C++ Tutorial

Rajeev Singh

The Institute of Mathematical Sciences, Chennai.

Day 1

aa_hello_world.c

```c
#include <stdio.h>

int main() {
    printf("Hello World from C\n");
    return 0;
}
```

ab_hello_world.cpp

```cpp
#include <iostream>

int main() {
    std::cout << "Hello World from C++";// << std::endl;
    return 0;
}
```

ac_hello_world.cpp

```cpp
#include <iostream>

using namespace std;

int main() {
    cout << "Hello World from C++" << endl;
    return 0;
}
```

ad_powers_of_integer.cpp

```
int main() {
    //int given_number;
    long int given_number;
    cout << "Enter an integer: ";
    cin  >> given_number;

    cout << "Given number = " << given_number << endl
         << "Square       = " << pow(given_number,2) << endl
         << "Cube         = " << pow(given_number,3) << endl
         << "Forth power  = " << pow(given_number,4) << endl;

    return 0;
}
```

ae_powers_of_real.cpp

```cpp
int main() {
    double given_number;
    //long double given_number;
    cout << "Enter a real number : ";
    cin  >> given_number;

    cout << "Given number = " << given_number << endl
         << "Square       = " << pow(given_number,2) << endl
         << "Square root  = " << pow(given_number,1./2) << endl
         << "Cube         = " << pow(given_number,3) << endl
         << "Forth power  = " << pow(given_number,4) << endl;

    return 0;
}
```

Day 2

af_pointer.cpp

```
int main() {
    int *np = NULL;
    int n = 10;

    cout << "Initial" << endl
         << "n   = " << n   << endl
         << "np  = " << np  << endl
         << "*np = " << "since np is NULL, printing *np gives segm
<< endl << endl;

    np = &n;
    cout << "After: np = &n" << endl
         << "n   = " << n   << endl
         << "np  = " << np  << endl
         << "*np = " << *np << endl << endl;
```

af_pointer.cpp

```
    *np = 22;
    cout << "After: *np = 22" << endl
         << "n   = " << n   << endl
         << "np  = " << np  << endl
         << "*np = " << *np << endl << endl;

    return 0;
}
```

ag_reference.cpp

```
int main() {
    int    n = 5;
    int & r = n;
    int    m;

    cout << "Initial" << endl
         << "n = " << n << endl
         << "r = " << r << endl
         << "m = " << m << endl << endl;

    m = r + 3;     // m == n + 3
    cout << "After: m = r + 3" << endl
         << "n = " << n << endl
         << "r = " << r << endl
         << "m = " << m << endl << endl;
```

ag_reference.cpp

```
    r = m;          // r still points to n and n == m
    cout << "After: r = m" << endl
         << "n = " << n << endl
         << "r = " << r << endl
         << "m = " << m << endl << endl;

    m = 0;          // r and n are unchanged
    cout << "After: m = 0" << endl
         << "n = " << n << endl
         << "r = " << r << endl
         << "m = " << m << endl << endl;

    int & s = m;
    r = s;          // r still points to n and n == m (== 0)
    cout << "After: r = s where s is new reference to m" << endl
         << "n = " << n << endl
         << "r = " << r << endl
         << "m = " << m << endl << endl;

    return 0;
}
```

ah_arithmetic_operators.cpp

```cpp
int main() {
    int m = 100,
        n = 200;

    cout << "Initial" << endl
         << "m = " << m << endl
         << "n = " << n << endl
         << "m + n = " << m + n << endl
         << "m - n = " << m - n << endl
         << "m * n = " << m * n << endl
         << "m / n = " << m / n << endl
         << "m % n = " << m % n << endl << endl;
```

ah_arithmetic_operators.cpp

```
//m = m + 200;
m += 200;          // both this commands are same
cout << "After: m += 200" << endl
     << "m = " << m << endl
     << "n = " << n << endl
     << "m + n = " << m + n << endl
     << "m - n = " << m - n << endl
     << "m * n = " << m * n << endl
     << "m / n = " << m / n << endl
     << "m % n = " << m % n << endl << endl;
```

