Task 3

# Deep Das

Question : Differences between float and double in C++ explain with example and its

Answer :

In C++, **float** and **double** are both used to represent floating-point numbers, but they differ in their precision and range.

**Precision**: **float** is a single-precision floating-point number, which means it has 32 bits (4 bytes) to store the number, providing about 7 decimal digits of precision. On the other hand, **double** is a double-precision floating-point number, with 64 bits (8 bytes) to store the number, providing about 15-16 decimal digits of precision.

**Range**: **float** typically ranges from approximately 1.5 × 10^-45 to 3.4 × 10^38, while **double** ranges from approximately 5.0 × 10^-324 to 1.7 × 10^308.

Example:

#include <iostream>

using namespace std;

int main() {

    int n = 40;

    float factFloat = 1.0f;

    double factDouble = 1.0;

    for (int i = 1; i <= n; ++i) {

        factFloat \*= i;

        factDouble \*= i;

    }

    cout << "Factorial using float: " << factFloat << endl;

    cout << "Factorial using double: " << factDouble << endl;

    return 0;

}

Output :

Factorial using float: inf

Factorial using double: 8.15915e+047

So we can clearly see the change in the range.

Question : Explain different types of Logic gates with their truth table.

Answer :

Sure, here are the basic types of logic gates along with their truth tables:

**AND Gate**:

The output of an AND gate is high (1) only when all of its inputs are high (1), otherwise, it's low (0).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**OR Gate**:

The output of an OR gate is high (1) when at least one of its inputs is high (1), otherwise, it's low (0).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**NOT Gate** (Inverter):

The output of a NOT gate is the inverse of its input. If the input is high (1), the output is low (0), and vice versa.

Truth table:

|  |  |
| --- | --- |
| A | Y |
| 0 | 1 |
| 1 | 0 |

**NAND Gate** (NOT-AND):

The output of a NAND gate is the opposite of an AND gate. It's low (0) only when all of its inputs are high (1), otherwise, it's high (1).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**NOR Gate** (NOT-OR):

The output of a NOR gate is the opposite of an OR gate. It's high (1) only when all of its inputs are low (0), otherwise, it's low (0).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

**XOR Gate** (Exclusive OR):

The output of an XOR gate is high (1) when the number of high inputs is odd, otherwise, it's low (0).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**XNOR Gate** (Exclusive NOR):

The output of an XNOR gate is high (1) when the number of high inputs is even, otherwise, it's low (0).

Truth table:

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |