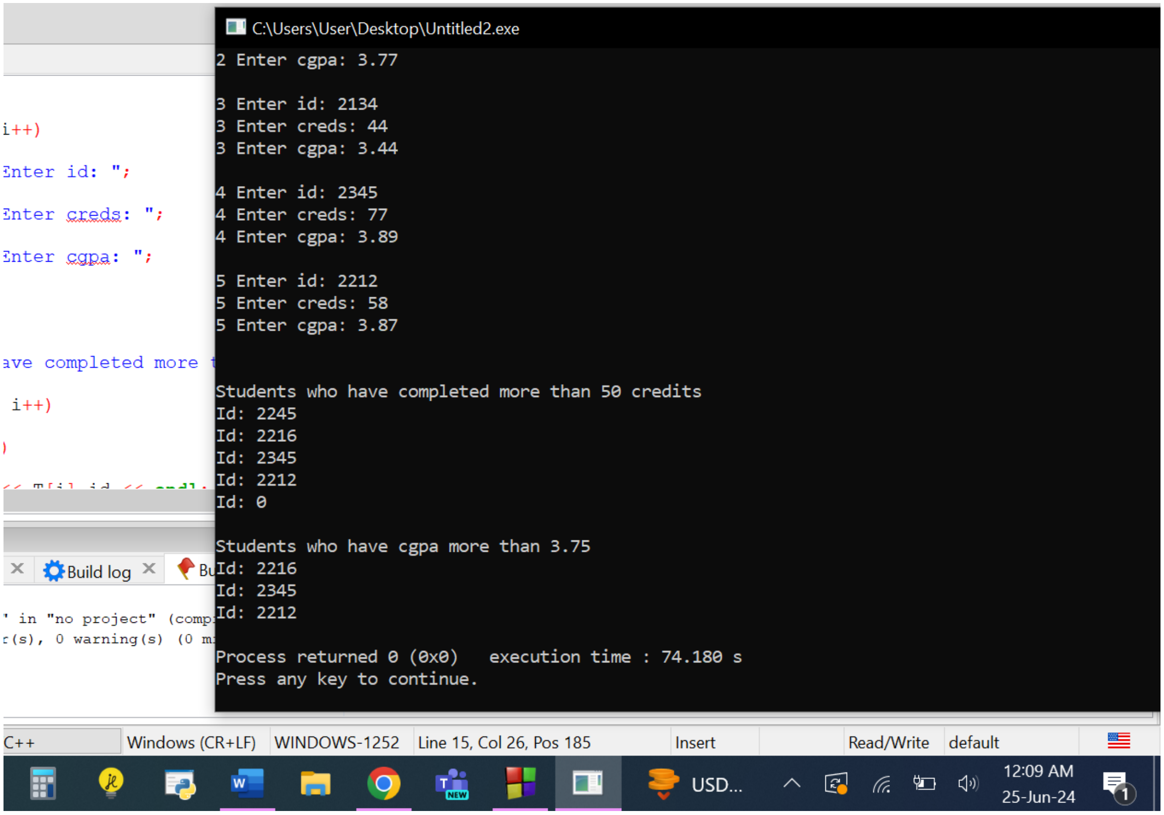
6. Write a programme with appropriate data structure to keep record of 10 students, each student will have the following information
1.Unique ID{you can use intiger for this}
2.number of credit complete3.CGPA

Print all student’s ID whose CGPA is more then 3.75
Print all student’s ID who has completed more then 50 credit!

<u>Your code here:</u>

```
#include <iostream>
using namespace std;
struct Student
{

 int id;

 int creds;

 float cgpa;
};
int main()

{

   struct Student T[10];
 for (int i = 0; i < 10; i++)

 {
 cout << i + 1 << " Enter id: ";

      cin >> T[i].id;

      cout << i + 1 << " Enter creds: ";

      cin >> T[i].creds;

      cout << i + 1 << " Enter cgpa: ";

      cin >> T[i].cgpa;

      cout << endl;

   }

   cout << endl;

   cout << "Students who have completed more than 50 credits" << endl;


 for (int i = 0; i < 10; i++)

   {
        if (T[i].creds > 50)
        {
            cout << "Id: " << T[i].id << endl;
        }
    }
  cout << endl;
  cout << "Students who have cgpa more than 3.75" << endl;
    for (int i = 0; i < 10; i++)
    {
        if (T[i].cgpa > 3.75)
        {
  cout << "Id: " << T[i].id << endl;
        }
    }
}
```

Output:



```
2 Enter cgpa: 3.77

3 Enter id: 2134
3 Enter creds: 44
3 Enter cgpa: 3.44

4 Enter id: 2345
4 Enter creds: 77
4 Enter cgpa: 3.89

5 Enter id: 2212
5 Enter creds: 58
5 Enter cgpa: 3.87


Students who have completed more than 50 credits
Id: 2245
Id: 2216
Id: 2345
Id: 2212
Id: 0

Students who have cgpa more than 3.75
Id: 2216
Id: 2345
Id: 2212

Process returned 0 (0x0)   execution time : 74.180 s
Press any key to continue.
```