ah_arithmetic_operators.cpp

```
    m++;
    cout << "After: m++" << endl
         << "m = " << m << endl
         << "n = " << n << endl
         << "m + n = " << m + n << endl
         << "m - n = " << m - n << endl
         << "m * n = " << m * n << endl
         << "m / n = " << m / n << endl
         << "m % n = " << m % n << endl << endl;

    return 0;
}
```

ai_relational_logical.cpp

```
int main() {
    int    x = 2;
    int    y = 4;
    int    z = 4;
    bool   b;

    cout << "x = " << x << endl
         << "y = " << y << endl
         << "z = " << z << endl << endl;

    // z == 4 is not tested
    b = ( x == 2 && y == 3 && z == 4 );
    cout << "b = ( x == 2 && y == 3 && z == 4 )" << endl
         << "b = " << b << endl << endl;
```

ai_relational_logical.cpp

```
    // only x == 2 is tested
    b = ( x == 2 || y == 3 || z == 4 );
    cout << "b = ( x == 2 || y == 3 || z == 4 )" << endl
         << "b = " << b << endl << endl;

    // correct, since x != 0 in "y/x"
    b = ( x != 0 && y/x > 1 );
    cout << "b = ( x != 0 && y/x > 1 )" << endl
         << "b = " << b << endl << endl;

    return 0;
}
```

Day 3

aj_blocks_scope.cpp

```
int main() {
    { // block 1
        int n1 = 1;
        double f1 = 0.0;
        cout << "Block 1 "     << endl;
        cout << "n1 = " << n1 << endl;
        cout << "f1 = " << f1 << endl;
    }

    { // block 2
        int n1 = 2;
        // n1 has value 2 in this block
        cout << "Block 2 "     << endl;
        cout << "n1 = " << n1 << endl;

        //int n1 = 5; // ERROR
    }

    return 0;
}
```

ak_scope.cpp

```
int main() {
    { // block 1
        int m, n1 = 1;
        { // block 1.1
            int n2 = 2;
            { // block 1.1.1
                m = n1 + n2;  // evaluates to m = 3
                cout << "Block 1.1.1: m = " << m << endl;
            }
        }

        { // block 1.2
            int n2 = 3;
            m = n1 + n2;       // evaluates to m = 4
            cout << "Block 1.2  : m = " << m << endl;
        }
    }

    return 0;
}
```

al_if_else.cpp

```cpp
int main() {
    int n = 1;

    if ( n > 0 )
    {
        n = n / n;
    }

    if ( n < 0 ) n += 5; // NOTE: trivial block!
    else         n -= 6;

    cout << "n = " << n << endl;

    return 0;
}
```

am_for_loop.cpp

```cpp
int main() {
    int n = 1;

    for (int i=1; i<10; i++) {
        n *= i;
        cout << "n = " << n << endl;
    }

    return 0;
}
```

an_while_loop.cpp

```
int main() {
    int n = 1;
    int i = 1;

    while (i < 10) {
        n *= i;
        i++;
        cout << "n = " << n << endl;
    }

    return 0;
}
```

ao_do_while_loop.cpp

```cpp
int main() {
    int n = 1;
    int i = 1;

    do {
        n *= i;
        i++;
        cout << "n = " << n << endl;
    } while (i < 10);

    return 0;
}
```

ap_break.cpp

```cpp
int main() {
    int n = 1;

    for ( int i = 1; i < 20; i++ ) {
        // avoid overflow
        if ( n > 21474836 )
            break;
        n *= i;
        cout << "n = " << n << endl;
    }

    return 0;
}
```

aq_break_nested_loop.cpp

```cpp
int main() {
    for ( int i = 1; i < 20; i++ ) {
        int n = 1;
        for ( int j = 1; j < i; j++ ) {
            if ( n > 21474836 )
                break;
            n *= j;
        }

        cout << "n = " << n << endl;
    }

    return 0;
}
```