Math 368: Homework 5.0
Spring 2019

James Skon
skonjp@kenyon.edu

Due date: March 29, 9:00am
Submission Instructions:

• Each group must submit one solution to this assignment in class on the due date.

• Assignments must be neatly written or typed. If the grader cannot (easily) read an answer, she will give it a 0.

• Each group must submit only one solution. If multiple solutions are handed in, the grader will randomly discard all but one of them.

• Your submission must have the name of every group member clearly marked. If your name is not on it, you do not get credit.

• 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): 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 2 (20 points): Consider the following pseudocode (a variation of the code from Problem Set 1):

```java
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 3 (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.