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

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. malloc — Number of Parameters: 1
2. calloc — Number of Parameters: 2
3. realloc — Number of Parameters: 2
4. free — Number of Parameters: 1

Grading: 1 pt for each function name, last point if ALL FOUR number of parameters are correct.

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 ( i<j ) {                                  // 1 pt

        if (list[i]+list[j] < target)                // 2 pts
            i++;                                     // 1 pt

        else if (list[i]+list[j] > target)           // 2 pts
            j--;                                     // 1 pt

        else                                         // 1 pt
            return 1;                                // 1 pts
     }

     return 0;                                       // 1 pt
}
```

Grading note: order of the if clauses can be any order. Last clause is worth 2 total instead of 3…

Max grade for $O(n^2)$ is 6 out of 10. (1 pt setup, 1 pt outer loop, 2 pts inner loop (1 pt only if allows repeat val), 2 pts check inside and return 1, 1 pt return 0 outside

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) {

    char** res = calloc(26, sizeof(char*));      // 3 pts
    int len = strlen(word);
    for (int i=0; i<26; i++) {                   // 1 pt
        res[i] = calloc(len+1, sizeof(char));    // 3 pts
        for (int j=0; j<len; j++)                    // 1 pt
            res[i][j] = ((word[j]-'A')+i)%26 + 'A';  // 2 pts
        res[i][len] = '\0';                          // 1 pt
    }

    return res;                                      // 1 pt

}
```

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);

    char** grid = eachShift(word);               // 2 pts

    for (int i=0; i<26; i++)                     // 1 pt
        printf("%s\n", grid[i]);                 // 1 pt

    for (int i=0; i<26; i++)                     // 1 pt
        free(grid[i]);                           // 2 pts
    free(grid);                                  // 1 pt

}
```