1. Prove using induction that \( n \) is \( O(2^n) \).

2. Prove that \( 2^{(n^2)} \) is not \( O(5^n) \). Do not use any theorems about Big-Oh that you might happen to know other than the definitions.

3. Solve the following recurrence. Give a tight bound, i.e., express your answer using the \( \Theta \) notation. Assume that \( T(n) = 1 \), when \( n = 1 \).

\[
T(n) = T(n-1) + 1/n
\]

4. Consider the following code fragment

```java
for(int i=1;i<=n;i=2*i)
    for (int k = i; k >0; k = k/2)
        print('*');
```

a. Compute the number of stars printed as a function of \( n \). You can assume that \( n = 2^m \).

b. Give a \( \Theta \)-bound.

5. Let \( X \) and \( Y \) be two \( n \)-digit numbers. Assume that the word size on your computer can store one base-10 digit. We want to compute the product \( XY \). Assume that \( n \) is a power of 2.

a. What is the running time of the multiplication algorithm that you learned in school?

b. Let \( X_h, Y_h \) be the higher order digits of \( X \) and \( Y \), respectively. Let \( X_l, Y_l \) be the lower order digits of \( X \) and \( Y \), respectively. Then the product \( XY \) can be written as

\[
XY = (X_h(10^{n/2}) + X_l)(Y_h(10^{n/2}) + Y_l) = X_hY_h(10^n) + (X_hY_l + X_lY_h)10^{n/2} + X_lY_l
\]

Use this to devise a divide and conquer algorithm for the problem.

c. Write a recurrence for your algorithm in part (b). What is the running time of your algorithm?

d. Consider the product \( (X_h + X_l)(Y_h + Y_l) \). Expand the product and after noticing the terms of this expansion, devise a divide and conquer algorithm that uses only 3 multiplications.

e. Write the recurrence for the algorithm in part (d) and derive its running time.
6. Modify the quicksort algorithm so that its running time is $O(n \log n)$ in the worst case. You may assume that all elements are distinct.

7. The Hotel Partner Problem is as follows. There are $2n$ people and $n$ hotel rooms. Each person maintains a preference list of the remaining $2n - 1$ people. The objective is to assign two people to each room such that the assignment is stable. An assignment is stable if there are no two people assigned to different rooms who prefer each other over their current partners. Give an instance of the Hotel Partner Problem for which there is no stable solution.

8. Suppose that you have a “black box” worst-case linear-time median subroutine. Give a simple, linear-time algorithm that solves the selection problem for any arbitrary order statistic, i.e., given an unsorted array $A$ containing $n$ elements and an integer $i$, in $O(n)$ time, your algorithm should return the element in $A$, which is the $i^{th}$ smallest element in $A$. Your algorithm must use the black-box median-finding subroutine. You may assume that $i$ lies within the bounds of the input array $A$ and that $n$ is a power of 2. Justify your answer.
Some Useful Facts

1. $\lg n = \log_2 n$
   $\ln n = \log_e n$

2. Below are some formulas that may come handy.
   \begin{itemize}
   \item $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
   \item $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$
   \item $\sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4}$
   \item $\sum_{i=0}^{n} c^i = \frac{c^{n+1}-1}{c-1}, c \neq 1$
   \item $\sum_{i=0}^{\infty} c^i = \frac{1}{1-c}, \left| c \right| < 1$
   \item $\sum_{i=1}^{\infty} c^i = \frac{c}{1-c}, \left| c \right| < 1$
   \item $\sum_{i=0}^{\infty} i c^i = \frac{c}{(1-c)^2}, \left| c \right| < 1$
   \item $H_n = \sum_{i=1}^{n} \frac{1}{i} = \ln n + O(1)$
   \end{itemize}

3. **Simplified Master Theorem.** Let $a \geq 1$, $b > 1$ be constants and let $T(n)$ be the recurrence
   
   $T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k)$
   
   defined for $n \geq 0$ (we assume that $n$ is a power of $b$, though this does not make a difference in asymptotic analysis). The base case, $T(1)$ can be any constant value. Then
   \begin{itemize}
   \item **Case 1:** if $a > b^k$, then $T(n) \in \Theta(n^{\log_b a})$.
   \item **Case 2:** if $a = b^k$, then $T(n) \in \Theta(n^k \log_b n)$.
   \item **Case 3:** if $a < b^k$, then $T(n) \in \Theta(n^k)$.
   \end{itemize}