CS 101 Lab 2 **Summer 2016** **Name** ______________________

Write the running time of each code fragment as the best Big O function of n.

1.	`for (i=1; i<=n; i++)` `    k++;`	O(n)
2.	`for (i=1; i<=1000; i++)` `    k++;`	O(1)

3.	`for (i=1; i<=n; i++)`
	`    for (j=1; j<=i; j++)`
	`        k++;`
		O(n^2)

4.	`for (i=1; i<=n; i++)`
	`    for (j=i; j<=n; j++)`
	`        k++;`
		O(n^2)

5.	`for (i=1; i<=n; i++)`
	`    for (j=i; j<=n; j++)`
	`        for (k=1; k<=j; k++)`
	`            m++;`
		O(n^3)

6.	`for (i=1; i<=n; i++)`
	`    for (j=1; j<=200; j++)`
	`        for (k=1; k<=5000; k++)`
	`            m++;`
		O(n)

7.	`k=1;`
	`for (i=1; i<=n; i++)` // n iterations
	`    k*=2;`
	`for (j=1; j<=k; j++)` // 2^n iterations
	`    m++;`
		O(2^n)

8.	`k=1;`
	`for (i=1; i<=n; i++)` // n iterations
	`    k*=2;`
	`for (j=1; j<=k; j*=2)` // lg 2^n = n iterations
	`    m++;`
		O(n)

9.	`for (j=1; j*j<=n; j++)` `    k++;`	O($\sqrt{n}$)
10.	`for (k=1; k<=n; k*=2)` `    j++;`	O(lg n)

11.	`for (k=1; k<=n; k*=2)`
	`    for (j=1; j<=n; j++)`
	`        m++;`
		$O(n \lg n)$

12.	`for (k=1; k<=n; k++)`
	`    for (j=1; j<=k; j*=2)`
	`        m++;`
		$O(n \lg n)$

13.	`for (k=1; k<=n; k*=2)`
	`    for (j=1; j<=k; j++)`
	`        m++;`
	// Geometric series: 1+2+4+8+16+32+…+n = 2n–1 iterations
		$O(n)$

14.	`for (i=1; i<=n; i*=2)`
	`    for (j=1; j<=n; j*=2)`
	`        k++;`
		$O(\lg^2 n)$

15.	`k=0;`
	`for (i=1; i<=n; i*=2)` // lg n iterations
	`    k++;`
	`for (j=1; j<=k; j++)` // lg n iterations
	`    m++;`
		$O(\lg n)$

16.	`k=1;`
	`for (i=1; i<=n; i++)` // n iterations
	`    k*=i;`
	`for (j=1; j<=k; j++)` // n! iterations
	`     m++;`
		$O(n!)$

17.	`for (i=1; i<=n; i++)`
	`    for (j=1; j<=n; j++)`
	`        if (i==j)` // condition is true n times
	`            for (k=1; k<=n; k++)`
	`                m++;`
	// $n*O(n) + (n^2 – n)*O(1)$
		$O(n^2)$

18.	`for (i=1; i<=n; i++)`
	`    for (j=1; j<=n; j++)`
	`        if (i!=j)` // condition is true $n^2 – n$ times
	`            for (k=1; k<=n; k++)`
	`                m++;`
	// $(n^2 – n)*O(n) + n*O(1)$
		$O(n^3)$

19.	`k=1;`
	`for (j=1; j<=n; j+=k)`
	`    k+=2;`
	// j = 1, 4, 9, 16, 25, 36, …, n
		$O(\sqrt{n})$

20.	`m=n*n*n;`
	`while (m>=0)` // n^3 / n iterations
	`    m-=n;`
		$O(n^2)$