Due date: October 24 @ 11:55pm

Submission Instructions:

- Each person must submit a solution to this assignment on Moodle on the due date.
- Assignments must be typed. I suggest using Latex. If the grader (me!) cannot (easily) read an answer, she will give it a 0.
- Each group will work together, and create one solution, but each member must submit a solution to Moodle.
- Your submission must have the name of every group member clearly marked.
- All arguments should be clear, concise, and (of course) correct. You will lose points for poor writing, bad grammar, etc…
- Use of the Internet for solving these problems is strictly forbidden, and will be treated as an honor code violation.
- Inter-group work: Members of different groups may discuss the questions. However:
  - With your submission you must provide a list of all non-group members with whom any member of your group discussed the assignment.
  - No written material may leave the inter-group discussion. If you talk with someone about the assignment, you must throw away and written notes at the end of the discussion.
Problem 1 (20 points):
Use the Master Theorem to bound each of these recurrences. If the Master Theorem is not applicable, then explain why – but do not then bound the recurrence.

1. \( T(n) = 2T(5n) + 1 \ (T(1) = 1) \)
2. \( T(n) = 4T(n/2) + n \log_2 n \ (T(1) = 5^7) \)
3. \( T(n) = 16T(n/4) + n^3 \ (T(1) = 1) \)
4. \( T(n) = T(n/9) + 2T(n/9) + \sqrt{n} \ (T(1) = 17) \)
5. \( T(n) = 25T(n/5) + n^2 \log_5 n \ (T(1) = 5) \)
6. \( T(n) = 8T(n/2) + n - \log(n) \ (T(1) = 42) \)
Problem 2 (20 points): Consider the following sorting algorithm (from problem set 1):

```c
void StoogeSort(Array A, int i, int j)
(1)   if A[i] > A[j]
(2)     swap(A[i], A[j])
(3)   if i < j-1
        // t will be 1/3 the size of the array segment (rounded down)
(4)      t <-- floor( (j-i+1)/3 )
(5)    StoogeSort(A, i, j-t)
(6)    StoogeSort(A, i+t, j)
(7)    StoogeSort(A, i, j-t)
```

Give a runtime recursion on this (e.g. $T(n) = ???$), and then use the Master Theorem to give a $\Theta$ bound on $T(n)$. 
Problem 3 (20 points): Consider the following pseudocode (a variation of the code from Problem Set 1):

double pow(x, n):
(1) if n == 0:
(2) return 1
(3) else:
(4) r = floor(n/2)
(5) v = pow(x,r) * pow(x,r)
(6) if n % 2 == 1:
(7) v = v * x
(8) return v

Give a recurrence for the runtime of pow, then use the Master theorem to give a $\Theta$ bound on the worst-case runtime.
Problem 4 (20 points): Now consider the actual pseudocode from Problem Set 1:

double pow(x, n):
(1) if n == 0:
(2) return 1
(3) else:
(4) r = floor(n/2)
(5) p = pow(x,r)
(6) v = p * p
(7) if n % 2 == 1:
(8) v = v * x
(9) return v

Give a recurrence and runtime bound.