Department of Computer Science

CS 201 – Data Structures

Mid Term II (Spring 2014)

Instructors: Adnan Waheed & Sara Saleem

April 8, 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**	**7**	**8**	**Total**
Marks	***3***	***5***	***2***	***5***	***5***	***6***	***5***	***9***	***40***

GOOD LUCK ☺

Question 1:	**Marks 3**

Consider following Node structure

```
struct Node
{
	int value;
	Node *Next;
};
```

Write a function which accepts a linear linked list **List** and converts it to a circular linked list. Where **List** is a pointer to the front of the list.

```
Node *Convert( Node *List)
{

if(List==NULL)
       return NULL;
Node *curr = List;
while (curr->Next!=NULL)
{
       curr= curr->Next;
}
curr->Next=List;
return curr;
}

}
```

Question 2: **Marks 5**

Consider following diagram

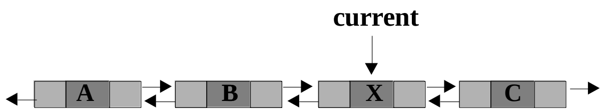

Figure: List *before* deleting Node with String "X"

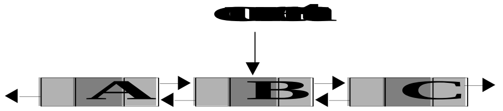

Figure: List *after* deleting Node with String "X"

Write some code segment for the following situation: Delete the "current" node from a double-linked list, as indicated in the picture below. You do not have to provide a complete method or class, nor do you need to worry about special cases such as deleting the last, first, or only Node in a list. Note: each node has three fields, prior, next, and data, and the constructor of the Node class takes one argument of type char

```
current.prev.next = current.next;
current.next.prev = current.prev;
Node *p = current;
current = current.prev;
delete p;
```

Question 3:	**Marks 2**

What will be the output of following code?

```
int [] values = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19 };
Stack s = new Stack( );

for (int 1 = 0; i < values.length; i++)
 s.push( values[ i ] );

int n = 25;
for (int i = 0; i < 4; i++)
 {
 n += s.pop( );
 }
 for (int i = 0; i < 2; i++)
 {
 n -= s.pop( );
 }
 Cout<<”\n”<<endl;
```

69

Question 4:	**Marks 5**

Using only the operations of the stack, write a function that determines if a string is a palindrome (i.e. reads the same backward and forward; e.g. “level”). e.g. “maham” is a palindrome your function should return true.

The prototype for this function is given below.

```
bool isPalindrome(String theString);
```

Question 5:	**Marks 5**

Write a function in C++ to insert in a circular Queue containing

Players information**(represented with the help of array of structure PLAYER).**

Consider following definition of Node

```
struct PLAYER
{
int PID;
char Pname[20]
NODE *Next
};
```

```
void QUESINSERT(PLAYER P[ ], int Front, int &Rear, int N])
{
        if (Friont-1=Rear)||(Front==0 && Rear=N-1)
        cout<<"Overflow!! Queue full"<<end;
        else
        {
        Rear=(Rear+1)%N;
        cout<<"Enter Player ID"
        cin>>P[Rear].PID;
        cout<<"Enter Player name";
        gets(P[Rear).Pname;
        }
}
```

Question 6:	**Marks 3+3**

Convert the expression into postfix notation and then verify if it is correct or not, direct answers for both conversion and verification will leads to negative marking for verification take dummy values for each variable.

a+b*(c/d-e)%(f+g*h)-i

Postfix notation	**Verification**
abcd/e-%fgh*+*+i-	

Question 7:	Marks 5

Write a non-recursive function for post order traversal for following prototype

```
void postOrderIterative(Node* root)
```

```
      void postOrderIterative(struct Node* root)
      {
          if (root == NULL)
          return;
        do
        {
            while (root)
          {
              if (root->right)
              push(stack, root->right);
              push(stack, root);
              root = root->left;
          }
             root = pop(stack);

         if (root->right && peek(stack) == root->right)
          {
            pop(stack);  // remove right child from stack
            push(stack, root);  // push root back to stack
            root = root->right; // change root so that the right
                      // child is processed next
          }
          else  // Else print root's data and set root as NULL
          {
            printf("%d ", root->data);
            root = NULL;
          }
        } while (!isEmpty(stack));
}
```

Question 8:	**Marks 2.5+1.5+1.5+0.5+3=9**

Draw the binary search tree that results from inserting the integers 57, 85, 35, 9, 47, 20, 26, 99, 93, 10, 50, 51, 52 starting with 57 and ending with 1.

1. What is the pre-order traversal of your tree
2. What is the post-order traversal of your tree
3. What is the in-order traversal of your tree
4. **True / False** (circle one). Inserting into a binary search tree always inserts a leaf node

5. Write an iterative (non-recursive) **BinarySearchTree** method named min that returns the smallest value in a binary search tree.

You are not expected to do this ☺