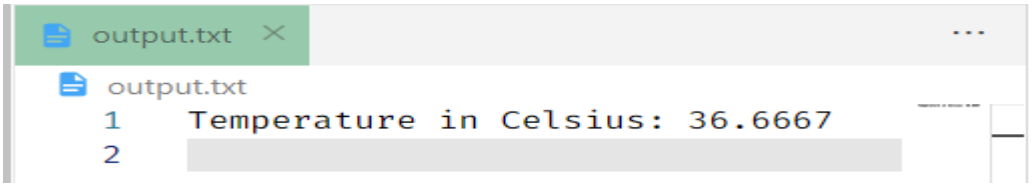
Simple Programs :

Convert fahrenheit to celsius:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    double fahrenheit;
    cout << "Enter temperature in Fahrenheit: ";
    cin >> fahrenheit;
    double celsius = (5.0 / 9.0) * (fahrenheit - 32);
    cout << "Temperature in Celsius: " << celsius << endl;

    return 0;
}
```

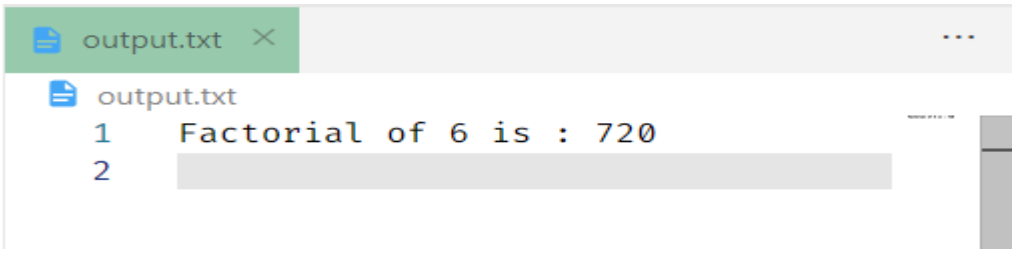
Output:



Factorial of N:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n;
    printf ("Enter a number : ");
    cin >> n;
    int factorial = 1;
    for (int i = 1; i <= n; i++) {
        factorial = factorial * i;
    }
    cout << "Factorial of " << n << " is : " << factorial << endl;

    return 0;
}
```

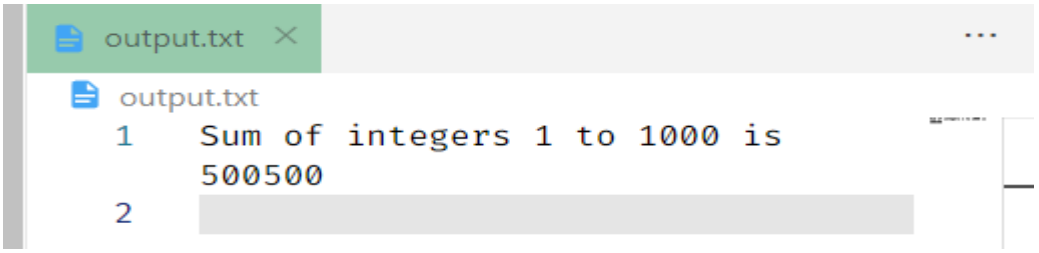
Output:



Calculate sum of integer 1…1000:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    int ans = (1000 * (1000 + 1)) / 2;
    cout << "Sum of integers 1 to 1000 is " << ans << endl;
    return 0;
}
```

Output:

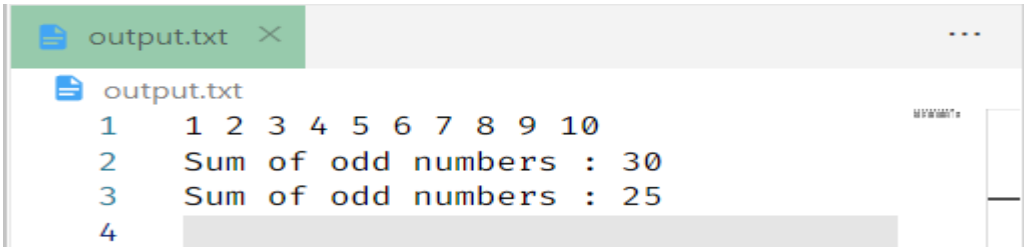


Sum of odd and even:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    int array[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    for (int i = 0; i < 10; i++) {
        cout << array[i] << " ";
    }
    int sum_odd = 0, sum_even = 0;
    for (int i = 0; i < 10; i++) {
        if (i % 2 == 0) {
            sum_even += array[i];
        } else {
```

```
            sum_odd += array[i];
        }
    }
    cout << endl;
    cout << "Sum of odd numbers : " << sum_odd << endl;
    cout << "Sum of odd numbers : " << sum_even << endl;
    return 0;
}
```

