Prof. D. Dahiya, Dr. Nilin, Dr. Yashwant

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

MID SEMESTER EXAMINATION 2015

B.Tech. II Semester (CSE,IT,ECE,BI)

COURSE CODE: 10B11CI211 MAX. MARKS: 30

COURSE NAME: Data Structures

COURSE CREDITS: 4 MAX. TIME: 2 HRS

Note: All questions are compulsory. Make suitable assumption if any.

Section A **(Marks: 6 [6*1])**

1. If you are using C language to implement the heterogeneous linked list, what pointer type will you use? And why?
2. Write the time and space count for the following code fragments?

```
for (i = 0; i < N; i++) {
   for (j = 0; j < N; j++) {
      sequence of statements
   }
}
for (k = 0; k < N; k++) {
sequence of statements
}
```

3. What does the following function do for a given Linked List ?

```
void fun2(struct node* head)
{
 if(head== NULL)
  return;
 printf("%d  ", head->data);
 if(head->next != NULL)
 fun2(head->next->next);
printf("%d  ", head->data);
}
```

4. Mention the time complexity (frequency count) in Big O notation for the following code fragments?

```
for (int i = N; i > 0; i /= 2)
for (int j = 1; j <= i; j *= 2)
                printf("o");
```

5. Explain the functionality (time count) of below recursive functions.

```
void fun1(int n)
{  int i = 0;
 if (n > 1)
  fun1(n-1);
```

```
for (i = 0; i < n; i++)
 printf(" * ");}
```

6. How will you find the nth node from the end of a singly link list in a single pass.

Section B **(Marks: 9 [3*3])**

1. Given a linked list that contains alphabets. The alphabets may be in upper case or in lower case. Write an algorithm/program to create two linked lists one which stores upper case alphabets and the other that stores lower case characters.
2. Write an algorithm/program to input an n digit number. Now break this number into its individual digits and then store every digit in a separate node thereby forming a linked list. For example, if you enter 1234, now there will be four nodes in the list containing nodes with values 1, 2, 3, 4.
3. Write an algorithm/program to delete the last occurrence of a given character in a linked list.

Section C **(Marks: 15 [5*3])**

1. Write the C function *Replicate* whose header is given below. *Replicate* adds new nodes to *list* so that nodes are replicated the number of times specified by parameter *count*. For example, if *list* is represented by

   ("apple","cat","xray")

   Then the call Replicate(list,3) should change *list* as shown below.

   ("apple","apple","apple","cat","cat","cat","xray","xray","xray")

   Note that *list* is **not** passed by reference. **Assume that Node has been modified so that it can be used for a doubly linked list.**

```
void Replicate(Node * list,int count)
// precondition: list = a1, a2, ... an, 1 < count
//                 doubly linked, NO header node,
// postcondition: list  = a1, a1, ... a1, ... an, an, ... an
//              where each element appears count times
```

2. Given an integer linked list of which both first half and second half are sorted independently. Write a C function to merge the two parts to create one single sorted linked list in place [do not use any extra space].

   Sample test cases:

   Input: List 1:1->2->3->4->5->1->2; Output: 1->1->2->2->3->4->5

   Input 2: 1->5->7->9->11->2->4->6; Output 2: 1->2->4->5->6->7->9->11

3. Given two linked lists, write a C function to insert nodes of second list into first list at alternate positions of first list.

   eg. 1->5->7->3->9

   6->10->2->4,

   the first list should become 1->6->5->10->7->2->3->4->9 and second list should become empty.

   The nodes of second list should only be inserted when there are positions available.

   If the first list is 1->2->3 and second list is 4->5->6->7, then first list should become 1->4->2->5->3->6 and second list to 7.