SINGLE PRECISION REPRESENTATION

EXP NO: 35

AIM: To write a C program to implement In Single Precision Representation.

ALGORITHM:

1. Extract the sign from the number. If the number is positive, set the sign bit to 0; if negative, set it to 1.
2. Convert the magnitude of the number to binary (ignore the sign for now).
3. Normalize the binary number by shifting the decimal point so that there's exactly one bit to the left of the decimal point (in the form 1.xxxxx).
4. Calculate the exponent by counting the number of shifts needed to normalize the number. Add 127 (bias) to this exponent.
5. Form the final representation:
   - Use the first bit for the sign.
   - Use the next 8 bits for the exponent.
   - Use the remaining 23 bits for the fractional part (mantissa) after the decimal point.
6. Combine the sign, exponent, and mantissa to get the single-precision floating-point number.

PROGRAM/OUTPUT SS:

Single precisiom representation.cpp

```c
#include <stdio.h>

void printBinary(int n, int i){
    int k;
    for (k = i - 1; k >= 0; k--) {
        if ((n >> k) & 1) {
            printf("1");
        } else {
            printf("0");
        }
    }
}

typedef union {
    float f;
    struct {
        unsigned int mantissa : 23;
        unsigned int exponent : 8;
        unsigned int sign : 1;
    } raw;
} myfloat;

void printIEEE(myfloat var) {
    printf("%d | ", var.raw.sign);
    printBinary(var.raw.exponent, 8);
    printf(" | ");
    printBinary(var.raw.mantissa, 23);
    printf("\n");
}

int main() {
    myfloat var;
    var.f = 1259.125;
    printf("IEEE 754 representation of %f is : \n", var.f);
    printIEEE(var);
    return 0;
}
```

```
C:\Users\praba\OneDrive\Des
IEEE 754 representation of 1259.125000 is :
0 | 10001001 | 00111010110010000000000

--------------------------------
Process exited after 0.1856 seconds with return value 0
Press any key to continue . . .
```

RESULT: Thus the C program has been executed successfully using DevC++.