1. (+10) Write a C ++ program that finds a practical measure of your machine's SP (32 bit) and DP (64 bit) floating point precision by taking the difference of 2 numbers and comparing these to zero in the same precision. What value do you obtain for $\epsilon_{\text{machine}}$ in both precisions? Hint: A loop (j) will be helpful here $1 - \left(1 + \frac{1}{2^j}\right)$ should do it.

```cpp
int question1() {
    // Float (32 bits)
    std::cout << "###### Single (32 bits) Precision: #######" << std::endl;
    float epsilon_float = 0.0f;
    for (int j = 0; j <= 100; j++) {
        float a = 1.0f;
        float b = 1.0f + 1.0f / pow(2.0f, float(j));
        float diff = a - b;
        if (diff == epsilon_float) {
            epsilon_float = 1.0f / pow(2.0f, float(j-1));
            std::cout << "Estimated single-precision (32 bit) machine epsilon: " << epsilon_float << std::endl;
            break;
        }
    }
    // Double (64 bits)
    std::cout << std::endl;
    std::cout << "###### Double (64 bits) Precision: #######" << std::endl;
    double epsilon_double = 0.0;
    for (int j = 0; j <= 100; j++) {
        double a = 1.0;
        double b = 1.0 + 1.0 / pow(2.0, double(j));
        double diff = a - b;
        if (diff == epsilon_double) {
            epsilon_float = (double)1.0 / pow(2.0, double(j-1));
            std::cout << "Estimated double-precision (64 bit) machine epsilon: " << epsilon_float << std::endl;
            break;
        }
    }
    return 0;
};
```

```
(base) chris@Chriss-MacBook-Pro-4 HW % cd "/Users/chris/Library/Cl
nor/AMATH 583/HW/"HW2
###### Single (32 bits) Precision: #######
Estimated single-precision (32 bit) machine epsilon: 1.19209e-07

###### Double (64 bits) Precision: #######
Estimated double-precision (64 bit) machine epsilon: 2.22045e-16
```

2. (+12) What are the largest and smallest SP (32 bit) and DP (64 bit) numbers that can be represented in IEEE floating point representation? Show work in terms of sign, mantissa, and exponent.

SP

S Sign bit : 1 bit

E Exponent : 8 bits

f fraction : 23 bits

$$(-1)^s \times (1.f) \times 2^{(E-127)}$$

$$1.f = 1 + \frac{\text{fraction bits}}{2^{23}}$$

$$1 \le E \le 254$$

largest SP: $s = 0$, $\Rightarrow (-1)^0 = 1$

$2^8 = 256$ values, 0 to 255

↳ 255 being represented as ∞, so $E \le 254$

0 being represented as $-\infty$, so $E \ge 1$

$$E = 254 \Rightarrow 2^{(254-127)} = 2^{127}$$

Largest fraction: $1.1111....$

$$0.111.... = 1 - 2^{23}$$

$$1.f = 1 + 1 - 2^{23} = 2 - 2^{23}$$

$$(-1)^0 \times (2 - 2^{23}) \times 2^{127} \approx \underline{3.4028 \times 10^{38}}$$

Smallest SP

$$s = 0 \Rightarrow (-1)^0 = -1$$

$$E = 1$$

smallest fraction $1.000.... = 1.0$

$$(-1)^0 \times 1.0 \times 2^{-126} \approx \underline{1.1755 \times 10^{-38}}$$

DP

Sign Bit : 1 bit

Exponent : 11 bits $\quad 1 \le E \le 2046 \quad (-1)^s \times (1.f) \times 2^{(E-1023)}$

