DSA-Lab-4

Roll No: 24P-0706 **Dept: BS-CS**

Name: Aazan Noor Khuwaja **Section : 3D**

Qns:1

```
#include<iostream>
using namespace std;
class student_record{
   string stu_id,stu_name;
   float stu_gpa;
   public:
   student_record():stu_id(""),stu_name(""),stu_gpa(0){

   }

   void set_stu_id(){
    string id;
    cout<<"Enter student ID:"<<endl;
    getline(cin,id);
    stu_id=id;
   }

   void set_stu_name(){
    string name;
    cout<<"Enter student name:"<<endl;
    getline(cin,name);
    stu_name=name;
   }
```

```
    void set_stu_gpa(){
     float gpa;
     cout<<"Enter student GPA:"<<endl;
     cin>>gpa;
     cin.ignore();
     stu_gpa=gpa;
    }

    string get_id()
    {
       return stu_id;
    }

    string get_name()
    {
       return stu_name;
    }

    float get_gpa(){
       return stu_gpa;
    }
};

class node {
    public:
    student_record data;
    node *next;
    node():next(NULL){}
};
```

```
class stu_list{
public:
node *head,*tail;
stu_list():head(NULL),tail(NULL){}

bool is_list_empty()
{
   return (head==NULL);
}

void add_student(){
   node *n_node=new node;
if(is_list_empty()){

   n_node->data.set_stu_id();
   n_node->data.set_stu_name();
   n_node->data.set_stu_gpa();
   head=n_node;
   tail=n_node;
   tail->next=head;
   return;
}
   n_node->data.set_stu_id();
   n_node->data.set_stu_name();
   n_node->data.set_stu_gpa();
   tail->next=n_node;
   tail = n_node;
   tail->next=head;
}
```

```
void dlt_student(){
    if(is_list_empty())
   {
      cout <<"List is empty there is no student!"<<endl;
      return ;
   }
   cout<<"Enter the id of the student you want to delete: "<<endl;
   string id_d;
   getline(cin,id_d);
   if(head->data.get_id()==id_d && head->next==head)
   {
      delete head;
      head=NULL;
      cout<<id_d<<" Student Removed!"<<endl;
      return;
   }
   node *temp=head;
   while(temp->next!=head )
   {
      temp=temp->next;
   }
   tail=temp;
   node *temp2=head;
   head=head->next;
   temp->next=head;
   delete temp2;
   node *previous=NULL,*current=head;
   do {
      previous=current;
```

```
      current=current->next;
      if(current->data.get_id() == id_d)
      {
         previous->next=current->next;
         if(current==head){
         tail=previous;
         }
         delete current;
         return;
      }
   }
   while(current!=head);
}

void search_students(){
   if(is_list_empty())
   {
      cout <<"List is empty there is no student!"<<endl;
      return ;
   }
   string search_id;
   cout<<"Enter student id to searh: "<<endl;
   getline(cin,search_id);
   node *tp=head;
   do {
      if(tp->data.get_id()==search_id){
         cout<<" student milgaya: \n ID:"<<tp->data.get_id()<<",
Name:"<<tp->data.get_name()<<", GPA: "<<tp->data.get_gpa()<<endl;
         return;
      }
```

```
      tp=tp->next;
   }
   while(tp!=head);
}

void display_students(){
    if(is_list_empty())
   {
      cout <<"List is empty there is no student!"<<endl;
      return ;
   }
 node *tp=head;
   do {
         cout<<"we found student : \n ID:"<<tp->data.get_id()<<",
Name:"<<tp->data.get_name()<<", GPA: "<<tp->data.get_gpa()<<endl;
         tp=tp->next;
      }
   while(tp != head);

}

void cal_avg_gpa(){
   if(is_list_empty())
   {
      cout <<"List is empty there is no student!"<<endl;
      return ;
   }
node *tp=head;
int count=0;
float avg_g=0;
```

```
   do {
         avg_g=avg_g+tp->data.get_gpa();
         count++;
         tp=tp->next;
   }
   while(tp != head);
   avg_g=avg_g/count;
   cout<<"Average GPA :"<<avg_g<<endl;
}

};
int main()
{
   stu_list s;
   int chose;
   cout<<"Welcome to the Student Registration System!"<<endl;
   while(true)
   {
      cout<<"\n1. Add a student\n2. Remove a student\n3. Search for a
student\n4. Display all students\n5. Calculate average GPA\n6.
Exit"<<endl;
      cin>>chose;
      cin.ignore();
      switch(chose)
      {
         case 1:
            s.add_student();
            break;

         case 2:
```

```
            s.dlt_student();
            break;

        case 3:
            s.search_students();
            break;

        case 4:
            s.display_students();
            break;

        case 5:
            s.cal_avg_gpa();
            break;

        case 6:
            exit(0);
            break;

        default:
            cout<<"Your input is not valid !"<<endl;
            break;
      }
   }

}
```

Qns:2

```
#include<iostream>
```

```
using namespace std;
class node{
   public:
   int data;
   node *next;
   node(int val):data(val),next(NULL){}
};
class sslist{
   public:
   // int node_c_1=0,node_c_2=0;
   node *head1,*head2;
   sslist():head1(NULL),head2(NULL){};
   bool is_first_head_empty()
   {
      return(head1==NULL);
   }
   bool is_second_head_empty()
   {
      return(head2==NULL);
   }
   void insert_in_first(){
      int n;
      cout<<"Enter values for first list:"<<endl;
      cin>>n;
      cin.ignore();
      node *n_node=new node(n);
      if(is_first_head_empty())
      {
         head1=n_node;
         head1->next=head1;
```

```
      return;
    }
    node *tp1=head1;
    while(tp1->next!=head1){

      tp1=tp1->next;
    }
    tp1->next=n_node;
    n_node->next=head1;
    //  node_c_1++;
}

    void display_l1()
{
    node *temp = head1;
    do {
    cout << temp->data << " , ";
    temp = temp->next;

    }
    while (temp != head1);
}

  void insert_in_second(){
    int n;
    cout<<"\nEnter values for second list:"<<endl;
    cin>>n;
    cin.ignore();
    node *n_node=new node(n);
    if(is_second_head_empty())
```

```
   {
      head2=n_node;
      head2->next=head2;
      return;
   }
   node *tp2=head2;
   while(tp2->next!=head2){

      tp2=tp2->next;
   }
   tp2->next=n_node;
   n_node->next=head2;
   //   node_c_2++;
}
    void display_l2()
{
   node* temp = head2;
   do {
   cout << temp->data << " -> ";
   temp = temp->next;

   }
   while (temp != head2);
}
void c_values()
{
   int count_c=0;
   node *temp1=head1;
   // if(node_c_1>node_c_2){
   //    for(int i=0;i<node_c_1;i++)
```

```
//    {
//       int st1=temp1->data;
//      for(int j=0;j<node_c_2;j++){
//       if(st1==temp2->data)
//       {
//          count_c++;

//       }
//        temp2=temp2->next;
//    }
//       temp1=temp1->next;

//    }
// }
// else{
//    for(int i=0;i<node_c_2;i++)
//    {
//       int st1=temp2->data;
//      for(int j=0;j<node_c_1;j++){
//       if(st1==temp1->data)
//       {
//          count_c++;

//       }
//        temp1=temp1->next;
//    }
//       temp2=temp2->next;

//    }
// }
```

```
        do{
        int st1=temp1->data;

        node *temp2=head2;
        do{
           if(st1==temp2->data) {
              count_c++;
              break;
           }
           temp2=temp2->next;
        }while(temp2!=head2);

        temp1=temp1->next;
     } while(temp1!=head1);
        cout<<"\nTotal common values are ="<<count_c<<endl;

     }


};
int main()
{
    sslist l;
    l.insert_in_first();
    l.insert_in_first();
    l.insert_in_first();
    l.insert_in_first();
    l.insert_in_first();
    l.display_l1();
    l.insert_in_second();
```

```
   l.insert_in_second();
   l.insert_in_second();
   l.insert_in_second();
   l.insert_in_second();
   l.display_l2();
   l.c_values();


}

Qns3:
#include<iostream>
using namespace std;
class node{
   public:
   int data;
   node *next;
   node(int val):data(val),next(NULL){}
};
class sslist{
   public:
   node *head1;
   sslist():head1(NULL){};
   bool is_empty()
   {
      return(head1==NULL);
   }
   void insert_in_first(){
      int n;
      cout<<"Enter values for list:"<<endl;
```

```
   cin>>n;
   cin.ignore();
   node *n_node=new node(n);
   if(is_empty())
   {
      head1=n_node;
      head1->next=head1;
      return;
   }
   node *tp1=head1;
   while(tp1->next!=head1){

      tp1=tp1->next;
   }
   tp1->next=n_node;
   n_node->next=head1;
}
void del_dup()
{
  if (is_empty() || head1->next == head1)
{
   return;
}

node *curr=head1;
do {
   node *prev=curr;
   node *check=curr->next;

   while (check!=head1) {
```

```
        if (check->data==curr->data) {
            prev->next=check->next;
            delete check;
            check=prev->next;
        } else {
            prev=check;
            check=check->next;
        }
      }

      curr=curr->next;
   } while(curr->next!=head1);
   }
    void display()
   {
      node *temp = head1;
      do {
      cout << temp->data << " , ";
      temp = temp->next;

      }
      while (temp != head1);
   }

};
int main() {
   sslist l;
   int n;
   cout<<"how many values you want to put in list:";
   cin>>n;
```

```
    for (int i=0;i<n;i++) {
        l.insert_in_first();
    }
    cout << "\nList (before):";
    l.display();
    l.del_dup();
    cout << "\nList (after):";
    l.display();
    return 0;
}
```