COP 3502 Section 1 Quiz #1 Version A (SLMP, Dynamic Memory Allocation)

Last Name: ___________________, First Name: ______________________

Lab Day/Time: ______________ , Lab Room (Circle): CB1-122 or HEC-118

1) (5 pts) What are the four functions associated with dynamic memory allocation that were taught in class? For each function, state how many parameters it takes in:

1. ___________________________ Number of Parameters: ________

2. ___________________________ Number of Parameters: ________

3. ___________________________ Number of Parameters: ________

4. ___________________________ Number of Parameters: ________

2) (10 pts) Complete the function below so that it returns 1 if there exist two different numbers in the array list that add up to exactly target, and 0 otherwise.. In order to earn full credit, the function must run in O(n) time.

```
// Pre-condition: list is length n, sorted, with unique values.
// Post-condition: Returns 1 if two different values in list add
//                 up to exactly target, and 0 if no such pair exists.
int add_to_target(int list[], int n, int target) {

    int i = 0, j = n-1;

    while ( _______________ ) {









     }

     _________________ ;
}
```

3) (12 pts) To encode a string, we can "count ahead" some number of letters (wrapping around with 'A' following 'Z' if necessary) For example, to encode "UCF" if we count ahead by 2 characters, we get "WEH". Using this system, there are 26 possible encodings (counting ahead by 0, 1, 2, .., 25). Write a function that takes in a string of uppercase letters and returns a dynamically allocated 2D array of char (array of strings), where each string stored is a possible encoding of the input parameter. The string in index i should be the one created by shifting the original string ahead by i characters. For the UCF example, the first 3 strings stored will be "UCF", "VDG" and "WEH". Your returned array should have 26 char*'s each of which are pointing to an array of the appropriate size storing the corresponding encoded string.

```
char** eachShift(char* word) {









}
```

4) (8 pts) Complete the code below so that it asks the user to enter a string of uppercase characters, calls the eachShift function to get all possible encodings of the string, prints each of these encodings out, and then frees the dynamically allocated memory.

```
#include <string.h>
#include <stdio.h>
int main() {
    char word[100]
    scanf("%s", word);





}
```