BM2043 - Algorithms and Data Structures

REPORT – PROJECT WORK

Dr Nagarajan Ganapathy

Version 1
Datum: 28.09.2022

```
#include <iostream>
using namespace std;
#include<string.h>
#include <fstream>
#include <sstream>
```

<u>Defining a structure with different data types for a node in linked list.</u>

<u>T~O(1)</u>

```
struct Node
{
 string  Name;
 string Department;
 string Type;
 string Period;
 string Feedback;
 string Phone;
 string Email;
 string Number;
 struct Node *next;
}*first=NULL;
```

<u>Sorting the list in alphabetical order of names:-</u>

<u>Directly we inserted a particular name in its sorted position while inserting in a in the linked list</u>

<u>T~O(n)</u>

```
void SortedInsert(struct Node *p,string Name,string Department,string Type,string Pe-
riod,string Feedback,string Phone,string Email,string Number)
{

 struct Node *t,*q=NULL;

 t=(struct Node*)malloc(sizeof(struct Node));
 t->Name=Name;
 t->Department=Department;
 t-> Type=Type;
 t-> Period=Period;
 t-> Feedback=Feedback;
 t-> Phone=Phone;
 t-> Email=Email;
 t-> Number=Number;
 t->next=NULL;

 if(first==NULL)
 first=t;
 else
 {
 while(p && p->Name.compare(Name)<0)
 {
 q=p;
 p=p->next;
 }
 if(p==first)
 {
 t->next=first;
 first=t;
 }
 else
 {
 t->next=q->next;
 q->next=t;
 }
 }

}
```

<u>Searching the doctor by name and then returning a pointer to that node</u>

<u>T~O(n)</u>

```
struct Node * NameSearch(struct Node *p,string Name)
{
if(p==NULL)
return NULL;
if(Name==p->Name)
return p;
return NameSearch(p->next,Name);
}
```

<u>Searching the doctor by name and printing its data</u>

<u>T~O(n)</u>

```
void SearchName(struct Node *p,string Name)
{
    struct Node *t;
    while(p)
    {
      if(Name==p->Name)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }
    return;
}
```

Searching the doctor by department and printing its data

T~O(n)

```
void SearchDepartment(struct Node *p,string Department)
{
    struct Node *t;
    while(p)
    {
      if(Department==p->Department)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t-> Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

Searching the doctor by Type of employment and printing its data

T~O(n)

```
void SearchType(struct Node *p,string Type)
{
    struct Node *t;
    while(p)
    {
      if(Type==p->Type)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

Searching the doctor by Period of Availability and printing its data

T~O(n)

```
void SearchPeriod(struct Node *p,string Period)
{
    struct Node *t;
    while(p)
    {
      if(Period==p->Period)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

Searching the doctor by feedback and printing its data

T~O(n)

```
void SearchFeedback(struct Node *p,string Feedback)
{
    struct Node *t;
    while(p)
    {
      if(Feedback==p->Feedback)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

<u>Searching the doctor by PhoneNo. and printing its data</u>

<u>T~O(n)</u>

```
void SearchPhone(struct Node *p,string Phone)
{
    struct Node *t;
    while(p)
    {
      if(Phone==p->Phone)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

Searching the doctor by Email. and printing its data

T~O(n)

```
void SearchEmail(struct Node *p,string Email)
{
    struct Node *t;
    while(p)
    {
      if(Email==p->Email)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

Searching the doctor by Number of patients he treated and printing its data

T~O(n)

```
void SearchNumber(struct Node *p,string Number)
{
    struct Node *t;
    while(p)
    {
      if(Number==p->Number)
      {
          t=p;
          cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t->
Feedback<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
      }
      p=p->next;
    }

    return;
}
```

<u>Search function just combines all the above Doctor search functions for a better user experience</u>

<u>T~O(n)</u>

```
void Search()
{
    int n;
    string Name;
    string Department;
    string Type;
    string Period;
    string Feedback;
    string Phone;
    string Email;
    string Number;
    cout<< "1. Search By Name of the doctor"<<endl;
    cout<< "2. Search By Department of the doctor"<<endl;
    cout<< "3. Search By Type of employment"<<endl;
    cout<< "4. Search By Period of availability"<<endl;
    cout<< "5. Search By Feedback Rating"<<endl;
    cout<< "6. Search By Phone No"<<endl;
    cout<< "7. Search By Email Id"<<endl;
    cout<< "8. Search By Number of patients"<<endl;
    cin>>n;
switch (n) {
  case 1:
    getline(cin>>ws,  Name);
    SearchName(first,Name);
    break;
  case 2:
    getline(cin>>ws, Department);
    SearchDepartment(first,Department);
    break;
  case 3:
    getline(cin>>ws, Type);
    SearchType(first,Type);
    break;
  case 4:
    getline(cin>>ws, Period);
    SearchPeriod(first,Period);
    break;
  case 5:
    getline(cin>>ws,Feedback);
    SearchFeedback(first,Feedback);
    break;
  case 6:
    getline(cin>>ws, Phone);
    SearchPhone(first,Phone);
    break;
  case 7:
    getline(cin>>ws, Email);
    SearchEmail(first,Email);
    break;
  case 8:
    getline(cin>>ws, Number);
    SearchNumber(first,Number);
    break;
}
}
```

Count function counts the number of nodes in linked list

T~O(n)

```
int count(struct Node* p)
```

```
{
    int len=0;
    while(p)
    {
      len++;
      p=p->next;
    }
    return len;
}
```

<u>Function to delete a doctor from linked list given its name</u>

<u>T~O(n)</u>

```
string Delete(struct Node *p,string Name)
{
struct Node* r,*s=NULL;
r=p;
s=NameSearch(first,Name);
int index=1;
while(r!=s&&r)
{


    index++;
    r=r->next;
}
struct Node *q=NULL;
string x="-1";int i;
if(index < 1 || index > count(p))
return x;
if(index==1)
{
q=first;
x=first->Name;
first=first->next;
free(q);
return x;
}
else
{
for(i=0;i<index-1;i++)
{
q=p;
p=p->next;
}
q->next=p->next;
x=p->Name;
free(p);
return x;
}
}
```

Displaying all the doctors present in the linked list

T~O(n)

```
void Display(struct Node *t)
{
 while(t!=NULL)
 {
 cout<<t->Name<<" "<<t->Department<<" "<<t-> Type<<" "<<t-> Period<<" "<<t-> Feed-
back<<" "<<t-> Phone<<" "<<t-> Email<<" "<<t-> Number<<endl;
 t=t->next;
 }
}
```

Function to update the details of doctor given its name

T~O(n)

```
void Update(string Name)
{
    string Department;
    string Type;
    string Period;
    string Feedback;
    string Phone;
    string Email;
    string Number;
    struct Node *t;
    t=NameSearch(first,Name);
    cout<<"Department of the doctor:";
    getline(cin>>ws,Department);
    cout<<"Type of employment:";
    getline(cin>>ws,Type);
    cout<<"Period of availability:";
    getline(cin>>ws,Period);
    cout<<"Feedback Rating:";
    getline(cin>>ws,Feedback);
    cout<<"Phone No:";
    getline(cin>>ws,Phone);
    cout<<"Email Id:";
    getline(cin>>ws,Email);
    cout<<"Number of patients:";
    getline(cin>>ws,Number);
    t->Department=Department;
    t-> Type=Type;
    t-> Period=Period;
    t-> Feedback=Feedback;
    t-> Phone=Phone;
    t-> Email=Email;
    t-> Number=Number;
}


int main()
```

