3502-1

Assignment # 1

1) Suppose we have an array to store all of the holiday presents we have purchased for this year. Now that the holidays are over and all the presents have been given out, we need to delete our list. Our array is a dynamically allocated array of structures that contains the name of each present and the price. The name of the present is a dynamically allocated string to support different lengths of strings. Write a function called delete_present_list that will take in the present array and free all the memory space that the array previously took up. Your function should take 2 parameters: the array called present_list and an integer, num, representing the number of presents in the list and return a null pointer representing the now deleted list. (Note: The array passed to the function may be pointing to NULL, so that case should be handled appropriately.)

```
struct present {
      char *present_name;
      float price;
};

struct present* delete_present_list(struct present* present_list, int num)
```

2) Suppose we have a singly linked list implemented with the structure below and a function that takes in the head of the list. typedef struct node { int num; struct node* next; } node; int whatDoesItDo (node * head) { struct node * current = head; struct node * other, *temp; if (current == NULL) return head; other = current->next; if (other == NULL) return head; other = other->next; temp = current->next; current->next = other->next; current = other->next; if (current == NULL) { head->next = temp; return head; } other->next = current->next; current->next = temp; return head; } If we call whatDoesItDo(head) on the following list, show the list after the function has finished. head -> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7