**Lab 3:**

**In lab task 1:**

**a)**

int insertNodeAtStart(struct node \*\* head)

{

/\*\* Complete this function \*\*/

struct node\* new\_node = (struct node\*) malloc(sizeof(struct node));

inputNodeData(new\_node);

new\_node->next = (\*head);

(\*head) = new\_node;

return;

}

**b)**

int insertNodeBefore(struct node \* head, int idx)

{

/\*\* Complete this function \*\*/

struct node \* temp= head;

if(isListEmpty(head)) /// if currently the list is empty return -1

{

return(-1);

}

struct node \* new\_node =(struct node \*) malloc(sizeof(struct node));

inputNodeData(new\_node);

if (idx == 0)

{

printf("\n data is inserted at the start.");

inputNodeAtStart(head);

return(-1);

}

for (int i=1; i<idx-1; i++)

{temp= temp->next;}

struct node \*aft=temp->next;

temp->next= new\_node;

new\_node->next=aft;

return (0);

}

**In lab task 2:**

**a)**

void deleteNodeFromStart(struct node \*\* head)

{

if (\*head == NULL)

return NULL;

struct node\* temp = \*head;

temp->next = \*head;

free(temp);

}

**b)**

int deleteNodeAfter(struct node \* head, int idx)

{

struct node \*temp=head;

struct node \*nex= head->next;

int delete, count = 0;

if(head == NULL)

{

printf("\n there's no node for deletion\n");

}

else

{delete = idx;

while (count != delete)

{

temp = temp->next;

nex = nex->next;

count++;

}

temp->next = nex->next;

free (temp);

}

**Post lab:**

int saveListToFile(struct node \* head, FILE \* fptr)

{

if(isListEmpty(head))

{

printf("\n list is empty");

return (0);

}

struct node\_d \* temp = head;

if(head == NULL)

return(-1);

do

{

fwrite(&(temp->data), sizeof(struct employee), 1, fptr);

temp = temp->next; /// scroll to the end of the list

}

while(temp != head);

return(0);

}