Output:

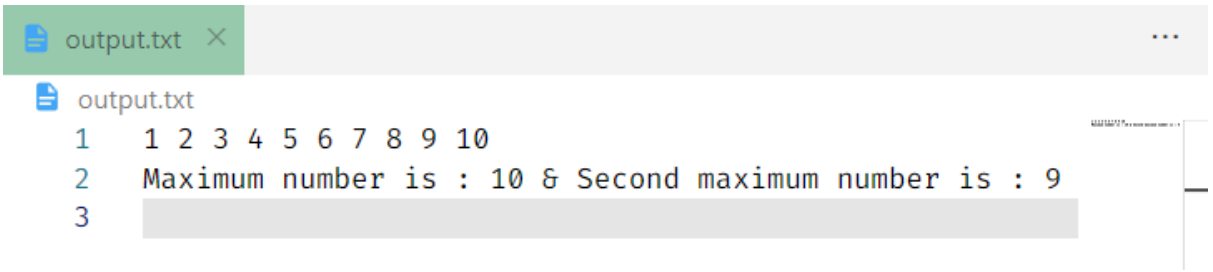


Greatest two numbers:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    int array[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    for (int i = 0; i < 10; i++) {
        cout << array[i] << " ";
    }
    cout << endl;
    int mx = INT_MIN;
    int second_mx = INT_MIN;
    for (int i = 0; i < 10; i++) {
        if (array[i] > mx) {
            second_mx = mx;
            mx = array[i];
        }
        else if (array[i] > second_mx) {
            second_mx = array[i];
        }
    }
    cout << "Maximum number is : " << mx << " & Second maximum number
is : " << second_mx << endl;

    return 0;
```

```
}
```

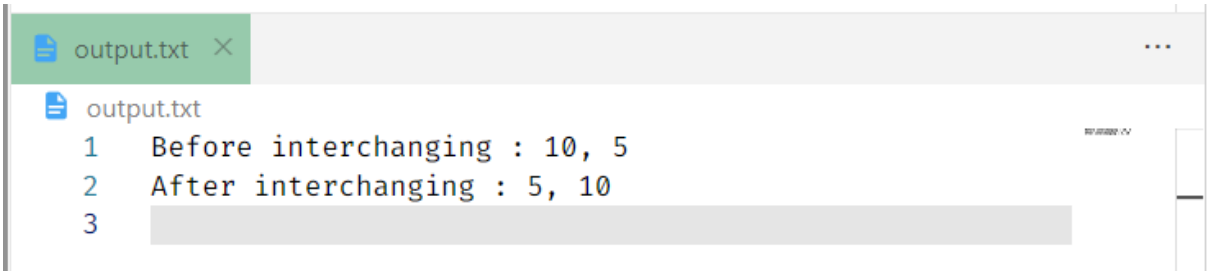
Output:



Interchange two number:

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int x = 10, y = 5;
    cout << "Before interchanging : "<< x << ", " << y << endl;
    int temporary = x;
    x = y;
    y = temporary;
    cout << "After interchanging : "<< x << ", " << y << endl;
    return 0;
}
```

Output:

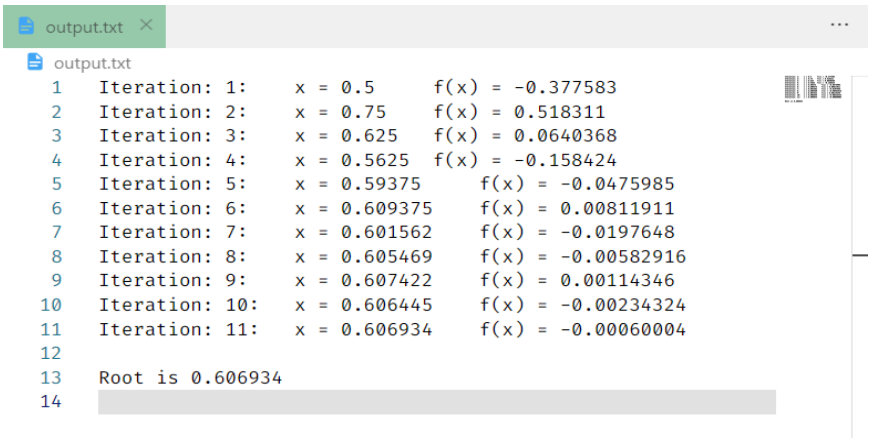


Bisection Method :

```
#include<bits/stdc++.h>
using namespace std;

#define f(x) 3 * x - cos(x) - 1
int main(){
    float x0, x1, x, f0, f1, f, e;
    int step = 1;

    up:
    cout << "Enter the first guess : ";
    cin >> x0;
    cout << "Enter the second guess : ";
    cin >> x1;
    cout << "Enter tolerable error : ";
    cin >> e;

    f0 = f(x0);
    f1 = f(x1);

    if (f0 * f1 > 0.0) {
        cout << "Incorrect initial guess" << endl;

        goto up;
    }

    do {
        x = (x0 + x1) / 2.0;
        f = f(x);

        cout << "Iteration - "  << step << ":\t x = " << setw(10) << x
<< " and f(x) = " << setw(10) << f(x) << endl;

        if (f0 * f < 0.0) {
            x1 = x;
        } else {
            x0 = x;
        }

        step++;
    } while (fabs(f) > e);
```

```
    cout << endl << "Root is : "  << x << endl;

    return 0;
}
```

