1

Find tightest O(f(n)) for each of the following functions: The tightest big-O bound is the narrowest upper bound within the big-O category.

a

$f(n) = 3n$

✓ **Answer**

$3n = O(n)$

b

$f(n) = \frac{\log(n)}{n^2}$

✓ **Answer**

$\frac{\log(n)}{n^2} = O(1)$

c

$f(n) = n \log n$

✓ **Answer**

$n \log n = O(n \log n)$

d

$f(n) = n + \frac{n}{2} + \frac{n}{4} + \ldots + \frac{n}{2^n}$

✓ **Answer**

$n + \frac{n}{2} + \frac{n}{4} + \ldots + \frac{n}{2^n} = O(n)$

e

$f(n) = (\log n)^n + n^4$

✓ **Answer**

$(\log n)^n + n^4 = O((\log n)^n)$

f

$f(n) = \frac{n! + n^n}{3n}$

✓ **Answer**

$\frac{n! + n^n}{3n} = O(n^{n-1})$

g-i

Show you work by using the definition of big-O and finding values for c and N.
Reminder f(n) is O(g(n)) — if a positive real number c and positive integer N exist such that f(n) ≤ c × g(n) for all n ≥ N

g

$2^{n-1} = O(n)$

✓ **Answer**

False, the asymptotic behavior of n is strictly less than 2^{n-1}

$\lim_{n\to\infty} \frac{n}{2^{n-1}} = \lim_{n\to\infty} \frac{1}{2^{n-1}\ln 2} = 0$
Whilst in order for it to be big Oh, it must be greater than 0.

h

$n(\log n)^3 = O(n^{4/3})$

✓ **Answer**

False, the asymptotic behavior of $n^{4/3}$ is strictly less than $n(\log n)^3$

$\lim_{n\to\infty} \frac{n^{4/3}}{n(\log n)^3}$
$= \lim_{n\to\infty} \frac{\frac{4}{3}n^{1/3}}{\frac{3}{\ln 10}(\log n)^2 + (\log n)^3}$
$= \lim_{n\to\infty} \frac{\frac{4}{9}n^{-2/3}}{\frac{6}{x\ln 10}(\log x) + \frac{3}{x}(\log x)^3}$

$= \lim_{n\to\infty} \frac{\frac{4}{9}n^{-5/3}}{\frac{6}{\ln 10}(\log x)+3(\log x)^3}$
$= 0$
Whilst in order for it to be big Oh, it must be greater than 0.

i

$\frac{n^4+1}{n^2} = O(n)$

✓ **Answer**

False, the asymptotic behavior of n is strictly less than $\frac{n^4+1}{n^2}$

$\lim_{n\to\infty} \frac{n}{\frac{n^4+1}{n^2}}$
$= \lim_{n\to\infty} \frac{n^3}{n^4+1}$
$= \lim_{n\to\infty} \frac{6}{24n}$
$= 0$
Whilst in order for it to be big Oh, it must be greater than 0.

2

Given the following code, analyze and give the tightest big-Θ bound. Show how you came to your answer by indicating what the big-Θ is for each line.

a

```
public static int sum1() {
        int sum = 0;
        for(int i = 0; i < n; i++) {
                if(sum < n) {
                        for(int j = 0; j < n; j++) {
                                sum++;
                        }
                }
        }
        return sum;
}
```

✓ **Answer**

```
public static int sum1() {
        int sum = 0;                                    //1
```

```
        for(int i = 0; i < n; i++) {                //1*
                if(sum < n) {                           //1
                        for(int j = 0; j < n; j++) {    //n*
                                sum++;                          //1
                        }
                }
        }
        return sum;                                 //1
}
```

$O(n)$

b

```
public static int sum2() {
        int sum = 0;
        for(int i = n; i > 1; i = i/3) {
                sum = sum + 2;
        }
        return sum;
}
```

✓ **Answer**

```
public static int sum2() {
        int sum = 0;                                //1
        for(int i = n; i > 1; i = i/3) {            //log_3 n*
                sum = sum + 2;                          //1
        }
        return sum;                                 //1
}
```

$O(\log_3 n)$

c

```
public static int sum3() {
        int sum = 0;
        for(int i = 0; i < n; i++) {
                for(int j = 0; j < n; j++) {
                        if(i < j) {
```

```
                                for(int k = i; k < j; k++) {
                                        sum++;
                                }
                        }
                }
        }
        return sum;
}
```

✓ **Answer**

```
public static int sum3() {
        int sum = 0;                                    //1
        for(int i = 0; i < n; i++) {                    //n*
                for(int j = 0; j < n; j++) {                //n*
                        if(i < j) {                             //1/2*
                                for(int k = i; k < j; k++) {  //n/2*
                                        sum++;
//1
                                }
                        }
                }
        }
        return sum;                                     //1
}
```

$O(n^3)$