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	19	18	7
12	11	10	9	8

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:

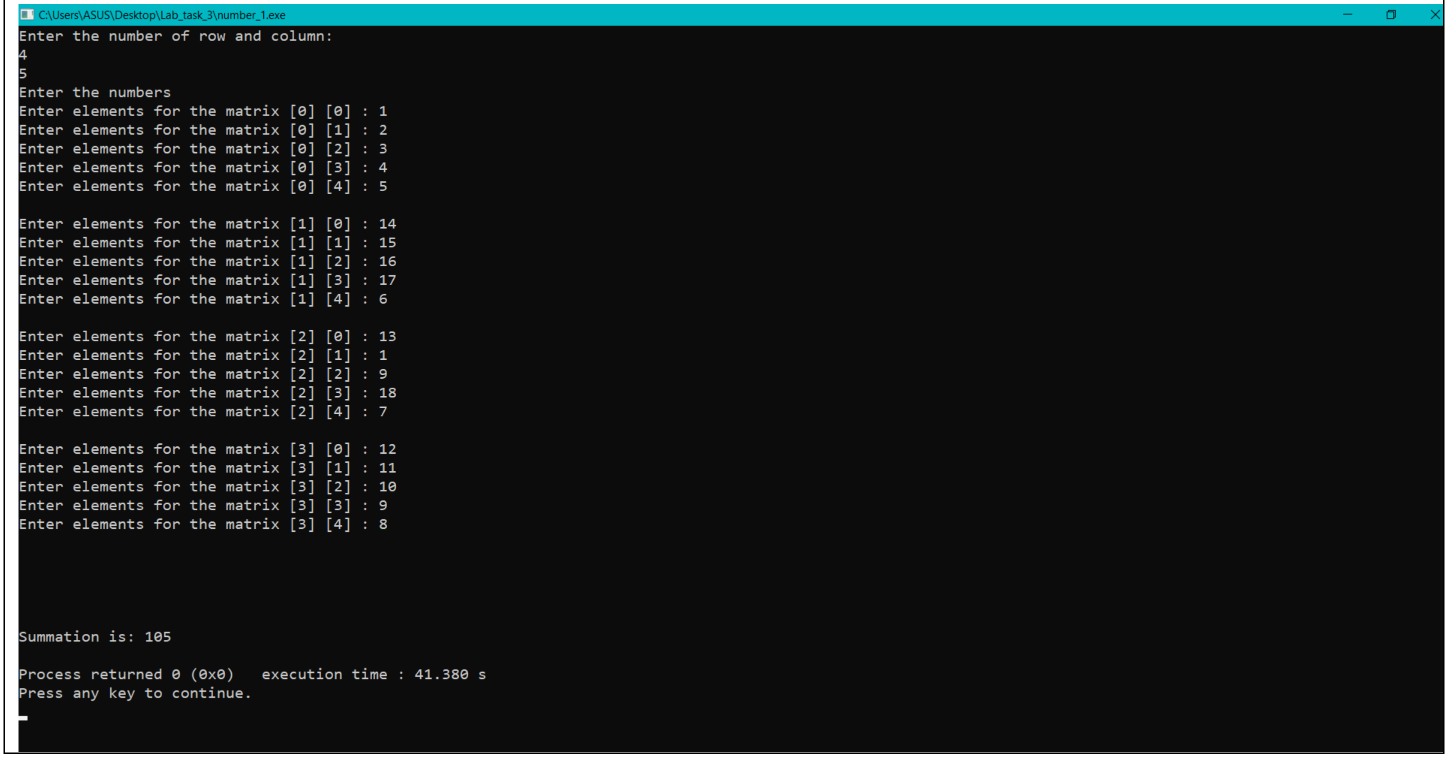
Summation is: 105

Your code here:

```
#include<iostream>
using namespace std;
int main()
{
   int p, q, sum = 0;
   int A[50][50];

   cout << "Enter the number of row and column: "<<endl;
   cin >> p >> q;

   cout << "Enter the numbers"<<endl;


   for(int i = 0; i < p; i++)
   {
      for(int j = 0; j < q; j++)
      {
         cout<< "Enter elements for the matrix ["<<i<<"] ["<<j<<"] : ";
         cin >> A[i][j];
      }
      cout<<endl;
   }
   for(int i = 0; i < p; i++)
      {
         for(int j = 0; j < q; j++)
      {

         if (i == 0 || j == 0 || i == p-1  || j == q-1)
      {
         sum = sum + A[i][j];
      }
      else
      cout << " "
          << " ";
```

```
        }
        cout << endl;
        }
        cout << "Summation is: " << sum << endl;
        return 0;
}
```

