CS330 — Big-Oh Problems

1. Give the computational complexity of the following piece of code in Big-Oh notation and explain how you arrived at your result:

```
for (i=1; i<=n; i=i+2){
   // constant number of operations
 }
```

2. Give the computational complexity of the following piece of code in Big-Oh notation and explain how you arrived at your result:

```
for (i=1; i<=n; i++){
  for (j=i; j<=n; j=j+2){
   // constant number of operations
  }
 }
```

3. Give the computational complexity of the following piece of code in Big-Oh notation and explain how you arrived at your result:

```
for (j=n; j>1; j=j/2){
   // constant number of operations
}
```

4. Give the computational complexity of the following piece of code in Big-Oh notation and explain how you arrived at your result:

```
for (i=1; i<=n; i++){
  for (j=n; j>1; j=j/2){
   // constant number of operations
  }
 }
```

5. Is $n^2 + 100$ of complexity $O(n^2)$? Use the formal definition of Big-Oh to answer this question.
6. Is $n^2 + 100$ of complexity $O(n^3)$? Use the formal definition of Big-Oh to answer this question.
7. Is 2^{n+1} of complexity $O(2^n)$? Use the formal definition of Big-Oh to answer this question.
8. Is 2^{2n} of complexity $O(2^n)$? Use the formal definition of Big-Oh to answer this question.
9. Explain why the statement "The running time of the algorithm is at least $O(n^2)$" does not provide any information.