Solve the following decimal expression in 32-bit single precision IEEE754 format:

0.1 + 0.2

0.1

Step 1: Convert the given decimal into binary form

Exponent: 0

Mantissa: Multiply 0.1 by 2

0.1*2 =	0	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	*Recurring!*
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	

Decimal:	0	.	1
Binary:	0	.	0001 1001 1001 1001 1001 1001...

Step 2: Represent the obtained binary in scientific notation

In scientific notation, a float binary is written in a form so that it begins with "1."
Hence, we must move the point 4 places to the *right* in the obtained binary.

We have,	0	.	0001 1001 1001 1001 1001 1001...

After bit-shifting,

1	.	1001 1001 1001 1001 1001 1001... * 2^ -4

Step 3: Convert the scientific notation into 32-bit single precision IEEE754 format

We need to represent the power of 2, -4 in our case, in bits. As per the IEEE rules, we must add a *bias* to the power, and then represent the resultant in bits. In 32-bit format,

Bias: 127

=> Exponent = 127 - 4 = 123

Binary of 123:

Quotient	Remainder
61	1
30	1
15	0
7	1
3	1
1	1

Now we can represent the exponent in 8 bits as per the 32-bit IEEE format.

0	1	1	1	1	0	1	1

So in single precision, 0.1 is represented as:

Sign	Exponent	Mantissa	
0	01111011	10011001100110011001100	
		10011001100110011001101	*Rounded!*

0.2

Step 1: Convert the given decimal into binary form

Exponent: 0

Mantissa: Multiply 0.2 by 2

0.2*2 =	0	
0.4*2 =	0	Recurring!
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	
0.2*2 =	0	
0.4*2 =	0	
0.8*2 =	1	
0.6*2 =	1	

0.2*2 =	0
0.4*2 =	0
0.8*2 =	1
0.6*2 =	1
0.2*2 =	0
0.4*2 =	0
0.8*2 =	1
0.6*2 =	1
0.2*2 =	0
0.4*2 =	0
0.8*2 =	1
0.6*2 =	1

Decimal:	0	.	2
Binary:	0	.	0011 0011 0011 0011 0011 0011...

Step 2: Represent the obtained binary in scientific notation

In scientific notation, a float binary is written in a form so that it begins with "1."
Hence, we must move the point 3 places to the *right* in the obtained binary.

We have, 0 . 0011 0011 0011 0011 0011 0011...

After bit-shifting,

1 . 1 0011 0011 0011 0011 0011 0011... * 2^ -3

Step 3: Convert the scientific notation into 32-bit single precision IEEE754 format

We need to represent the power of 2, -3 in our case, in bits. As per the IEEE rules, we must add a *bias* to the power, and then represent the resultant in bits. In 32-bit format,

Bias: 127

=> Exponent = 127 - 3 = 124

Binary of 124:

Quotient	Remainder
62	0
31	0
15	1
7	1
3	1
1	1

Now we can represent the exponent in 8 bits as per the 32-bit IEEE format.

0	1	1	1	1	1	0	0

So in single precision, 0.2 is represented as:

Sign	Exponent	Mantissa	
0	01111100	10011001100110011001100	
		10011001100110011001101	*Rounded!*

0.1 + 0.2

Step 1: To find the sum, we consider the respective IEEE binaries of each operand in scientific notation.

0.1	1	.	1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 * 2^ -4
0.2	1	.	1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 * 2^ -3

Step 2: As per IEEE, the above binaries should have the *same* exponent to carry out addition. Therefore,

0.1	0	.	1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 * 2^ -3	point moved 1 place to the *left*
0.2	1	.	1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 * 2^ -3	
	1 0	.	0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 * 2^ -3	Sum

Step 3: Convert the sum into scientific notation, again.

We move the point one place to the left.

1	.	0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 * 2^ -3	Sum

Step 4: Convert the scientific notation into 32-bit single precision IEEE754 format

We need to represent the power of 2, -2 in our case, in bits. As per the IEEE rules, we must add a *bias* to the power, and then represent the resultant in bits. In 32-bit format,

Bias: 127

=> Exponent = 127 - 2 = 125

Binary of 125:

Quotient	Remainder
62	1
31	0
15	1
7	1
3	1
1	1

Now we can represent the exponent in 8 bits as per the 32-bit IEEE format.

0	1	1	1	1	1	0	1

So in single precision, 0.1 + 0.2 is represented as:

Sign	Exponent	Mantissa	
0	01111101	0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1	
		0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 0	*Rounded!*

Sign	0			
	0			
	1			
	1			
Exponent	1			
	1			
	1			
	0			
	1			
	0	2^ -1	0	
	0	2^ -2	0	
	1	2^ -3	0.125	
	1	2^ -4	0.0625	
	0	2^ -5	0	
	0	2^ -6	0	
	1	2^ -7	0.0078125	
	1	2^ -8	0.00390625	
	0	2^ -9	0	
Mantissa	0	2^ -10	0	
	1	2^ -11	0.00048828125	
	1	2^ -12	0.000244140625	
	0	2^ -13	0	
	0	2^ -14	0	
	1	2^ -15	0.000030517578125	
	1	2^ -16	0.0000152587890625	
	0	2^ -17	0	
	0	2^ -18	0	
	1	2^ -19	0.0000019073486328125	
	1	2^ -20	0.00000095367431640625	
	0	2^ -21	0	
	1	2^ -22	0.0000002384185791015625	
	0	2^ -23	0	
			0.2000000476837158203125	Sum

Cross-verification:

The above sum is in IEEE format, so it must be converted to decimal. The standard formula for the same:

(-1)^Sign * (1 + Mantissa) * 2^Exponent

=> (-1)^0 * (1 + 0.2000000476837158203125) * 2^ -2

= 1.2000000476837158203125 / 4

= **0.300000011920928955078125**