Fraction : 52 bits

Largest DP: $\quad s = 0 \Rightarrow (-1)^0 = 1$

$E = 2046 \Rightarrow 2^{2046-1023} = 2^{1023}$

$f = 1.111 \dots = (2 - 2^{-52})$

$(-1)^0 \times (2 - 2^{-52}) \times 2^{1023} \approx 1.798 \times 10^{308}$

Smallest DP: $\quad s = 0 \Rightarrow (-1)^0 = 1$

$E = 1 \Rightarrow 2^{1-1023} = 2^{-1022}$

$f = 1.0$

$(-1)^0 \times 1.0 \times 2^{-1022} \approx 2.2251 \times 10^{-308}$ #

3. (+5) Write a C++ program to multiply the integers $200 * 300 * 400 * 500$ on your computer? What is the result? Name the effect you observe.

```
int question3() {
    // Using int type
    int result_int = (int) 200 * (int) 300 * (int) 400 * (int) 500;
    std::cout << "Question 3 (int type): " << result_int << std::endl;
    // Using long type
    long result_long = (long) 200 * (long) 300 * (long) 400 * (long) 500;
    std::cout << "Question 3 (long type): " << (long) result_long << std::endl;
    // Using long long type
    long long result_long_long = (long long) 200 * (long long) 300 * (long long) 400 * (long long) 500;
    std::cout << "Question 3 (long long type): " << (long long) result_long_long << std::endl;
    return 0;
}
```

```
(base) chris@Chriss-MacBook-Pro-4 HW % cd "/Users/chris/Library/CloudStorage/OneDrive-UW/4. Seinor/AMATH 583
nor/AMATH 583/HW/"HW2
HW2.cpp:39:56: warning: overflow in expression; result is -884901888 with type 'int' [-Winteger-overflow]
   39 |     int result_int = (int) 200 * (int) 300 * (int) 400 * (int) 500;
      |                                                        ^
1 warning generated.
Question 3 (int type): -884901888
Question 3 (long type): 12000000000
Question 3 (long long type): 12000000000
```

Integer multiplication effect : overflow

4. (+5) Given C++ code segment below, what is the final value of *counter*?

```
unsigned int counter = 0;
for (int i = 0; i < 3; ++i) --counter;
```

```cpp
int question4() {
    unsigned int counter = 0;
    for (int i = 0; i < 3; i++) --counter;
    std::cout << "Final Counter Value: " << counter << std::endl;
    return 0;
}
```

```
clang++: error: linker command failed with
(base) chris@Chriss-MacBook-Pro-4 HW % cd
nor/AMATH 583/HW/"HW2
Final Counter Value: 4294967293
(base) chris@Chriss-MacBook-Pro-4 HW %
```

5. (+10) Count and report how many IEEE SP (32 bit) normalized and denormalized floating point numbers there are. Please count and label infinities and NANs as well. Show work.

Normalized numbers

sign bit : 1 bit (0 or 1)
Exponent : 1 to 254 ⇒ 254 values
fraction : 23 bits ⇒ 2^{23} values

Total # of normalized numbers = $2 \times 254 \times 2^{23}$ = 4,261,412,864 #

Denormalized numbers

sign bit = 1 bit
Exponent : 0
Fraction : 23 bits ⇒ $2^{23} - 1$ (minus 0 value)

Total # of denormalized numbers = $2 \times (2^{23} - 1)$ = 16,777,214 (denormalized) #

Zeros : +0 or −0 ⇒ 2 (zeros) #   Infinity : +∞ or −∞ ⇒ 2 (Infinities) #

NaN :
sign bit : 1 bit
Exponent : 255
fraction : 23 bits ⇒ $2^{23} - 1$

$\Bigg\}$ $2 \times (2^{23} - 1)$ = 16,777,214 (NaN) #

6. (+15) Consider a 6 bit floating point system with $s = 1$ (1 sign bit), $k = 3$ (3 bit exponent field), and $n = 2$ (2 bit mantissa).

(a) Calculate by hand all the representable normalized numbers. Show work.

(b) Calculate by hand all the representable denormalized numbers. Show work.

