Assignment 3-5 Floating Point Representation: Infinity Lab 4

1. **Make an assembly program that shows the infinite numbers**
    a. **Largest Positive Numbers x 1.0e3**
        i. **The largest normalized positive number**

```
## Program that multiply the Positive Largest number by 1.0E3
        .data
val1:    .word  0x7F7FFFFF      # Positive Largest Number
val2:    .float 1000.0          # 1.0E3
         .text
        .globl  main
```

2. **And then make an assembly code to multiply 1.0e3**

```
## Program that multiply the Positive Largest number by 1.0E3
        .data
val1:    .word  0x7F7FFFFF      # Positive Largest Number
val2:    .float 1000.0          # 1.0E3
         .text
        .globl  main

main:
    li $v0, 2       # print floating service code
    l.s       $f5, val1
    l.s    $f7, val2
    mul.s          $f9, $f5, $f7
    li          $v0, 2
    mfc1   $t0, $f9
    mov.s   $f12, $f9
    syscall
## End of file
```

3. **Print the result of the multiplication**
    mov.s $f12, $f9 = Move result to $f12 for printing
    syscall = Print the floating-point result

```
FG8      = 0.00000
FG9      = Infinity
FG10     = 0.00000
FG11     = 0.00000
```

Output

```
inf
```

4. **Elaborate on the value with IEEE 754 format**
    a. **.word 0x7F7FFFFF**

0x7FFFFFFF = 0111 1111 0111 1111 1111 1111 1111 1111
0 = sign bit = **positive**
1111 1110 = Exponent = 254 - 127 = **127** (largest normalized positive exponent value)

111 1111 1111 1111 1111 1111 = Mantissa = Represents a normalized fraction **approximately equal to 1.99999** and so on.

b. The register value $t0 after multiplication

0x7F800000 = 0111 1111 1000 0000 0000 0000 0000 0000
0 = sign bit = **positive**
1111 1111 = Exponent = When the exponent bits are all 1's, it indicates that the value is **either Infinity or NaN.**
000 0000 0000 0000 0000 0000 = When the mantissa bits are all 0's when the exponent bits are all 1's, this tells us that the value is **Infinity**.
So the value is **Positive Infinity**