Name: Ameya Barapatre

Roll No: 06

```
#include<stdlib.h>
#include<stdio.h>
struct Node{
 int data;
 struct Node *next;
};
void deleteStart(struct Node** head){
 struct Node* temp = *head;

 // If head is NULL it means Singly Linked List is empty
 if(*head == NULL){
 printf("Impossible to delete from empty Singly Linked List");
 return;
 }

 // move head to next node
 *head = (*head)->next;
 printf("Deleted: %d\n", temp->data);
 free(temp);
}
void insertStart(struct Node** head, int data){
```

```
 // dynamically create memory for this newNode
 struct Node* newNode = (struct Node*) malloc(sizeof(struct Node));

 // assign data value
 newNode->data = data;
 // change the next node of this newNode
 // to current head of Linked List
 newNode->next = *head;
 //re-assign head to this newNode
 *head = newNode;
 printf("Inserted %d\n",newNode->data);
}
void display(struct Node* node){
 printf("\nLinked List: ");
 // as linked list will end when Node is Null
 while(node!=NULL){
 printf("%d ",node->data);
 node = node->next;
 }
 printf("\n");
}
int main()
{
 struct Node* head = NULL;
```

```
insertStart(&head,100);

insertStart(&head,80);

insertStart(&head,60);

insertStart(&head,40);

insertStart(&head,20);


display(head);


deleteStart(&head);

deleteStart(&head);

display(head);


return 0;

}
```

```
/tmp/sGA8VCfh5w.o
Inserted 100
Inserted 80
Inserted 60
Inserted 40
Inserted 20

Linked List: 20 40 60 80 100
Deleted: 20
Deleted: 40

Linked List: 60 80 100


=== Code Execution Successful ===
```