Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pledge: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Give the complexity of the following functions. Choose the most appropriate notation from among , , and .

1. **void** **function1**(**int** n) {

**for** (**int** i = 1; i <= n; i++) {

**for** (**int** j = i; j <= n; j += 2) {

cout << "\*";

}

}

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function2**(**int** n) {

**int** count = 0;

**for** (**int** i = 1; i \* i \* i <= n; i++) {

count++;

}

cout << count;

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function3**(**int** n) {

**int** count = 0;

**for** (**int** i = 1; i \* i <= n; i++) {

**for** (**int** j = 1; j + n/2 <= n; j++) {

**for** (**int** k = 1; k <= n; k \*= 2) {

count++;

}

}

}

cout << count;

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function4**(**int** n) {

**int** count = 0;

**for** (**int** i = n/2; i <= n; i++) {

**for** (**int** j = 1; j <= n; j \*= 2) {

**for** (**int** k = 1; k <= n; k \*= 2) {

count++;

}

}

}

cout << count;

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function5**(**int** n) {

**if** (n % 2 == 0) {

**return**;

}

**for** (**int** i = 1; i <= n; i++) {

**for** (**int** j = 1; j <= n; j++) {

cout << "\*";

**break**;

}

}

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function6**(**int** n) {

**int** count = 0;

**for** (**int** i = 1; i <= n/2; i++) {

**for** (**int** j = 1; j <= n/3; j++) {

**for** (**int** k = 1; k <= n/4; k++) {

count++;

}

}

}

cout << count;

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function7**(**int** n) {

**for** (**int** i = 1; i <= n; i++) {

**for** (**int** j = 1; j <= n; j += i) {

cout << "\*";

}

}

}

Answer: \_\_\_\_\_\_\_\_

1. **void** **function8**(**int** n) {

**int** i = 1, s = 1;

**while** (s <= n) {

i++;

s += i;

cout << "\*";

}

}

Answer: \_\_\_\_\_\_\_\_

1. Processing Arrays
   1. Suppose you have an unsorted array of integers of length and want to sum all the elements inside it. What is the running time of your algorithm? \_\_\_\_\_\_\_\_
   2. Suppose you have an unsorted array of integers of length and want to determine if all the values inside are positive. What is the running time of your algorithm? \_\_\_\_\_\_\_\_
   3. Suppose you have a sorted array of integers of length and want to determine the median value. What is the running time of your algorithm? \_\_\_\_\_\_\_\_
2. \_\_\_\_ T / F

If true, prove it by giving *integral* values for the required constants , , and . Choose the tightest values possible for the and constants. If false, show the contradiction.