(c) Plot both sets of numbers (ignoring NANs and infinities) as a number line to see the gaps of (un)representable numbers.

$$\text{Value} = (-1)^s \times \left(1 + \frac{\text{fraction bits}}{2^n}\right) \times 2^{(e - \text{bias})}$$

$s = 0/1$, $n = 2$, $\text{bias} = 2^{k-1} - 1 = 2^2 - 1 = 3$

$E = e - \text{bias} = \{1, 2, 3, 4, 5, 6\} - 3 = \{-2, -1, 0, 1, 2, 3\}$

$M = 1 + \frac{fb}{4}$

$\text{Value} = (-1)^s \times M \times 2^E$ $\qquad fb = \{\overset{0}{00}, \overset{1}{01}, \overset{2}{10}, \overset{3}{11}\}$

$M = \{1.0, 1.25, 1.50, 1.75\}$

a)

$S = 1$ (Positive)

$E = -2$, Value $= (1.0) \times 2^{-2} = 0.25$
$(1.25) \times 2^{-2} = 0.3125$
$(1.50) \times 2^{-2} = 0.375$
$(1.75) \times 2^{-2} = 0.4375$

$E = -1$, Value $= (1.0) \times 2^{-1} = 0.5$
$(1.25) \times 2^{-1} = 0.625$
$(1.50) \times 2^{-1} = 0.75$
$(1.75) \times 2^{-1} = 0.875$

$E = 0$, Value $= (1.0) \times 2^{0} = 1.0$
$(1.25) \times 2^{0} = 1.25$
$(1.50) \times 2^{0} = 1.50$
$(1.75) \times 2^{0} = 1.75$

$E = 1$, Value $= (1.0) \times 2^{1} = 2.0$
$(1.25) \times 2^{1} = 2.5$
$(1.50) \times 2^{1} = 3.0$
$(1.75) \times 2^{1} = 3.5$

$E = 2$, Value $= (1.0) \times 2^{2} = 4.0$
$(1.25) \times 2^{2} = 5.0$
$(1.50) \times 2^{2} = 6.0$
$(1.75) \times 2^{2} = 7.0$

$E = 3$, Value $= (1.0) \times 2^{3} = 8.0$
$(1.25) \times 2^{3} = 10.0$
$(1.50) \times 2^{3} = 12.0$
$(1.75) \times 2^{3} = 14.0$

Same value with negative sign for $S = 0$

b) Denormalized value

$$M = \frac{fb}{4} = \{0.0,\ 0.25,\ 0.50,\ 0.75\}$$

$$E = 1 - bias = 1 - 3 = -2$$

$$Value = (-1)^S \times M \times 2^E$$

S = 1 (Positive)

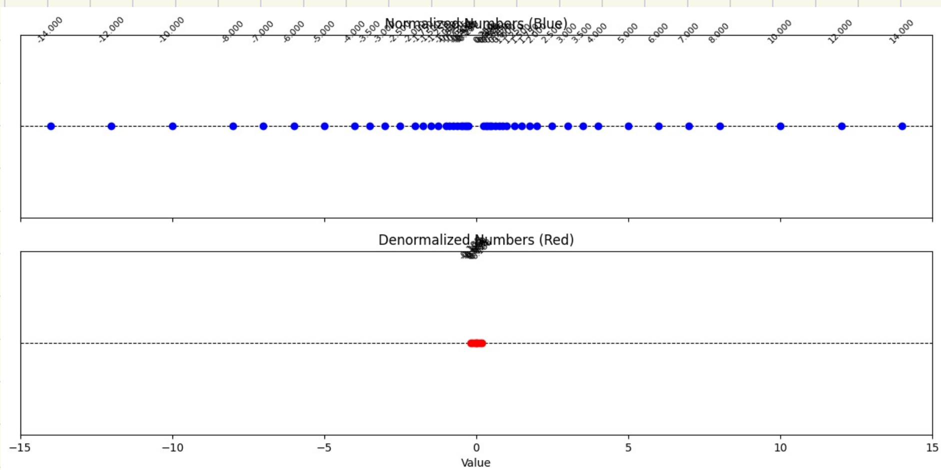
$$Value = 0.0 \times 2^{-2} = 0$$
$$0.25 \times 2^{-2} = 0.0625$$
$$0.50 \times 2^{-2} = 0.125$$
$$0.75 \times 2^{-2} = 0.1875$$

S = -1 (Negative)

$$Value = 0$$
$$= -0.0625$$
$$= -0.125$$
$$= -0.1875$$

zero $\Rightarrow \pm 0$,

c)



7. (+10) Conversions. Show work.