Your whole Screenshot here: (Console Output):



```
Enter the number of row and column:
4
5
Enter the numbers
Enter elements for the matrix [0] [0] : 1
Enter elements for the matrix [0] [1] : 2
Enter elements for the matrix [0] [2] : 3
Enter elements for the matrix [0] [3] : 4
Enter elements for the matrix [0] [4] : 5

Enter elements for the matrix [1] [0] : 14
Enter elements for the matrix [1] [1] : 15
Enter elements for the matrix [1] [2] : 16
Enter elements for the matrix [1] [3] : 17
Enter elements for the matrix [1] [4] : 6

Enter elements for the matrix [2] [0] : 13
Enter elements for the matrix [2] [1] : 1
Enter elements for the matrix [2] [2] : 9
Enter elements for the matrix [2] [3] : 18
Enter elements for the matrix [2] [4] : 7

Enter elements for the matrix [3] [0] : 12
Enter elements for the matrix [3] [1] : 11
Enter elements for the matrix [3] [2] : 10
Enter elements for the matrix [3] [3] : 9
Enter elements for the matrix [3] [4] : 8




Summation is: 105

Process returned 0 (0x0)   execution time : 41.380 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

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

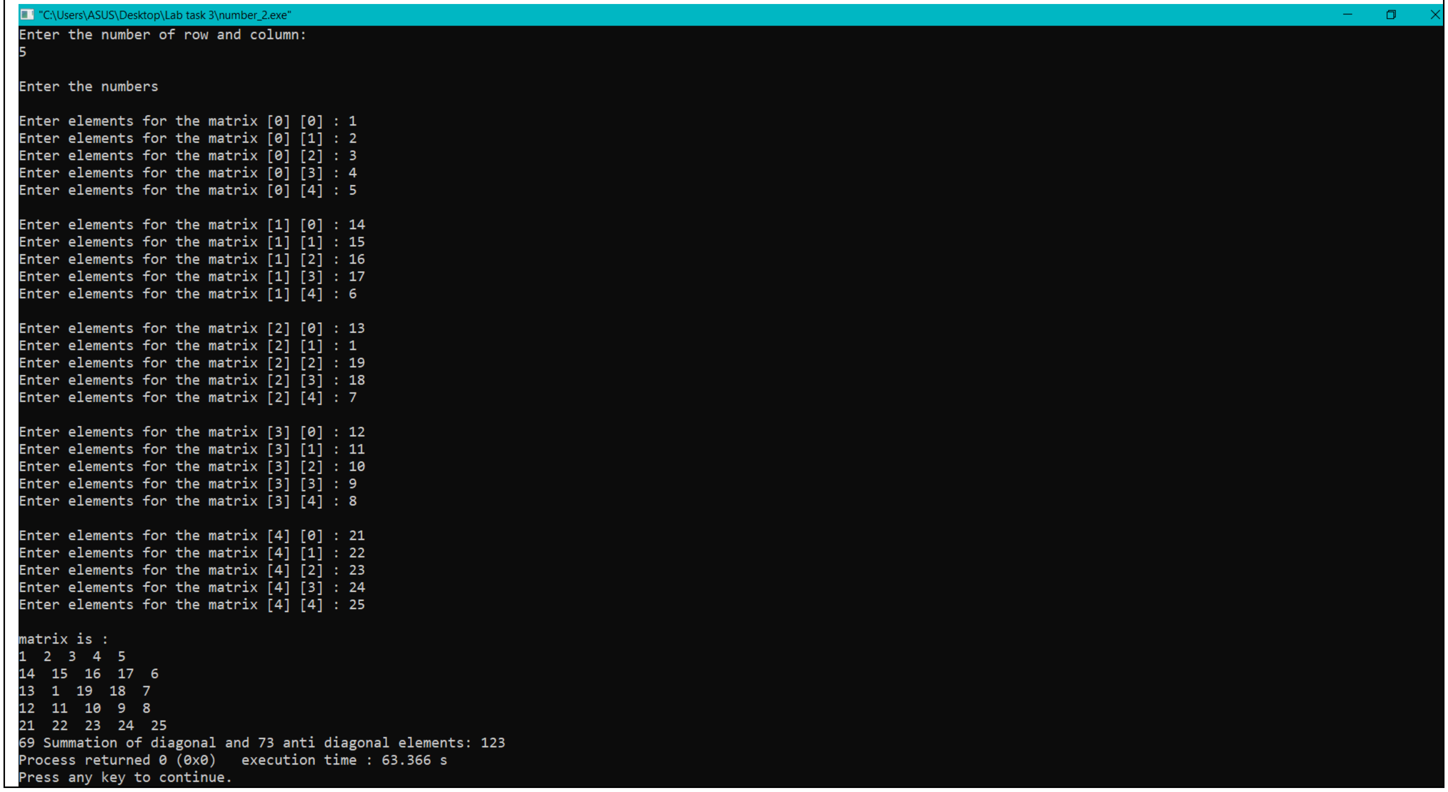
Output:

Summation is: 123

Your code here:

```
#include<iostream>
using namespace std;
int main()
{
  int num[50] [50],i,j,sum=0,sum1=0,n,s;

  cout << "Enter the number of row and column: "<<endl;
  cin >>n;

  cout << "\nEnter the numbers"<<endl<<endl;

  for(int i = 0; i < n; i++)
  {
    for(int j = 0; j < n; j++)
     {
       cout<< "Enter elements for the matrix ["<<i<<"] ["<<j<<"] : ";
        cin>>num[i] [j];
     }
     cout<<endl;
  }
  cout<< "matrix is :"<<endl;
  for(int i=0;i<n;i++)
  {
    for(int j =0;j< n;j++)
    {
      cout<<num[i][j]<<"  ";
    }
    cout<<endl;
  }
  for(int i=0;i<n;i++)
  {
    for(int j=0;j<n;j++)
    {
      if(i==j)
      {
        sum=sum+num[i][j];
      }
      if(i+j==n-1)
      {
        sum1=sum1+num[i][j];
      }
    }
  }

  s=sum+sum1-19;
  cout<<sum<<" Summation of diagonal and "<<sum1<<" anti diagonal elements: " <<s;
  return 0;

}
```

Your whole Screenshot here: (Console Output):



```
Enter the number of row and column:
5

Enter the numbers

Enter elements for the matrix [0] [0] : 1
Enter elements for the matrix [0] [1] : 2
Enter elements for the matrix [0] [2] : 3
Enter elements for the matrix [0] [3] : 4
Enter elements for the matrix [0] [4] : 5

Enter elements for the matrix [1] [0] : 14
Enter elements for the matrix [1] [1] : 15
Enter elements for the matrix [1] [2] : 16
Enter elements for the matrix [1] [3] : 17
Enter elements for the matrix [1] [4] : 6

Enter elements for the matrix [2] [0] : 13
Enter elements for the matrix [2] [1] : 1
Enter elements for the matrix [2] [2] : 19
Enter elements for the matrix [2] [3] : 18
Enter elements for the matrix [2] [4] : 7

Enter elements for the matrix [3] [0] : 12
Enter elements for the matrix [3] [1] : 11
Enter elements for the matrix [3] [2] : 10
Enter elements for the matrix [3] [3] : 9
Enter elements for the matrix [3] [4] : 8

Enter elements for the matrix [4] [0] : 21
Enter elements for the matrix [4] [1] : 22
Enter elements for the matrix [4] [2] : 23
Enter elements for the matrix [4] [3] : 24
Enter elements for the matrix [4] [4] : 25

matrix is :
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
69 Summation of diagonal and 73 anti diagonal elements: 123
Process returned 0 (0x0)   execution time : 63.366 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

