Bapuji Educational Association (Regd.)
Bapuji Institute of Engineering and Technology
Affiliated to Visvesvaraya Technological University, Belgavi, Accredited by NBA and NAAC with 'A' grade,
Recognized by UGC under 2(F) and 12(B)

Computer Science and Engineering

Batch: 2018_2022 Academic Year: 2019 - 2020 Date: Nov 5, 2019

Concept design: Prof. Naveen Kumar K R

SUBJECT: Data Structures and Applications [18CS32]

Topic: Write Codes for Diagrams

In recent times, educators have been focusing on innovative teaching techniques to enhance the learning experience for students in various fields. As part of this trend, groundbreaking teaching methods have been implemented in the computer science program aimed at improving the programming skills of 3rd semester students. In particular, the focus has been on enhancing their knowledge of linked lists, a fundamental data structure in computer science.

The teaching approach adopted involved a detailed analysis of the stacks and heaps of memory layouts, providing students with a deeper understanding of the underlying concepts. This approach allowed students to appreciate the importance of linked lists as a way of organizing data in a variety of fields. An examination was then conducted to evaluate the students' understanding, where they were given sample questions accompanied by diagrams of memory layouts. The students were required to write code based on the diagrams, thereby applying their newfound knowledge to practical scenarios.

The approach of using diagrams to teach programming concepts proved highly successful, not only improving students' programming skills but also boosting their confidence in working with linked lists. By utilizing innovative teaching methods, the program delivered a rich learning experience that enhanced the theoretical knowledge of students, while also emphasizing the importance of practical problem-solving skills. As such, the approach could be considered a model for the future, demonstrating the benefits of using cutting-edge teaching techniques to enhance student learning and development.

After the completion of the course on innovative teaching methods for enhancing programming skills with a focus on linked lists, the CSE students will be able to:

1. Develop a comprehensive understanding of the concepts related to linked lists, memory layouts and programming skills.
2. Apply the knowledge and skills learned in creating efficient and optimized programs using linked lists.
3. Analyze and understand memory layout concepts to design more efficient programs.
4. Develop the ability to design and implement data structures for complex programming scenarios using linked lists.
5. Apply the skills and knowledge to real-world practical scenarios.
6. Demonstrate confidence while working with linked lists and other related concepts.
7. Demonstrate an ability to apply problem-solving and critical thinking skills while working with linked lists and related concepts.
8. Communicate effectively with peers and instructors regarding the programming concepts and linked lists.
9. Collaborate and work in teams to develop efficient and optimized programs using linked lists.
10. Evaluate and assess programming solutions using linked lists in order to identify possible

47/60 Waveen 28/11/19

Bapuji Educational Association (Regd.)
Bapuji Institute of Engineering and Technology, Davangere-577004
Department of Computer Science and Engineering

Subject: Data Structures & Applications (18CS32) 2019

WRITE CODES WITH DIAGRAM

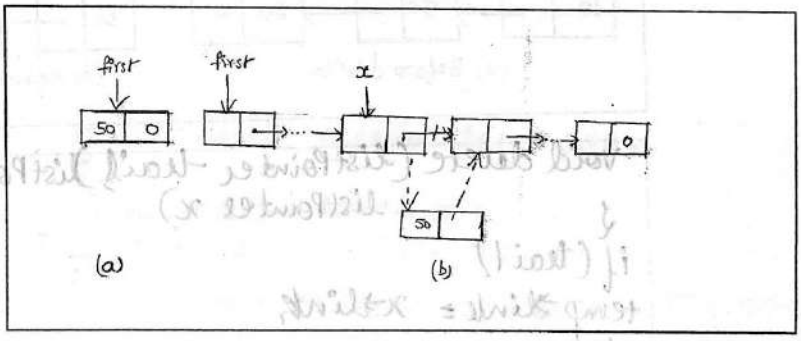
Performance Assessment for Third Semester Students A/B-Section

Linked List

Prepared by Mr. Naveen Kumar K R & Mr.Naveen H M

Name:	SURAKSHA.V. KANDI	Date: 05 | Nov | 19
USN:	4BD18CS110	

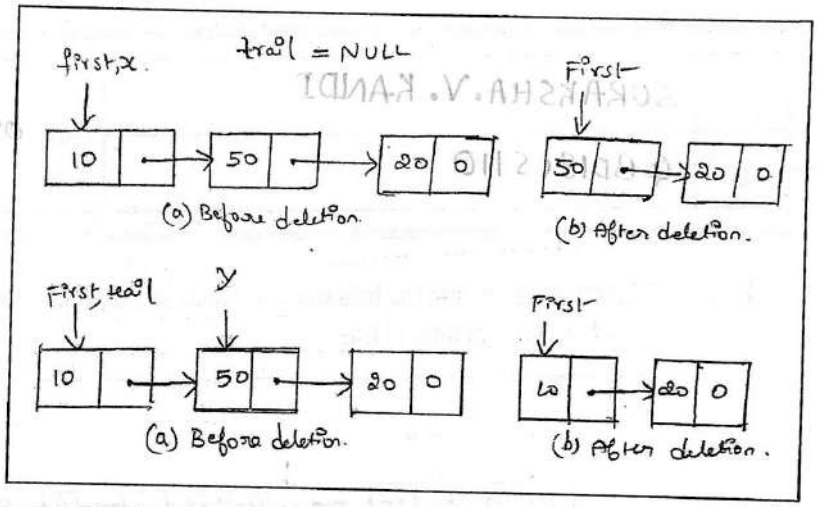
1. Write C functions to perform the following operations on singly linked list.
   a. Simple insert into front of list.




```
void insert (listPointer *first, listPointer x)
{
listPointer temp;
malloc (temp, sizeof (temp));
temp->data = 50;
if (*first);
```

```
temp->link = x->link
x->link = temp
}else{
temp->link = NULL
*first = temp
}
```

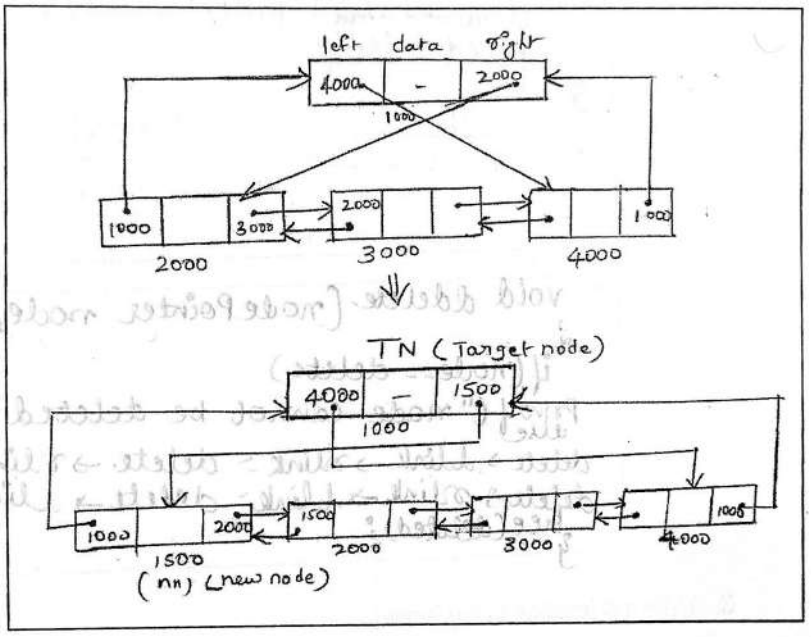
b. Deletion from a list




```
void delete (listPointer trail, listPointer *first
             listPointer x)
{
if (trail)
temp->link = x->link
else
*first = (*first)->link
free(x);
}
```

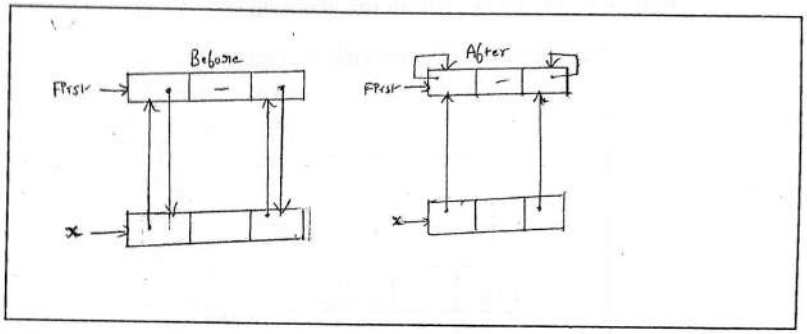
2. Write C functions to perform the following operations.

   a. Insertion into a Doubly Linked circular list.




```
void dinsert(nodePointer node,
{
newnode->llink = node;
newnode->rlink = node->rlink;
newnode->llink->rlink = newnode;
node->rlink = newnode;
}
```

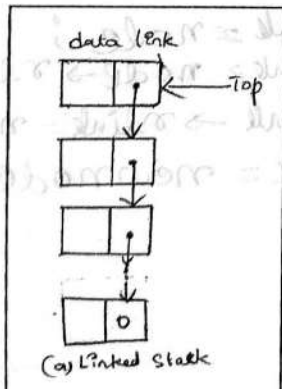
b. Deletion from a Doubly Linked circular list.




```
void ddelete (nodePointer node, nodePointer delete)
{
  if (node == delete)
    Printf ("node cannot be deleted ");
  else
    delete -> llink -> rlink = delete -> rlink;
    delete -> rlink -> llink = delete -> llink;
    free (deleted);
}
```

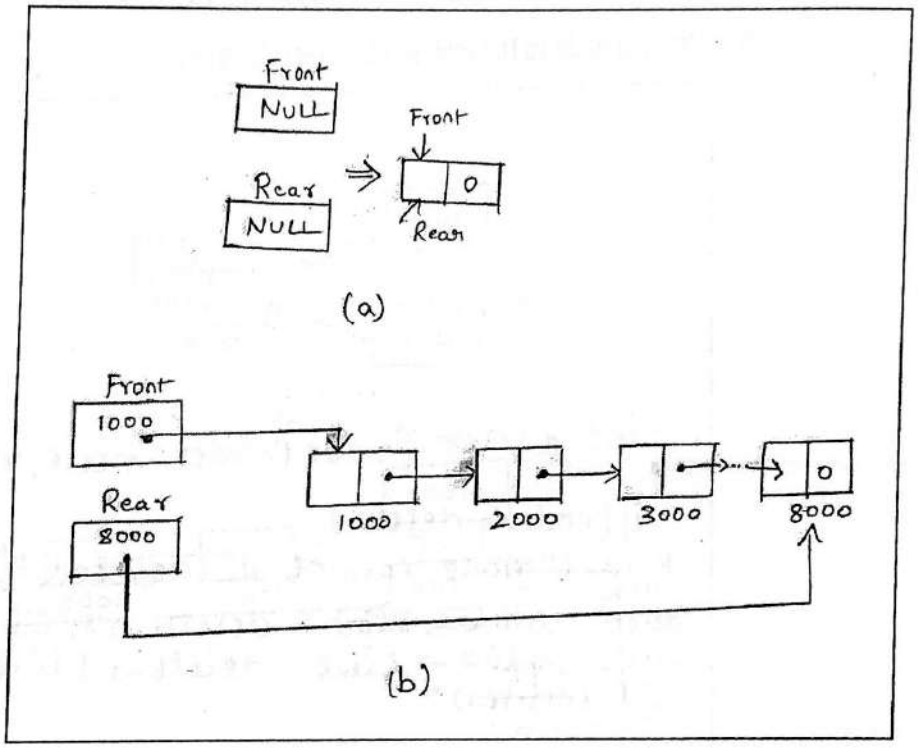
3. Write a C function to implement..

a. Linked Stacks




```
void push (int i, element data)
{
  StackPointer temp;
  malloc (temp, size of (temp));
  temp -> data = item;
  temp -> link = top [i];
  top [i] = temp;
}
```

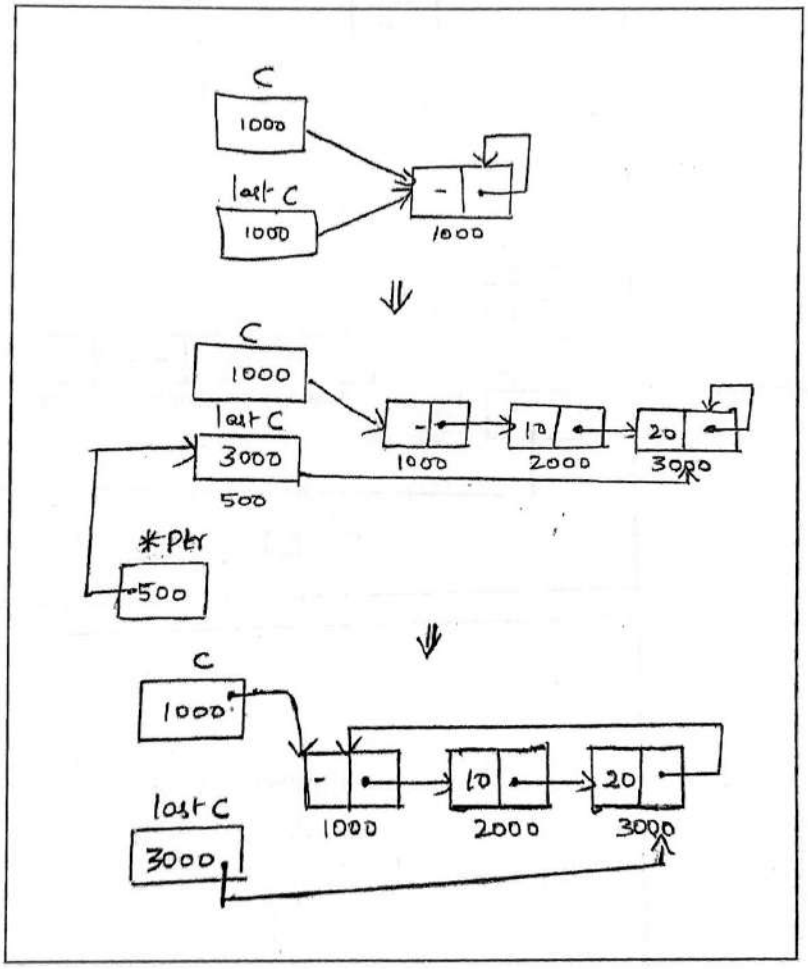
b. Linked Queues



WRITE CODE HERE

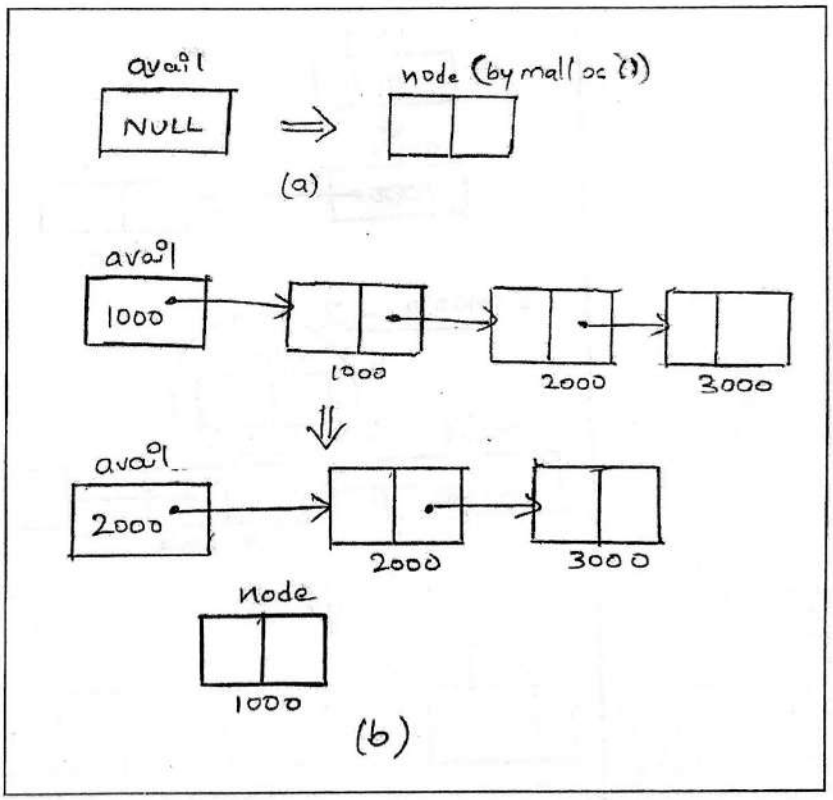
4. Write C functions to perform the following operations.

   a. Attach a node to the end of a list (Circular List)



WRITE CODE HERE	WRITE CODE HERE

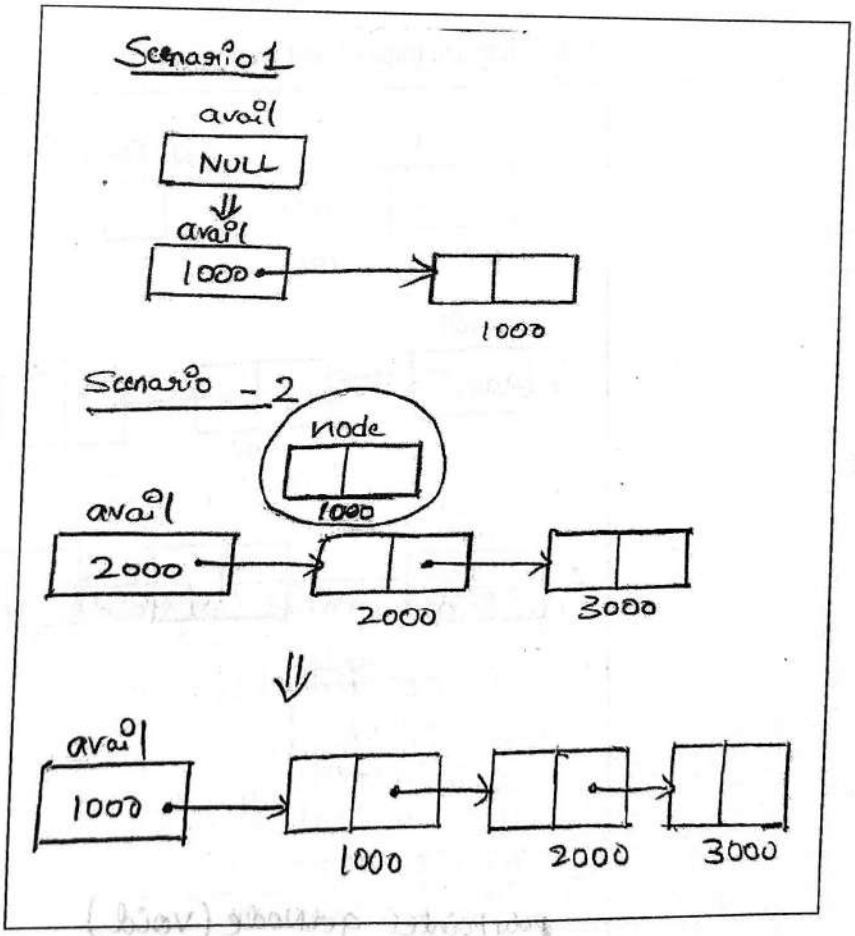
b. getNode function (replacement for malloc())




```
polyPointer getNode(void)
polyPointer node;
```

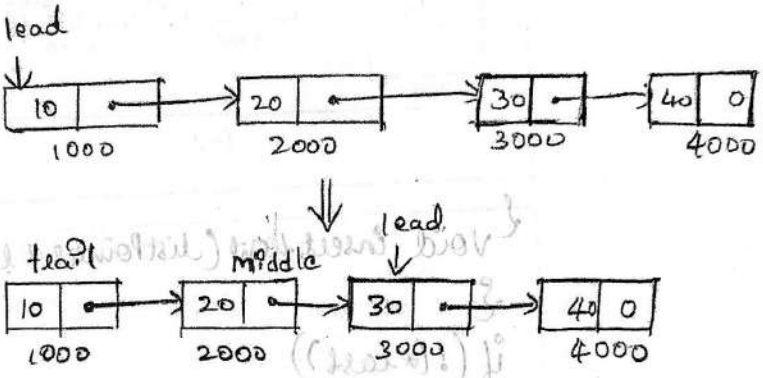
```
if(avail){
node=avail;
avail = avail->link;
}
else
malloc(node, sizeof(node));
}return node;
```

c. retNode function (replacement for free())



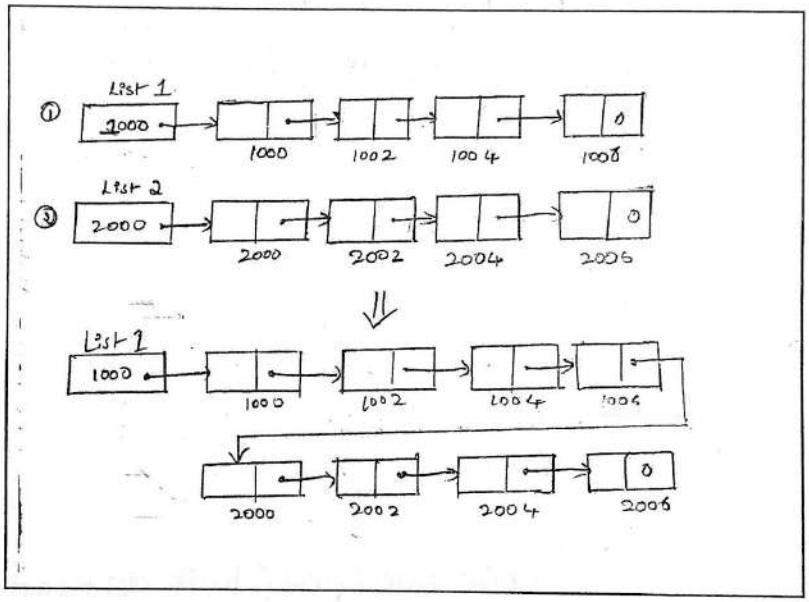
```
void retNode (polyPointer node)
{
 node -> link = avail;
  avail = node;
}
```

d. Inverting a Singly Linked List




```
listPointer invert (listPointer lead)
{
listPointer middle, trail;
 middle = NULL;
 while (lead);
 trail = middle;
 middle = lead;
 lead = lead -> link;
 middle -> link = trail;
}
return middle;
}
```

f. Concatenating Singly Linked List




```
listPointer Concatenation (listPointer ptr1,
                           listPointer ptr2);
{
if(!ptr1) = return ptr2
if(!ptr2) = return ptr1
temp->link = ptr2
}
```

-2-

1. Naveen H M 2. Naveen Kumar K R
Course Coordinator (Faculty Incharge)

Program Coordinator (HOD)