2CS 2223 B17 - Project 3 Report - 12/6/17
Nikolas Gamarra

Executive summary:

Intro:

My program is structured by the 3 algorithms to be tested, a tester function, and the main function. The main function run when the program launches and checks for user input if provided and makes sure the input.txt file is in a valid format. Then it calls the tester. The tester times all 3 functions. When necessary it massages the data for the algorithm.

Time/Space Efficiency:

Key: W=capacity, n=number of inputs

Algorythm	Time Efficiency	Space Efficiency
Exhaustive Search	$O(n2^n)$	O(1)
Dynamic Programming	O(nW)	O(nW)
My Function (recursion)	$O(2^n)$	O(n)

The Exhaustive Search is $O(n2^n)$ time efficient because it is based on all permutations of the list and O(1) space efficient because it uses one large array of all the data that is passed in and makes no new data structures.

The Dynamic Programming is O(nW) time efficient because it has to traverse the 2-D array of n by W and O(nW) space efficient because it creates a 2-D array of n by W.

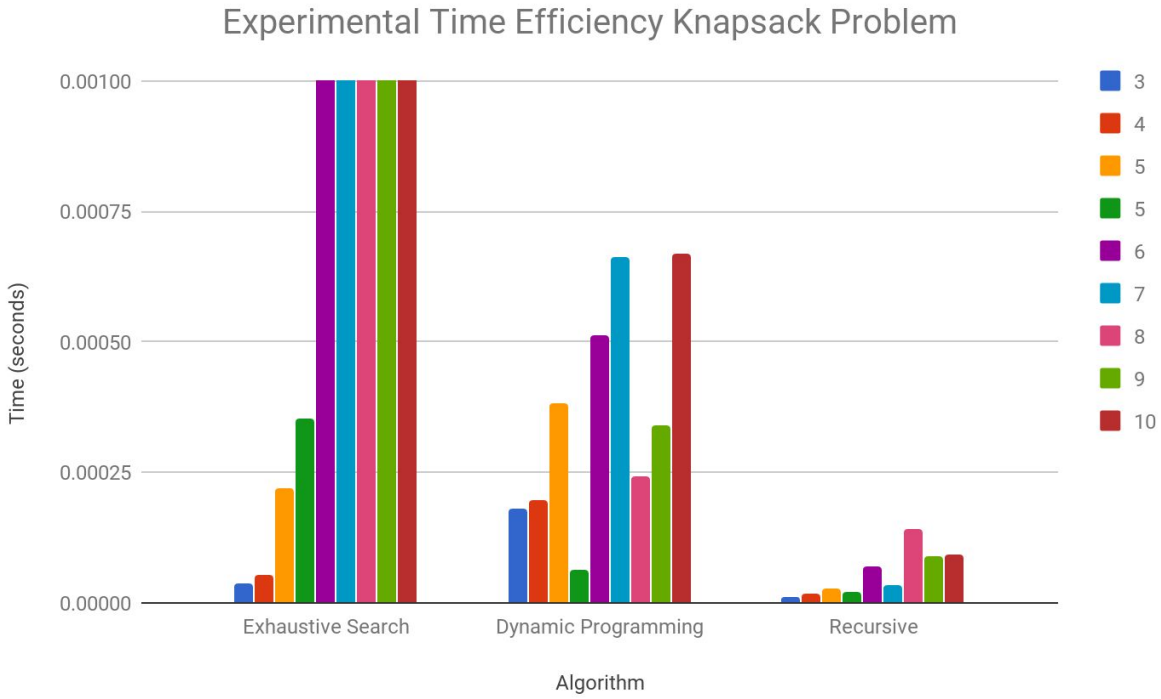
My Recursive function is $O(2^n)$ time efficient because of the calls to itself it will need to make and O(n) space efficient because it has to make n recursive calls.

Conclusion:

The best algorithm by far was the Recursive one I provided. It was able to provide accurate and relatively fastest solutions.

Data:

- Note:
  - The numbers in the legend represent the number of items in the list.
  - Max Y value truncated on graph due to extreme exponentiality. See table for values.
- Graph:



- Table**:**

Run	Algorithm:	Alg 1 - Brute Force	Alg 2 - Recursive	Alg 2 - Recursive
1	File:	**input-1.txt**	**input-1.txt**	**input-1.txt**
	Number of inputs (n):	**5**	**5**	**5**
	Value	**198**	**198**	**198**
	Time (seconds):	**2.19E-04**	**3.81E-04**	**2.74E-05**
2	File:	**input-2.txt**	**input-2.txt**	**input-2.txt**
	Number of inputs (n):	**5**	**5**	**5**
	Value	**65**	**65**	**65**
	Time (seconds):	**3.54E-04**	**6.25E-05**	**2.19E-05**
3	File:	**input-3.txt**	**input-3.txt**	**input-3.txt**
	Number of inputs (n):	**10**	**10**	**10**
	Value	**2.828427125**	**2.828427125**	**2.828427125**
	Time (seconds):	**9.570246935**	**0.001147270203**	**0.0001718997955**
4	File:	**input-4.txt**	**input-4.txt**	**input-4.txt**
	Number of inputs (n):	**3**	**3**	**3**
	Value	**1**	**1**	**1**
	Time (seconds):	**3.81E-05**	**1.81E-04**	**1.19E-05**
5	File:	**input-5.txt**	**input-5.txt**	**input-5.txt**

	Number of inputs (n):	**6**	**6**	**6**
	Value	**460**	**460**	**460**
	Time (seconds):	**2.96E-03**	**5.11E-04**	**6.79E-05**
6	File:	**input-6.txt**	**input-6.txt**	**input-6.txt**
	Number of inputs (n):	**9**	**9**	**9**
	Value	**377**	**377**	**377**
	Time (seconds):	**9.10E-01**	**3.41E-04**	**8.77E-05**
7	File:	**input-7.txt**	**input-7.txt**	**input-7.txt**
	Number of inputs (n):	**10**	**10**	**10**
	Value	**188**	**188**	**188**
	Time (seconds):	**1.00E+01**	**6.70E-04**	**9.39E-05**
8	File:	**input-8.txt**	**input-8.txt**	**input-8.txt**
	Number of inputs (n):	**8**	**8**	**8**
	Value	**95**	**95**	**95**
	Time (seconds):	**1.04E-01**	**2.42E-04**	**1.42E-04**
9	File:	**input-9.txt**	**input-9.txt**	**input-9.txt**
	Number of inputs (n):	**7**	**7**	**7**
	Value	**179**	**179**	**179**
	Time (seconds):	**1.08E-02**	**6.63E-04**	**3.31E-05**
10	File:	**input-10.txt**	**input-10.txt**	**input-10.txt**
	Number of inputs (n):	**4**	**4**	**4**
	Value	**250**	**250**	**250**
	Time (seconds):	**5.36E-05**	**1.98E-04**	**1.67E-05**