5 4 3 2 1 0

(a) Write $(D3B701)_{16}$ as an integer in base-10.

(b) Write $(1010\underbrace{0001}\underbrace{0011}\underbrace{1111})_2$ as an integer in base-16, i.e. as a hexadecimal number.

a)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
										10	11	12	13	14	15

$$13 \cdot 16^5 + 3 \cdot 16^4 + 11 \cdot 16^3 + 7 \cdot 16^2 + 0 \cdot 16^1 + 1 \cdot 16^0$$

$$= \underline{13874945}\ \#$$

b)

1010	0001	0011	1111
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$2^3+2=10$	1	$2+1=3$	$2^3+2^2+2^1+2^0=15$
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
A	1	3	F

12

$8+4+2+1=$

$\underline{(A13F)_{16}}\ \#$

8. (+5) Are there $a, b, c \in \mathbb{Z}$ s.t. $6a + 9b + 15c = 107$? Show work.

$\gcd(6, 9, 15) = 3$

Any linear combination of $6, 9, 15$, $6a+9b+15c$ $a, b, c \in \mathbb{Z}$ must be a scalar multiple of 3. Since 107 is not divisible by 3, there does not exist integers a, b, c s.t. $6a+9b+15c=107$.

```
int question8() {
    int a_max = 107 / 6 + 1;
    int b_max = 107 / 9 + 1;
    int c_max = 107 / 15 + 1;
    int target = 107;
    for (int i = 0; i <= a_max; i++) {
        for (int j = 0; j <= b_max; j++) {
            for (int k = 0; k <= c_max; k++) {
                if ((6 * i + 9 * j + 15 * k) == target) {
                    std::cout << i << ", " << j << ", " << k << std::endl;
                    break;
                }
            }
        }
    }
    std::cout << "DNE" << std::endl;
    return 0;
}
```

```
(base) chris@Chriss-MacBook-Pro
nor/AMATH 583/HW/"HW2
Question 8: DNE
```

9. (+10) Equivalence classes modulo n. $\forall a, b \in \mathbf{Z}$ then $a \equiv b \pmod{n}$ means $n \mid (a-b)$ or $a = b + k \cdot n$ and $k \in \mathbf{Z}$. $\mathbf{Z}_n$ is the set of equivalence classes $\{[0], [1], \ldots, [n-1]\}$. Is $(\mathbf{Z}_n, +, \cdot)$ a ring? Hint: If $s \in [i]$, then $n \mid (s-i)$. Show work (use ring properties).

If $[a] = [a']$ and $[b] = [b']$

Show $[a+b] = [a'+b']$ and $[a \times b] = [a' \times b']$

Since $[a] = [a']$ means $a \equiv a' \bmod n$, $n \mid (a-a')$
$[b] = [b']$ $b \equiv b' \bmod n.$ $n \mid (b-b')$

$(a+b) - (a' - b') = (a-a') + (b-b')$ is also divisible by n.

Operations well-defined.

Ring Axiom Verification

R1: Closure under $+$ and $\times$

If $[a], [b] \in \mathbb{Z}\ \mathbb{Z}_n$, then $\left.\begin{array}{l}[a]+[b] = [a+b] \\ [a] \cdot [b] = [ab]\end{array}\right\}$ are still equivalence class

R2: Associativity

$([a]+[b]) + [c] = [(a+b) + c] = [a + (b+c)] - [a] + ([b] + [c])$.

R3: Commutativity $+$

$[a] + [b] = [b] + [a]$ because $a+b = b+a$ in $\mathbb{Z}$

R4: Existence of additive identity

$[a] + [0] = [a+0] = [a]$

R5: Inverse additive

$[a] + [-a] = [a + (-a)] = [0]$

R6: Commutativity of $\times$

$[a] \times [b] = [ab] = [ba] = [b] \times [a]$

R7: Existence of Multiplicative Identity

$[a] \times [1] = [a \times 1] = [a]$

R8: Distributivity

$([a] + [b]) \times [c] = [(a+b) \times c] = [ac + bc] = [ac] + [bc] = ([a] \times [c]) + ([b] \times [c])$