PRACTICE PROBLEMS – Linked List

Q1)

```
struct Node {
   char data;
   Node* next;
};
int Mutate(Node* head, char d){
          Node* ptr = head;
          bool check = false;
   Node* last_start = NULL, *last_end = NULL, *pre = NULL;
   while(ptr != NULL){
      if(ptr->data == d){
         if(last_start){
            Node* temp = last_start->next;
            last_start->next = last_end->next;
            pre->next = temp;
            last_end->next = ptr;
            last_start = pre;
         }
         last_start = pre;
         last_end = ptr;
         while (ptr->data == d)
         {
            last_end = ptr;
            ptr = ptr->next;
         }

      }
      pre = ptr;
      ptr  =ptr->next;
   }
   Node* temp = last_start->next;
   last_start->next = last_end->next;
   pre->next = temp;
   last_end->next = ptr;
   last_start = pre;
   return 0;
}
```

Train = A -> C -> B -> C -> C -> A -> D -> C -> C -> C -> B -> NULL

What will be the resultant List train after passing Mutate(List, 'C');

What is Time Complexity?

Q2)

```
void func(Node* head){
   bool ex;
   Node* iNode = head;
   Node* cNode = NULL, *pNode = NULL;
   do{
      ex = false;
      while(cNode = iNode && cNode->next != pNode){
         iNode = iNode->next;
         if(cNode->data < iNode.data){
            int temp = cNode->data;
            cNode->data = iNode->data;
            iNode->data = temp;
            ex = true;
         }
      }
      pNode = iNode;
      iNode = head;
   }while(ex);

}
```

head = 9 -> 8 -> 7 -> 1 -> 5 -> NULL

What will be the list after passing func(head)?

Q3)

Write a print function that prints a node, then skips two nodes to print another node. For example, for the list

Head = 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 ->8

The function prints

1 -> 4 -> 7

Q4)

What will be the linked list after the following function?

Head = 6 -> 13 -> 11 -> 6 -> 13 -> 11 -> 5 -> 4 -> 5 -> 10

```
struct Node{
   int data;
   Node* next;
}
void func(Node* head){
   Node* ptr1, *ptr2, *nod;
   ptr1 = start;
   while(ptr1 && ptr1->next){
      ptr2 = ptr1;
      while (ptr2->next)
      {
         if(ptr1->data == ptr2->next->data){
            nod = ptr2->next;
            ptr2->next = ptr2->next->next;
            delete nod;
         }
         else{
            ptr2 = ptr2->next;
         }
      }
      ptr1 = ptr1->next;
   }
}
```

Q5)

Write a function in C++ which counts the duplicates in a linked List. For example, the list

7 -> 9 -> 7 -> 8 -> 9 -> NULL **will return 2.**

Q6)

```
int func(Node* head) {

   Node* ptr1 = head;
   Node* ptr = head;

   while (ptr != NULL && ptr->next != NULL) {
      ptr = ptr->next->next;

      ptr1 = ptr1->next;
   }
   return ptr1->data;
}
```

What will be the output if List = 1 -> 4 -> 5 -> 2 -> 9 -> 5 -> NULL is passed in function?