CSE 2320 - Homework 6

NAME:_____

Total points: 115/100 (points past 100 are bonus) Topics: Memoization, Greedy, Dynamic Programming (Knapsack: unbounded, 0/1, fractional)

P1 (4 pts) Given this solution information, for the **unbounded** Knapsack problem below, recover the choices that gave the optimal answer for knapsack capacity 19. Show your work (highlight or circle cells).

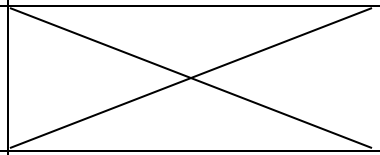
```
Item   |   A|   B|   C|   D|
Weight|   3|   4|   7|   8|
Value | the item values are hidden as they should not be used in recovering the solution.


        |  0|  1|  2|  3|  4|  5|  6|  7|  8|  9| 10| 11| 12| 13| 14| 15| 16| 17| 18| 19|
picked |   |   |   |  A|  B|  B|  A|  C|  D|  D|  A|  B|  B|  A|  C|  C|  D|  A|  B|  B|
```

Items picked for capacity 19: ……

P2 (61 pts) Given the item types below, solve the following problems. Fill in the answer in the table and show your work below.

Item:	A	B	C	D
Weight:	3	4	6	7
Value:	4	7	10	12

	Unbounded Knapsack	0/1 Knapsack	0/1, Fractional Knapsack
Dynamic Programming	$$: Items:	$$: Items:	
Greedy	$$: Items:	$$: Items:	$$: Items:

a) (20 pts) Solve the **unbounded** Knapsack problem. Recover the items in the solution and show how you did that (e.g. highlight or circle cells). Show your work as done in class.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A, 3, 4															
B, 4, 7															
C, 6, 10															
D, 7, 12															

b) (20 pts) Solve the **0/1** Knapsack problem below (15pts). Use a star to show if the current item was used or not in the solution (8pts). Recover the items in the solution and show how you did that (e.g. highlight or circle cells) (7 pts). Show your work as done in class.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A, 3, 4															
B, 4, 7															
C, 6, 10															
D, 7, 12															

c) (8 pts) What items will a Greedy algorithm based on the **ratio**, choose for an **unbounded** Knapsack problem of size 14? Show your work.

d) (8 pts) What items will a Greedy algorithm based on the **ratio**, choose for a **0/1** Knapsack problem of size 14? Show your work.

e) (5 pts) What items will a Greedy algorithm based on the **ratio**, choose for a **0/1 Fractional** Knapsack problem of size 14? Assume you have only one of each item. Show your work.

P3 (50 pts) Consider this recursive function:

```
int foo(int N){
    if (N <= 1) return 5;
    int res1 = 3*foo(N/2);
    int res2 =   foo(N-1);
    if (res1 >= res 2)
        return res1;
    else
        return res2;
}
```

a) (6 points) Write the recurrence formula for the TIME COMPLEXITY of this function, including the base cases for N>=0. You do NOT need to solve the recurrence. Remember not to confuse the time complexity of the function with what the function calculates.

b) (8 points) Draw the tree that shows the function calls performed in order to compute foo(5) (the root will be foo(5) and it will have a child for each recursive call.) Also show what each call returns by using an arrow pointing back from the child to the parent.

The foo.c file has the 'driver' code in the main function. Add to it the implementation of the following functions:

c) (10 pts) `int foo_iterative (int N)` - ITERATIVE solution of this code. (Stub provided, add functional code to it.)

d) (20 pts) … `foo_memoized(…. )` - the MEMOIZED solution. The function signature can be whatever you want. YOU MUST PRINT THE RECURSIVE and BASE CASE CALLS. They should show the N that the function was called for and the text should be indented based on the depth. This will show the tree of recursive calls.
The depth starts from 0 (the depth of the first call is 0).
[See file fact_2_rec.c for an example using an implementation of the factorial function that solves N! by multiplying the first half of the numbers and the second half and then the two results together. A line is printed for each recursive call to fact_2_rec and the deeper the recursive call is, the more to the right the text is indented. File sample_run__fact_2_rec.txt shows a sample run (with user input).]

e) (6 pts) `int foo_wrapper(int N)` - wrapper function that calls the `foo_memoized`. (Stub provided, add functional code to it.)

Your code should not show any errors with Valgrind. (6 points penalty of it does)
If main does not call your other functions, they cannot be graded and you get 0 points.

Files:

- foo.c – Starting code. You should write your code in this file.
- sample_run_stubs.txt - sample run of foo.c in the given version (with stubs) with input redirection from data1.txt .
- data1.txt - sample file to be used with input redirection

- sample_run_redirect.txt - sample run with input redirection after the necessary methods were added.
- sample_run_user.txt - sample run with user entered data (from keyboard) after the necessary methods were added

- For your reference: fact_2_rec.c, and sample_run__fact_2_rec.txt .

Remember to include your name at the top.

Write your answers in this document or a new document called **2320_H6.pdf**. It can be hand-written and scanned, but it must be uploaded electronically. Place **2320_H6.pdf** and **foo.c** in a folder called **hw6**, zip that and send it.