1. Prove the following polynomial is \( \Theta(n^2) \).

\[ P(n) = 5n^2 - 100n + 1000. \]

(a) Prove \( O(n^2) \):

\[ \begin{array}{c|c}
\text{GRADE} & /20 \\
1 & /20 \\
2 & /20 \\
3 & /20 \\
4 & /20 \\
5 & /20 \\
\hline
\text{SUM} & /100 \\
\end{array} \]

(b) Prove \( \Omega(n^2) \):
2. Find the exact number of times (in terms of $n$) the innermost statement ($X = X + 1$) is executed in the following code. That is, find the final value of $X$. Then express the total running time in terms of $O(\ )$.

```plaintext
X = 0;
for i = 1 to n - 1
  for j = i to 2n + 1
    X = X + 1;
```
3. Consider the following divide-and-conquer algorithm (recursive function). Input array \( A \) is Boolean, with each element either 0 or 1. Parameter \( i \) is the starting index of the array, and \( n \) is the number of elements. The initial call is \( \text{COMPUTE}(A,0,n) \).

```c
int \text{COMPUTE} (\text{Boolean } A[\ ], \text{int } i, \text{int } n) \{
    \text{if } (n == 1) \text{ return } A[i];
    n1 = \lfloor n/2 \rfloor; \quad \text{//Length of first half of array}
    n2 = \lceil n/2 \rceil; \quad \text{//Length of second half of array}
    C1 = \text{COMPUTE } (A, i, n1);
    C2 = \text{COMPUTE } (A, i + n1, n2);
    \text{return } (C1 + C2)
\}
```

(a) Figure out what the function does. (What does it compute?) Explain briefly.

(b) Let \( f(n) \) be the number of times the arithmetic operation \( (C1 + C2) \) is performed by this algorithm. Assume that \( n \) is a power of 2. Write a recurrence for \( f(n) \). Find the solution of the recurrence by repeated substitution.
4. (a) Use repeated substitution to find the solution of the following recurrence.

\[ T(n) = \begin{cases} 
T(n/2) + n, & n \geq 2 \\
1, & n = 1.
\end{cases} \]

(b) Prove by induction that the solution of the following recurrence is \( T(n) = An^2 + Bn \), and determine the constants \( A \) and \( B \).

\[ T(n) = \begin{cases} 
4T(n/2) + n, & n \geq 2 \\
1, & n = 1.
\end{cases} \]
5. We have four sorted lists, each with $n/4$ elements. (Elements are real-valued.) We want to merge these lists into a single sorted list of $n$ elements.

(a) First consider the following naive approach.
   - Merge the first and second list into a sorted list of $2n/4$ elements,
   - Merge the result with the third list to get a sorted list of $3n/4$ elements,
   - Merge the result with the fourth list.

   Analyze the worst-case number of key comparisons. (Find the exact worst-case number, not order of it.)

(b) Describe a more efficient algorithm for this problem based on a divide-and-conquer technique. Use a diagram to help explain your algorithm. Analyze the worst-case number of key comparisons. (Again, find the exact worst-case number, not order of it.)