Name:…… Q12

CS2123 Data Structures

Q1. (10pt) Suppose we have the following declarations and dynamic memory allocation statements. (We assume that all memory allocations are successful so we are not checking if the returned address is NULL or not after each malloc()).

```
typedef struct point {
   int x;
   struct point *next;
} myDataT;

myDataT *a, *b;
```

```
typedef struct point {
   int x;
   struct point *next;
} *myDataTptr;

myDataTptr a, b;
```

```
a = (myDataT *) malloc(sizeof(myDataT));
a->x = 5;
a->next = (myDataT *) malloc(sizeof(myDataT));
a->next->x = 8;
a->next->next = (myDataT *) malloc(sizeof(myDataT));
a->next->next->x = 10;
a->next->next->next = NULL;
```

a. (2pt) Show how the linked list conceptually looks like after the above code.

b.(3pt) Write a function `Display(myDataT *h)` that can print the values stored in the list h.

b. (5pt) As you see, the values in the above linked list are sorted. Now suppose we create a new element pointed by `b` as follows

```
b = (myDataT *) malloc(sizeof(myDataT));
b->x = 9;
b->next=NULL;
```

and we would like to add this to the right position in a sorted list (like the one pointed by `a)`. Your solution should be general enough to work with any sorted list!

Write the necessary code segments/statements to ***first search and find the right position*** for the new element, and ***then add the new element at that position*** in the sorted list so that the list will be still sorted. If needed, declare new variables and/or pointers here.