1. Asymptotics Introduction

Give the runtime of the following functions in $\Theta$ notation. Your answer should be as simple as possible with no unnecessary leading constants or lower order terms.

```java
private void f1(int N) {
    for (int i = 1; i < N; i++) {
        for (int j = 1; j < i; j++) {
            System.out.println(quote("hello tony"));
        }
    }
}
$$\Theta(\text{___})$$

private void f2(int N) {
    for (int i = 1; i < N; i *= 2) {
        for (int j = 1; j < i; j++) {
            System.out.println(quote("hello hannah"));
        }
    }
}
$$\Theta(\text{___})$$
```
2 Finish the Runtimes

Below we see the standard nested for loop, but with missing pieces!

```
for (int i = 1; i < ______; i = ______) {
    for (int j = 1; j < ______; j = ______) {
        System.out.println("We will miss you next semester Akshit :(");
    }
}
```

For each part below, some of the blanks will be filled in, and a desired runtime will be given. Fill in the remaining blanks to achieve the desired runtime! There may be more than one correct answer.

**Hint:** You may find Math.pow helpful.

(a) Desired runtime: Θ($N^2$)

```
for (int i = 1; i < N; i = i + 1) {
    for (int j = 1; j < ______; j = ______) {
        System.out.println("This is one is low key hard");
    }
}
```

(b) Desired runtime: Θ($\log(N)$)

```
for (int i = 1; i < N; i = i * 2) {
    for (int j = 1; j < ______; j = j * 2) {
        System.out.println("This is one is mid key hard");
    }
}
```

(c) Desired runtime: Θ($2^N$)

```
for (int i = 1; i < N; i = i + 1) {
    for (int j = 1; j < ______; j = j + 1) {
        System.out.println("This is one is high key hard");
    }
}
```

(d) Desired runtime: Θ($N^3$)

```
for (int i = 1; i < ______; i = i * 2) {
    for (int j = 1; j < N * N; j = ______) {
        System.out.println("yikes");
    }
}
```
3 Bit Operations

In the following questions, use bit manipulation operations to achieve the intended functionality and fill out the function details -

(a) Implement a function `isPalindrome` which checks if the binary representation of a given number is palindrome. The function returns true if and only if the binary representation of `num` is a palindrome.

For example, the function should return true for `isPalindrome(9)` since binary representation of 9 is `1001` which is a palindrome.

```java
/** 
 * Returns true if binary representation of num is a palindrome 
 */

public static boolean isPalindrome(int num) {
    // stores reverse of binary representation of num
    int reverse = 0;

    while (num > 0) {
        reverse = (reverse << 1) | (num & 1);
        num >>= 1;
    }

    return num == reverse;
}
```
(b) Implement a function \texttt{swap} which for a given integer, swaps two bits at given positions. The function returns the resulting integer after bit swap operation. For example, when the function is called with inputs \texttt{swap(31, 3, 7)}, it should reverse the 3rd and 7th bits from the right and return 91 since 31 (00011111) would become 91 (01011011).

```java
/**
 * Function to swap bits at position a and b (from right) in integer num
 */
public static int swap(int num, int a, int b) {
    return num;
}
```

4 Bits Runtime

Determine the best and worst case runtime of \texttt{tricky}.

```java
public void tricky(int n) {
    if (n > 0) {
        tricky(n & (n - 1));
    }
}
```

Best Case: $\Theta(\phantom{\emptyset})$, Worst Case: $\Theta(\phantom{\emptyset})$