{

<u>Commented section was used only for test datas</u>

```
  /* int n;
 cin>>n;


 string Name;
 string Department;
 string Type;
 string Period;
 string Feedback;
 string Phone;
 string Email;
 string Number;
 for(int i=0;i<n;i++)
 {
    cout<<"Name of the doctor:";
    getline(cin>>ws, Name);
    cout<<"Department of the doctor:";
    getline(cin>>ws,Department);
    cout<<"Type of employment:";
    getline(cin>>ws,Type);
    cout<<"Period of availability:";
    getline(cin>>ws,Period);
    cout<<"Feedback Rating:";
    getline(cin>>ws,Feedback);
    cout<<"Phone No:";
    getline(cin>>ws,Phone);
    cout<<"Email Id:";
    getline(cin>>ws,Email);
    cout<<"Number of patients:";
    getline(cin>>ws,Number);
    if(NameSearch(first,Name)==NULL)
    SortedInsert(first,Name,Department,Type,Period,Feedback,Phone,Email,Number);
 }
```

```
cout<<"Delete:";
getline(cin>>ws, Name);
Display(first);
Delete(first,Name);
Display(first);
cout<<"Update:";
getline(cin>>ws, Name);
Update(Name);
Display(first);
Search();
*/
```

Reading data from csv file and storing all the information in a linked list

```
  ifstream inputFile;
  inputFile.open("BM2043_PROJECT_DATA.csv");

  string line="";
  getline(inputFile, line);
  line="";
  cout<<"DATA"<<endl;
  while(getline(inputFile, line))
  {
    string Name;
    string Department;
    string Type;
    string Period;
    string Feedback;
    string Phone;
    string Email;
    string Number;

    stringstream inputString(line);
    getline(inputString, Name, ',');
    getline(inputString, Department, ',');
```

```
 getline(inputString, Type, ',');
 getline(inputString, Period, ',');
 getline(inputString, Feedback, ',');
 getline(inputString, Phone, ',');
 getline(inputString, Email, ',');
 getline(inputString, Number, ',');

 line="";
```

<u>While inserting in a linked list 2 things are taken care of:</u>

1) ***<u>If doctor is already present ,don't insert that doctor in the linked list</u>***
2) ***<u>While inserting the doctor it should be inserted in its right position that is in its sorted position</u>***

```
 if(NameSearch(first,Name)==NULL)
 SortedInsert(first,Name,Department,Type,Period,Feedback,Phone,Email,Number);

}
```

<u>Showing the features of the project.(Update,Delete,Search and more)</u>

```
string Name;
cout<<"Delete:";
getline(cin>>ws, Name);
Display(first);
Delete(first,Name);
Display(first);
cout<<"Update:";
getline(cin>>ws, Name);
Update(Name);
Display(first);
Search();
string Department;
cout<<"Enter Department of doctor you need"<<endl;
getline(cin>>ws, Department);
SearchDepartment(first,Department);
cout<<"Enter Name of doctor you want"<<endl;
getline(cin>>ws, Name);
```

```
    cout<<"Appointment Done"<<endl<<"Appointed Doctor is"<<endl;
    SearchName(first,Name);
    cout<<"Enter feedback rating"<<endl;
    string Feedback;
    getline(cin>>ws, Feedback);
    struct Node *t;
    t=NameSearch(first,Name);
    int feedback=stoi(t->Feedback);
    int number=stoi(t->Number);
    int newfeedback=stoi(Feedback);
```

Calculation of feedback
Lets say initially the doctor had 5 feedback rating and number of patients were 17
Feedback rating given by new patient=4
New Feedback rating of doctor={(5*17)+4}/18

```
    float calc=(feedback*number+newfeedback)/(float)(number+1);
    t-> Feedback=to_string(calc);
    t-> Number=to_string(number + 1);
    cout<<"Data of Doctor after updating feedback and no.of patients"<<endl;
    SearchName(first,Name);


  return 0;
}
```

Exactly same things are done for the patient's directory so its report having all the time complexities will be exactly similar to this only.

We included the appointment and feedback management system directly with the doctor's directory itself.