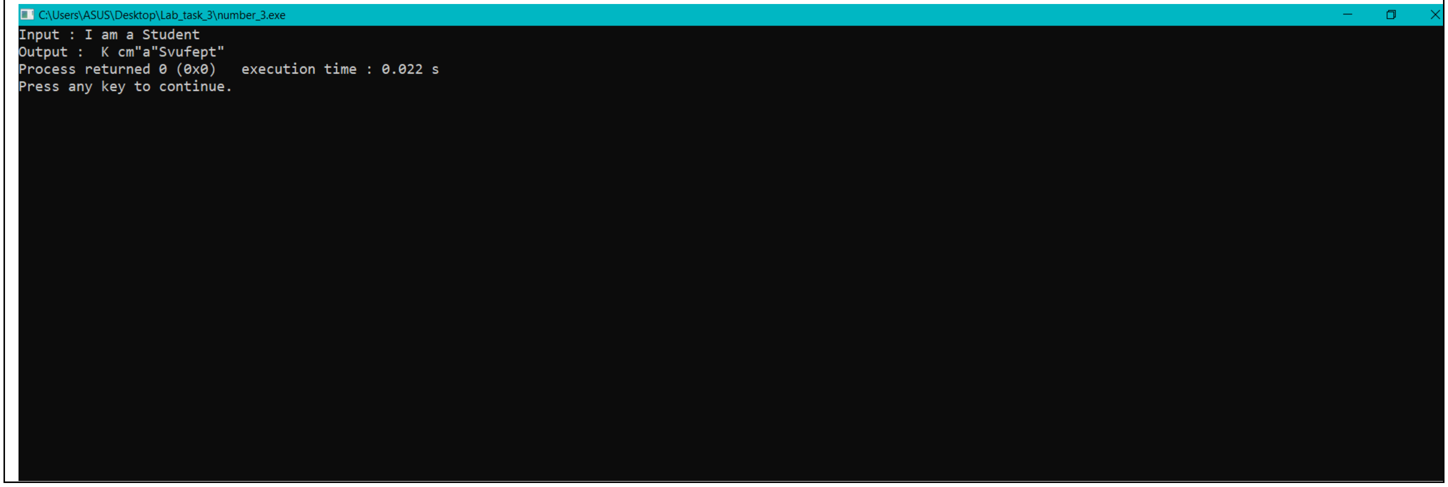
Sample Integer (j): 2

Converted String: K co c uvwfgpy

Your code here:

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

void encode(string s, int x)
{
cout<<"Input :"<<s<<endl;

for(int i=0;i<s.length();i=i+x)
   {
          if (i==0 )
           {continue;}
           s[i]=s[i]+2;
               i++;
   }
            cout<<"Output : "<<s;
   }

int main()
{



encode(" I am a Student ",1);

return 0;
}
```

Your whole Screenshot here: (Console Output):