Output:



Regular False Position Method :

```
#include<bits/stdc++.h>
using namespace std;

#define f(x) 3 * x - cos(x) - 1
int main(){
    float x0, x1, x, f0, f1, f, e;
    int step = 1;


    up:
    cout << "Enter the first guess : ";
    cin >> x0;
    cout << "Enter the second guess : ";
    cin >> x1;
    cout << "Enter tolerable error : ";
    cin >> e;


    f0 = f(x0);
    f1 = f(x1);
```

```
    if (f0 * f1 > 0.0) {
        cout << "Incorrect initial guess" << endl;


        goto up;
    }

    do {
        x = (x0 * f1 - x1 * f0) / (f1 - f0);
        f = f(x);

        cout << "Iteration - "  << step << ":\t x = " << setw(10) << x
<< " and f(x) = " << setw(10) << f(x) << endl;


        if (f0 * f < 0.0) {
            x1 = x;
            f1 = f;
        } else {
            x0 = x;
            f0 = f;
        }


        step++;
    } while (fabs(f) > e);

    cout << endl << "Root is : "  << x << endl;

    return 0;
}
```

Output:

output.txt

```
1   Iteration - 1:    x =    0.578085 and f(x) =  -0.103255
2   Iteration - 2:    x =    0.605959 and f(x) = -0.00408077
3   Iteration - 3:    x =    0.607057 and f(x) = -0.000159085
4
5   Root is : 0.607057
6
```

One Point / Fixed Point Iteration Method :

```
#include<bits/stdc++.h>
using namespace std;

#define f(x) 3 * x - cos(x) - 1

#define g(x) (cos(x) + 1) / 3
int main(){
    float x0, x1, e;

    int step = 1, N;
    cout << "Enter initial guess value: ";
    cin >> x0;
    cout << "Enter tolerable error : ";
    cin >> e;
    cout << "Enter maximum number of iterations: ";
    cin >> N;
    do {
        x1 = g(x0);
        cout << "Iteration - "  << step << ":\t x1 = " << setw(10) <<
x1 << " and f(x1) = " << setw(10) << f(x1) << endl;

        step++;
        if (step > N) {
            break;
        }
        x0 = x1;
    } while (fabs(f(x1)) > e);
    return 0;
}
```

Output:

output.txt

```
1   Iteration - 1:    x1 = 0.00056841 and f(x1) =    -1.99829
2   Iteration - 2:    x1 =    0.666667 and f(x1) =    0.214113
3   Iteration - 3:    x1 =    0.595296 and f(x1) = -0.0420955
4   Iteration - 4:    x1 =    0.609328 and f(x1) = 0.00794959
5   Iteration - 5:    x1 =    0.606678 and f(x1) = -0.00151366
6   Iteration - 6:    x1 =    0.607182 and f(x1) = 0.000287771
7
```

Newton Raphson Method :

```
#include<bits/stdc++.h>
using namespace std;

#define f(x) 3 * x - cos(x) - 1

#define d(x) 3 + sin(x)
int main(){
    float x0, x1, f0, f1, d0, e;

    int step = 1, N;
    up:
    cout << "Enter initial guess value: ";
    cin >> x0;
    cout << "Enter initial guess value: ";
    cin >> x1;
    cout << "Enter tolerable error : ";
    cin >> e;
    cout << "Enter maximum number of iterations: ";
    cin >> N;

    f0 = f(x0);
    f1 = f(x1);
    if (f0 * f1 > 0.0) {
        cout << "Incorrect initial guess" << endl;
        goto up;
    }

    do {
        f0 = f(x0);
        d0 = d(x0);

        if (d0 == 0.0) {
            cout << "Mathematical error" << endl;
            return 0;
        }
        x1 = x0 - (f0 / d0);
        cout << "Iteration - "  << step << ":\t x1 = " << setw(10) <<
x1 << " and f(x1) = " << setw(10) << f(x1) << endl;
        x0 = x1;
        step++;
```

```
        if (step > N) {
            break;
        }
        f1 = f(x1);
    } while (fabs(f1) > e);
    return 0;
}
```

Output: