## DS LAB-A WEEK-2

## -20103153\_AVNI\_B6

```
}
Q1.
#include<iostream>
using namespace std;
                                                      void display()
class Node
                                                        Node* temp = head;
                                                        while(temp != NULL)
public:
                                                          cout << temp->data << " ";</pre>
    int data;
    Node *next;
                                                          temp = temp->next;
    Node(int info)
                                                        cout << endl;</pre>
        data=info;
                                                  };
        next=NULL;
                                                  int main() {
    }
                                                    List 1;
};
                                                    // inserting elements
                                                    1.insert(6);
class List
                                                    1.insert(9);
                                                    l.insert(1);
public:
                                                    1.insert(3);
    Node *head;
                                                    1.insert(7);
    int c;
                                                    1.insert(6);
    Node *tail;
                                                    1.insert(9);
    List()
                                                    l.insert(1);
                                                    1.insert(3);
        head=NULL;
                                                    1.insert(7);
                                                    cout << "Current Linked List: ";</pre>
        c=0;
        tail=NULL;
                                                    1.display();
                                                  }
    List* createList()
    {
                                                  Current Linked List: 7 3 1 9 6 7 3 1 9 6
                                                  Process returned 0 (0x0)
                                                                              execution time
        List *new1=new List;
                                                  Press any key to continue.
        return new1;
                                                  02.
    void insert(int val)
                                                  #include<iostream>
                                                  using namespace std;
      Node * new_node = new Node(val);
                                                  class Node
      if (head == NULL)
        head = new_node;
                                                  public:
                                                      int data;
      else
                                                      Node *next;
        new_node->next = head;
                                                      Node(int info)
        head = new_node;
      }
                                                          data=info;
```

```
int main() {
        next=NULL;
    }
                                                    List 1;
};
                                                     // inserting elements
                                                     1.insert(6);
class List
                                                     1.insert(9);
                                                     l.insert(1);
public:
                                                     1.insert(3);
    Node *head;
                                                     1.insert(7);
    int c;
                                                     1.insert(6);
    Node *tail;
                                                     1.insert(9);
    List()
                                                     1.insert(1);
    {
                                                     1.insert(3);
        head=NULL;
                                                     1.insert(7);
        c=0;
                                                     1.count();
        tail=NULL;
    List* createList()
    {
                                                   TOTAL number of nodes= 10
                                                   MAXIMUM VALUE= 9 and MINIMUM VALUE= 1
        List *new1=new List;
                                                  Process returned 0 (0x0) execution time : 0.387 s
        return new1;
                                                  Press any key to continue.
    }
    void insert(int val)
                                                  Q3.
      Node * new_node = new Node(val);
                                                  #include<iostream>
                                                  using namespace std;
      if (head == NULL)
        head = new_node;
                                                  class Node
                                                  {
      else
                                                  public:
                                                       int data;
        new node->next = head;
                                                       Node *next;
        head = new node;
      }
                                                       Node(int info)
    }
                                                       {
                                                           data=info;
    void count()
                                                           next=NULL;
                                                       }
      Node* temp = head;
      int mn=INT_MAX,mx=INT_MIN,c=0;
                                                  };
      while(temp != NULL)
                                                  class List
                                                  {
        mn=min( temp->data,mn);
                                                  public:
        mx=max( temp->data,mx);
                                                       Node *head;
        C++;
                                                       int c;
        temp = temp->next;
                                                       Node *tail;
      }
                                                       List()
      cout<<"TOTAL number of nodes=
                                                       {
"<<c<endl;
                                                           head=NULL;
      cout << "MAXIMUM VALUE= "<<mx<<"
                                                           c=0;
and MINIMUM VALUE= "<<mn<<endl;</pre>
                                                           tail=NULL;
    }
                                                       }
};
```

```
List* createList()
                                                   using namespace std;
    {
                                                   class Node
        List *new1=new List;
        return new1;
                                                   public:
    }
                                                       int data;
    void insertAtBeginning(int val)
                                                       Node *next;
      Node * new_node = new Node(val);
                                                       Node(int info)
      if (head == NULL)
                                                           data=info;
        head = new node;
                                                           next=NULL;
                                                       }
      else
                                                   };
        new_node->next = head;
        head = new_node;
                                                   class List
      }
    }
                                                   public:
                                                       Node *head;
    void display()
                                                       int c;
                                                       Node *tail;
      Node* temp = head;
                                                       List()
      while(temp != NULL)
                                                       {
                                                           head=NULL;
        cout << temp->data << " ";</pre>
                                                           c=0;
                                                           tail=NULL;
        temp = temp->next;
      }
                                                       }
      cout << endl;</pre>
                                                       List* createList()
                                                       {
};
int main() {
                                                           List *new1=new List;
                                                           return new1;
  List 1;
                                                       }
  // inserting elements
                                                       void insertAtLocation(int val,int
  1.insertAtBeginning(6);
                                                   pos)
  1.insertAtBeginning(9);
                                                         Node * new node = new
  1.insertAtBeginning(1);
                                                   Node(val),*temp;
  1.insertAtBeginning(3);
  1.insertAtBeginning(1);
                                                         temp=head;
                                                         if(pos==1)
  1.insertAtBeginning(3);
cout << "Current Linked List: ";</pre>
                                                         {
                                                               new_node->next = head;
  1.display();
  cout << "Inserting 7 at the Beginning</pre>
                                                               head = new_node;
of the Linked List: ";
  1.insertAtBeginning(7);
                                                         }
  1.display();
                                                         else{
}
                                                         pos--;
                                                         while(--pos)
Current Linked List: 3 1 3 1 9 6
                                                         {
Inserting 7 at the Beginning of the Linked List: 7 3 1 3 1
                                                              temp=temp->next;
Process returned 0 (0x0) execution time : 4.382 s
Press any key to continue.
                                                         new_node->next=temp->next;
Q4.
                                                         temp->next=new_node;
#include<iostream>
                                                         }
```

```
}
                                                   {
    void insert(int val)
                                                   public:
                                                       int data;
      Node * new_node = new Node(val);
                                                       Node *next;
      if (head == NULL)
                                                       Node(int info)
        head = new_node;
                                                            data=info;
      else
                                                            next=NULL;
                                                       }
        new_node->next = head;
        head = new_node;
                                                   };
      }
    }
                                                   class List
    void display()
                                                   public:
                                                       Node *head;
      Node* temp = head;
                                                       int c;
      while(temp != NULL)
                                                       Node *tail;
                                                       List()
        cout << temp->data << " ";</pre>
                                                       {
        temp = temp->next;
                                                            head=NULL;
                                                            c=0;
      cout << endl;</pre>
                                                            tail=NULL;
                                                       List* createList()
};
int main() {
                                                       {
  List 1;
                                                            List *new1=new List;
  // inserting elements
                                                            return new1;
  1.insert(6);
                                                       }
  1.insert(9);
                                                       void insert(int val)
  l.insert(1);
  l.insert(3);
                                                         Node * new_node = new Node(val);
  1.insert(7);
  cout << "Current Linked List: ";</pre>
                                                         if (head == NULL)
  1.display();
                                                            head = new_node;
  cout << "Inserting 8 at position 2 of
Linked List: ";
                                                         else
  1.insertAtLocation(8,2);
                                                            new_node->next = head;
                                                            head = new_node;
  1.display();
                                                         }
}
                                                       }
Current Linked List: 7 3 1 9 6
                                                       void display()
Inserting 8 at position 2 of Linked List: 7 8 3 1 9 6
                                                         Node* temp = head;
Process returned 0 (0x0)
                       execution time : 0.324 s
ress any key to continue.
                                                         while(temp != NULL)
                                                         {
                                                            cout << temp->data << " ";</pre>
Q5.
                                                            temp = temp->next;
#include<iostream>
                                                         }
using namespace std;
                                                          cout << endl;</pre>
class Node
                                                       }
```

```
};
int main() {
                                                              List *new1=new List;
                                                              return new1;
  List 1;
  int n;
                                                         void insert(char val)
  // inserting elements
                                                           Node * new_node = new Node(val);
  cout<<"Enter any number : ";</pre>
  cin>>n;
  while(n!=0)
                                                           if (head == NULL)
                                                              {head = new_node;
      l.insert(n%10);
                                                               tail=new node;
      n=n/10;
  }
  cout << "Current Linked List: ";</pre>
                                                           else
  1.display();
}
                                                              tail->next = new_node;
                                                              tail = new_node;
Enter any number : 13651434
                                                           }
Current Linked List: 1 3 6 5 1 4 3 4
                                                         }
Process returned 0 (0x0)
                         execution time : 4.026 s
Press any key to continue.
                                                         void display()
                                                           Node* temp = head;
Q6.
                                                           while(temp != NULL)
#include<iostream>
using namespace std;
                                                              cout << temp->data << " ";</pre>
                                                              temp = temp->next;
class Node
                                                           }
                                                           cout << endl;</pre>
public:
                                                         }
    char data;
                                                     };
    Node *next;
                                                     int main() {
    Node(char info)
                                                       List 1;
                                                       string n;
                                                       // inserting elements
         data=info;
                                                       cout<<"Enter any Name : ";</pre>
         next=NULL;
                                                       cin>>n;
    }
                                                       for(int i=0;i<n.size();i++)</pre>
                                                       {
};
                                                           1.insert(n[i]);
class List
                                                       cout << "Current Linked List: ";</pre>
public:
                                                       1.display();
    Node *head;
                                                     }
    int c;
    Node *tail;
                                                     Enter any Name : AVNI
    List()
                                                     Current Linked List: A V N I
    {
                                                     Process returned 0 (0x0) execution time : 4.807 s
         head=NULL;
                                                     Press any key to continue.
         c=0;
         tail=NULL;
    }
                                                    Q7.
    List* createList()
    {
```

```
Node* temp = head;
#include<iostream>
                                                        while(temp != NULL)
using namespace std;
                                                          Node* temp1 = 12.head;
class Node
                                                          for(int i=0;i<3;i++)
public:
    char data;
                                                              if(temp->data==temp1->data
    Node *next;
                                                 && temp->next->data==temp1->next->data
                                                 && temp->next->next->data==temp1->next-
    Node(char info)
                                                 >next->data)
                                                              {
        data=info;
                                                                           Node* curr =
        next=NULL;
                                                 head;
    }
                                                                           Node* prev =
                                                 head;
                                                                           while(curr
};
                                                 !=NULL && curr->data!=temp->data)
class List
{
                                                                               prev = curr;
public:
                                                                               curr = curr-
    Node *head;
                                                 >next;
    int c;
                                                                               }
    Node *tail;
    List()
                                                                           prev->next=temp-
    {
                                                 >next->next->next;
        head=NULL;
        c=0;
                                                              }
        tail=NULL;
    }
                                                              temp1=temp1->next;
    List* createList()
                                                          }
                                                          temp = temp->next;
                                                        }
        List *new1=new List;
        return new1;
                                                       display();
    void insert(char val)
                                                       void display()
                                                        Node* temp = head;
      Node * new_node = new Node(val);
                                                        while(temp != NULL)
      if (head == NULL)
                                                        {
        {head = new_node;
                                                          cout << temp->data << " ";</pre>
                                                          temp = temp->next;
         tail=new_node;
                                                        }
        }
                                                        cout << endl;</pre>
      else
                                                     }
                                                 };
        tail->next = new_node;
                                                 int main() {
        tail = new_node;
      }
                                                   List 11,12;
    }
                                                   string s1="abcdefghij",s2="defab";
    void remove(List 12)
                                                   for(int i=0;i<s1.size();i++)</pre>
    {
                                                   {
```

```
11.insert(s1[i]);
                                                         newNode->next = NULL;
                                                         newNode->prev = NULL;
                                                         if(head == NULL) {
  }
  for(int i=0;i<s2.size();i++)</pre>
                                                           head = newNode;
                                                         } else {
      12.insert(s2[i]);
                                                           Node* temp = head;
                                                           while(temp->next != NULL)
                                                             temp = temp->next;
  }
  cout << "Current Linked List - 1 : ";</pre>
                                                           temp->next = newNode;
  l1.display();
                                                           newNode->prev = temp;
  cout << "Current Linked List - 2 : ";</pre>
                                                         }
                                                       }
  12.display();
  cout << "After calling remove function</pre>
Linked List 1 becomes : ";
                                                       void push_at(int newElement, int
  11.remove(12);
                                                  position) {
                                                         Node* newNode = new Node();
}
                                                         newNode->data = newElement;
                                                         newNode->next = NULL;
Current Linked List - 1 : a b c d e f g h i j
                                                         newNode->prev = NULL;
Current Linked List - 2 : d e f a b
                                                         if(position < 1) {</pre>
After calling remove function Linked List 1 becomes : a b c g h i
                                                           cout<<"\nposition should be >=
Process returned 0 (0x0) execution time : 0.992 s
                                                  1.";
 ress any key to continue.
                                                         } else if (position == 1) {
                                                           newNode->next = head;
08.
                                                           head->prev = newNode;
The list contains: 10 20 30
                                                           head = newNode;
Inserting an 100 at position 2
                                                         } else {
The list contains: 10 100 20 30
                                                           Node* temp = head;
                                                           for(int i = 1; i < position-1;</pre>
Inserting an 200 at position 1
                                                  i++) {
The list contains: 200 10 100 20 30
                                                             if(temp != NULL) {
                                                               temp = temp->next;
Process returned 0 (0x0)
                                 execution
                                                             }
Press any key to continue.
                                                           }
                                                           if(temp != NULL) {
                                                             newNode->next = temp->next;
#include <iostream>
                                                             newNode->prev = temp;
using namespace std;
                                                             temp->next = newNode;
                                                             if(newNode->next != NULL)
struct Node {
                                                                newNode->next->prev =
    int data;
                                                  newNode;
    Node* next;
                                                           } else {
    Node* prev;
                                                             cout<<"\nThe previous node is
};
                                                  null.";
                                                           }
class LinkedList {
                                                         }
  private:
                                                       }
    Node* head;
  public:
                                                       void PrintList() {
    LinkedList(){
                                                         Node* temp = head;
      head = NULL;
                                                         if(temp != NULL) {
    }
                                                           cout<<"The list contains: ";</pre>
                                                           while(temp != NULL) {
    void push back(int newElement) {
                                                             cout<<temp->data<<" ";</pre>
      Node* newNode = new Node();
                                                             temp = temp->next;
      newNode->data = newElement;
```

```
}
        cout<<endl;
                                                         second_last->next = NULL;
      } else {
        cout<<"The list is empty.\n";</pre>
                                                         return head;
      }
                                                  }
    }
};
                                                  void push(struct Node** head_ref, int
                                                  new_data)
int main() {
                                                  {
                                                         struct Node* new_node = new Node;
  LinkedList MyList;
                                                         new node->data = new data;
                                                         new node->next = (*head ref);
  MyList.push back(10);
  MyList.push_back(20);
                                                         (*head_ref) = new_node;
  MyList.push_back(30);
                                                  }
  MyList.PrintList();
                                                  int main()
  cout<<"Inserting an 100 at position</pre>
                                                  {
2"<<endl;
                                                         Node* head = NULL;
  MyList.push_at(100, 2);
                                                         push(&head, 12);
  MyList.PrintList();
                                                         push(&head, 29);
                                                         push(&head, 11);
  cout<<"Inserting an 200 at position
                                                         push(&head, 23);
1"<<endl;
                                                         push(&head, 8);
  MyList.push_at(200, 1);
                                                         cout<<"Currently List is: ";</pre>
  MyList.PrintList();
                                                       for (Node* temp = head; temp !=
                                                  NULL; temp = temp->next)
  return 0;
                                                               cout << temp->data << " ";</pre>
}
                                                         head = removeLastNode(head);
Q9.
                                                         cout<<endl<<"After removing last</pre>
#include <iostream>
                                                  element The list becomes: ";
                                                         for (Node* temp = head; temp !=
using namespace std;
                                                  NULL; temp = temp->next)
struct Node {
                                                               cout << temp->data << " ";</pre>
      int data;
      struct Node* next;
                                                         return 0;
};
                                                   Currently List is: 8 23 11 29 12
                                                   After removing last element The list becomes: 8 23 11 29
                                                   Process returned 0 (0x0) execution time : 0.955 s
                                                   Press any key to continue.
Node* removeLastNode(struct Node* head)
{
                                                  010.
      if (head == NULL)
                                                  #include<iostream>
             return NULL;
                                                  using namespace std;
                                                  string
      if (head->next == NULL) {
                                                  name[4]={"first","second","third","fourt
             delete head;
                                                  h"};
             return NULL;
                                                  class Node
      }
                                                  {
                                                  public:
      Node* second last = head;
                                                       int data;
      while (second_last->next->next !=
                                                       Node *next;
NULL)
                                                       Node *prev;
             second_last = second_last-
>next;
                                                       Node(int info)
                                                       {
      delete (second_last->next);
```

```
data=info;
        next=NULL;
        prev=NULL;
    }
};
class List
public:
    Node *head;
    int c=0;
    Node *tail;
    List()
    {
        head=NULL;
        c=0;
        tail=NULL;
    List* createList()
    {
        List *new1=new List;
        return new1;
    }
    void insert(int val)
        C++;
             Node* new node = new
Node(val);
             new_node->next = head;
             if (head != NULL)
                    head->prev =
new_node;
                    head = new_node;
                }
                else{
                 head=new node;
                 tail=new_node;
                }
    }
    void display()
      Node* temp = head;
      while(temp != NULL)
        cout << temp->data << " ";</pre>
        temp = temp->next;
      }
      cout << endl;</pre>
    }
```

```
void extremeSwap()
         Node* t1 = head;
         Node* t2 = tail;
         for(int i=0;i<c/2;i++)
              cout<<"Calling EXTREME Swap
function for "<<name[i]<<" time:"<<endl;</pre>
              swap(t1->data,t2->data);
              t1=t1->next;
              t2=t2->prev;
              display();
              cout<<endl;</pre>
         }
    }
};
int main() {
  List 1;
  l.insert(1);
  1.insert(2);
  1.insert(3);
  1.insert(4);
  1.insert(5);
  1.insert(6);
  1.insert(7);
  1.insert(8);
  cout << "Current Linked List: ";</pre>
  1.display();
  1.extremeSwap();
Current Linked List: 8 7 6 5 4 3 2 1
Calling EXTREME Swap function for first time:
17654328
Calling EXTREME Swap function for second time:
1 2 6 5 4 3 7 8
Calling EXTREME Swap function for third time:
1 2 3 5 4 6 7 8
Calling EXTREME Swap function for fourth time:
1 2 3 4 5 6 7 8
Process returned 0 (0x0)
                        execution time : 2.956 s
Press any key to continue.
```