```
Input : I am a Student
Output :  K cm"a"Svufept"
Process returned 0 (0x0)   execution time : 0.022 s
Press any key to continue.
```

4. Write a program with appropriate data structure to keep records of 10 students. Each student will have the following information:
   1. Unique ID (you can use *integer* for this)
   2. Number of Credits Completed
   3. CGPA

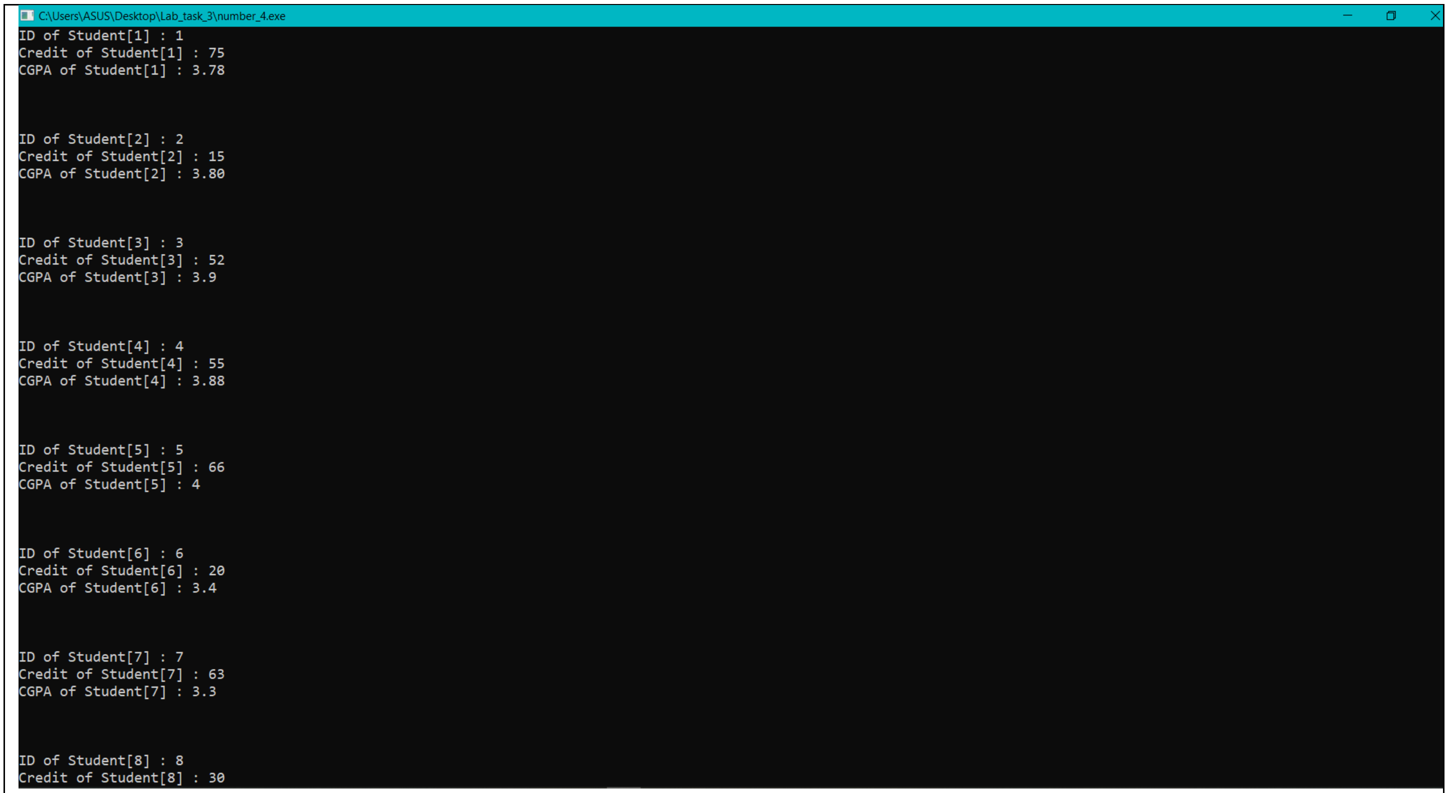
Print all the student’s ID whose CGPA is more than **3.75**.
Print all the student’s ID who has completed more than **50** credits.

Your code here:

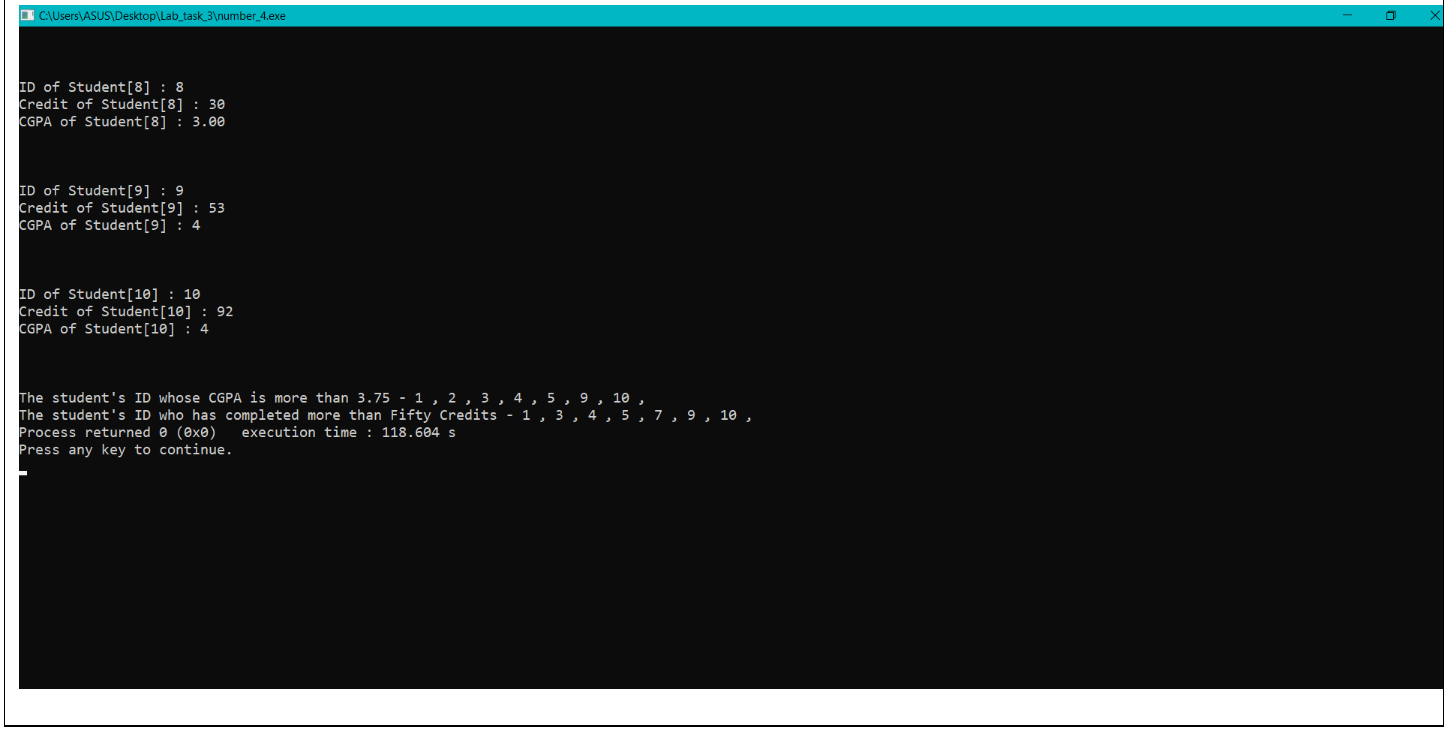
```
#include <iostream>

 using namespace std;

 struct student

 {
    int id;

    int credits;

    float cgpa;

    };
```

```
int main()

{
   student arr[10];

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

     {

       cout<<"ID of Student"<<"["<<i+1<<"]"<<" : ";
   cin>>arr[i].id;


   cout<<"Credit of Student"<<"["<<i+1<<"]"<<" : ";
   cin>>arr[i].credits;


   cout<<"CGPA of Student"<<"["<<i+1<<"]"<<" : ";
   cin>>arr[i].cgpa;
   cout<<endl;
   cout<<endl;
   cout<<endl;

   }

   cout << "The student's ID whose CGPA is more than 3.75 - ";


   for (student i : arr)
     if(i.cgpa>3.75)

     cout<<i.id<<" , ";
     cout<<endl;


     cout << "The student's ID who has completed more than Fifty Credits - ";


     for (student i : arr) if(i.credits>50) cout<<i.id<< " , ";


     return 0;

}
```

Your whole Screenshot here: (Console Output):



```
ID of Student[1] : 1
Credit of Student[1] : 75
CGPA of Student[1] : 3.78


ID of Student[2] : 2
Credit of Student[2] : 15
CGPA of Student[2] : 3.80


ID of Student[3] : 3
Credit of Student[3] : 52
CGPA of Student[3] : 3.9


ID of Student[4] : 4
Credit of Student[4] : 55
CGPA of Student[4] : 3.88


ID of Student[5] : 5
Credit of Student[5] : 66
CGPA of Student[5] : 4


ID of Student[6] : 6
Credit of Student[6] : 20
CGPA of Student[6] : 3.4


ID of Student[7] : 7
Credit of Student[7] : 63
CGPA of Student[7] : 3.3


ID of Student[8] : 8
Credit of Student[8] : 30
```



```


ID of Student[8] : 8
Credit of Student[8] : 30
CGPA of Student[8] : 3.00


ID of Student[9] : 9
Credit of Student[9] : 53
CGPA of Student[9] : 4


ID of Student[10] : 10
Credit of Student[10] : 92
CGPA of Student[10] : 4


The student's ID whose CGPA is more than 3.75 - 1 , 2 , 3 , 4 , 5 , 9 , 10 ,
The student's ID who has completed more than Fifty Credits - 1 , 3 , 4 , 5 , 7 , 9 , 10 ,
Process returned 0 (0x0)   execution time : 118.604 s
Press any key to continue.
```