Floating Point Representation

Bin2Float

Write a C++ program to convert a 32-bit binary number to its corresponding single precision floating point number. The 32-bit binary number representation is stored as an **unsigned int** which is 4 bytes long.

Version 1 – Manual

The program will perform the necessary conversion steps as presented in slide 36 of chapter 3. It must manually perform the steps as shown in the slide to do the conversion.

Version 2 – Automatic

For this version, the program can use any standard C++ command and/or built-in library to do the conversion.

Float2Bin

Write a C++ program to convert a single precision floating point number to its corresponding 32-bit binary representation. The program starts with a given floating point number. It will then convert this float to its corresponding 32-bit binary representation.

Version 1 – Manual

The program will perform the necessary conversion steps as presented in slides 33 to 35 of chapter 3. It must manually perform the steps as shown in the slides to do the conversion.

Version 2 – Automatic

For this version, the program can use any standard C++ command and/or built-in library to do the conversion.

Single precision floating point representation format

| S | 8-bit Exponent | 23-bit Fraction |

| 1 | 10000001 | 01000000000000000000000 | = -5

| 1 | 10000111 | 01011001010000000000000 | = -345.25

| 0 | 01111100 | 11000000000000000000000 | = 0.21875

| 0 | 10000100 | 01101110010100000000000 | = 45.7890625

Some useful code

```
#include <bitset>  // needed to work with binary strings
using namespace std;

unsigned long ul = 0xbe600000;              // 0x for hex constant
ul = 0b10111110011000000000000000000000;   // 0b for bin constant
cout << sizeof(ul) << endl;
cout << "dec=" << ul << endl;               // print ul in decimal
cout << "hex=" << hex << ul << endl;        // print ul in hex

bitset<32> binaryBits(ul);       // convert 32-bit decimal ul to bits
                      // the bits are stored in the variable binaryBits
cout << "bits=" << binaryBits << endl;   // print ul in binary

binaryBits[2] = 1;                          // set bit 2 to a 1
unsigned long ul = binaryBits.to_ulong();  // converts bitset to
unsigned long
```