Department of Computer Science

CS 201 – Data Structures

Mid Term I (Spring 2014)

Instructors: Adnan Waheed & Sara Saleem

February 24, 2014

Total Marks: 40	**Time Allowed: 60 minutes**

Instructions:

(1) Understanding the question is part of exam. NO QUERIES WILL BE ENTERTAINED.
(2) Use answer sheet for rough work and provide solutions in the given space.
(3) Write neat & clean.
(4) Use permanent ink pens only.
(5) Poor programming approaches will decrease your marks.
(6) Think about the boundary conditions.

Roll No. _____Name:_____Section: _____

Question No.	**1**	**2**	**3**	**4**	**5**	**6**	**Total**
Marks	***5***	***5***	***5***	***5***	***5***	***5***	***30***

GOOD LUCK ☺

Question 1:	Marks 5

The following program is supposed to allocate nodes, put them into a linked list, print them out and then destroy them. Fix the bugs in the program and rewrite in a clean hand writing

```
struct linkedList{
		int number;
		linkedList* next;
};
		void AddNode()
			{
		  linkedList* head=NULL;
		  linkedList* current;
		  for(int i=0;i<10;i++){
		  current = new linkedList;
		  current->number = i;
		  current.next = head;
		  head = current;
				}
				while(head->next!=NULL){
					cout << head->number << endl;
					current = head->next;
					delete current;
					head = current;
				}
				return 0;
		}
```

```
struct linkedList{
		int number;
		linkedList* next;
};
		int Add() {
				linkedList* head=NULL;
				linkedList* current;
				for(int i=0;i<10;i++){
					current = new linkedList;
					current->number = i;
					current->next = head;
					head = current;
				}
				while(head!=NULL){
					cout << head->number << endl;
					current = head->next;
					delete head;
					head = current;
				}
					system("pause");
				return 0;

			}
```

Question 2:	**Marks 5**

```
struct myList{
		int data;
		myList* next;
};
int main() {
		myList* head;
		myList* cur;
		myList* previous=NULL;
		for(int i=0;i<4;i++){
				head=new myList;
				head->data=0;
				head->next=previous;
				for(cur=previous;cur!=NULL;cur=cur->next)
						head->data+=1+2*cur->data;
				previous=head;
		}
		while(previous!=NULL){
				cout<<previous->data<<endl;
				cur=previous;
				previous=previous->next;
				delete cur;
		}
		return 0;
}
```

Write output for the above code you have to do your rough work in the space given below.

Output: 13 4 1 0

Question 3:	Marks 5

The function below should insert a value as the head of a given *UNORDERED* linked list. Be careful to first search for the given value. The value is inserted only if it does not exist in the list. If the value exists, the function does nothing.

```
struct Node{
              int data;
              Node* next;
};
```

The function prototype is given. Implement the function.

```
void insert(Node* &head, int value);
```

Note: You are not allowed to change the prototype of function for each incorrect statement you will get -1, be careful while solving this question

```
void insert(linkedList* &head, int value){
              linkedList* cur;
              for(cur=head; cur!=NULL; cur = cur->next)
                  if(cur->data == value)
                    return;
              cur  = new  linkedList;
              cur->data = value;
              cur->next = head;
              head = cur;
}
```

Question 4:	Marks 5

Consider following Linked List:

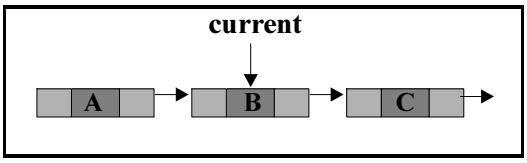
```
struct Node {
                int data;
                Node *next;
class LList {
public:
                LList();
                ~LList();

private:
                Node *head;
                int size;
};

LList::LList () {
                head = NULL;
                size = 0;
}
```

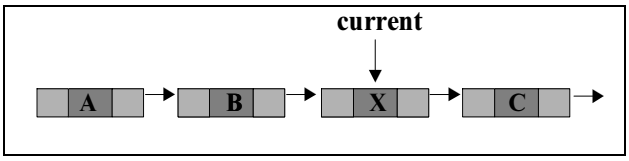
Implement the destructor `LList::~LList ()`**. Each mistake will lead to -1**

```
LList::~LList ()
{
                while(head != NULL) {
                                Passenger *cur = head;
                                head = head->next;
                                delete cur;
                }
}
```

Question 5:	Marks 5

Write some code segment that inserts a node immediately after the "current" node, as depicted below:





List before inserting Node with String "X"

```
Node *p = new Node("X");
p->next = current.next;
current->next = p;
current = p;
```

Question 6:	Marks 5

The Linked List contains certain Nodes containing data of integer type, You cannot use any other type of Array or data structure neither doubly Linked List. Use only singly linked list, Implement a function which can reverse the List.

```
Node* ReverseList( Node ** List )
{

	Node *temp1 = *List;
	Node * temp2 = NULL;
	Node * temp3 = NULL;

	while ( temp1 )
	{
		*List = temp1; //set the head to last node
temp2= temp1->pNext; // save the next ptr in temp2
		temp1->pNext = temp3; // change next to privous
		temp3 = temp1;
		temp1 = temp2;
	}

	return *List;
}
```