CITY DATABASE IMPLEMETATION

A PROJECT REPORT

Submitted by

BL.EN.U4EAC19012 DIPTESH TARAFDAR

As a part of

19EAC203 - DATA STRUCTURES AND ALGORITHMS

Offered by

Mr. Nandivardhan H.R.
Department of Electronics and Communications Engineering

OCTOBER-2020

AMRITA SCHOOL OF ENGINEERING, BENGALURU CAMPUS

AMRITA VISHWA VIDYAPEETHAM

BENGALURU 560 035

CONTENTS

1. PROBLEM DEFINITION

Implement a city database. Each database record contains the name of the city (a string of arbitrary length) and the coordinates of the city expressed as integer x and y coordinates. Your database should allow records to be inserted, deleted by name or coordinate, and searched by name or coordinate. Another operation that should be supported is to print all records within a given distance of a specified point (capital city).

2. Specifications and Flow (Use Case) explanation

Operations required to be implemented in the program:

a) Insertion of records into the database

b) Deletion of records from the database by
   - Name of the city
   - Coordinates of the city

c) Searching for a record in the database by
   - Name of the city
   - Coordinates of the city

d) print all records within a given distance from capital city

e) Printing all records present in the database

3. DESIGN

My choice of Data Structure is the Doubly Linked List, which is a linked data structure that consists of a set of sequentially linked records called nodes. Each node contains two fields, called links, that are references to the previous and to the next node in the sequence of nodes.

Reason:

- Traversal in both forward and backward direction because contains an extra pointer to link the previous node
- The delete operation in DLL is more efficient if pointer to the node to be deleted is given.

Functions used in the program:

a) float toRadians(const float degree)
   - function to convert city coordinates, i.e., latitude and longitude from degrees to radians
   - Receives coordinate(longitude or latitude) as argument
   - Returns radian value of floating type

b) float distance(float lat2,float long2)
   - function to calculate distance b/w two points using their coordinates
   - Receives coordinates(latitude and longitude) of the city as float arguments
   - Returns distance between capital city(Bengaluru) and city in the record

c) void append(struct node** head_ref)
   - Function to insert a new city record(node) into the database(LL)
   - Does not return any values
   - Receives address of pointer to the head node as argument
   - Algorithm to insert new node:

Step 1 - Create a newNode with given value and newNode → next as NULL.

Step 2 - Check whether list is Empty (head == NULL)

Step 3 - If it is Empty, then assign NULL to newNode → previous and newNode to head.

Step 4 - If it is not Empty, then, define a node pointer temp and initialize with head.

Step 5 - Keep moving the temp to its next node until it reaches to the last node in the list (until temp → next is equal to NULL).

Step 6 - Assign newNode to temp → next and temp to newNode → previous.

d) void display(struct node* head_ref)
- function to display all records from database(LL)
- Receives pointer to the head node as argument
- Does not return any values

e) void searchName(struct node* head_ref,char k_name[])
- Function to search record by name of the city
- Receives pointer to the head node and name of city to be searched as arguments
- Does not return any values

f) void searchCord(struct node* head_ref,float X, float Y)
- Function to search record by coordinates of the city
- Receives coordinates(latitude and longitude) of the city to be searched for as float arguments
- Does not return any values

g) void delNode(struct node** h_ref, struct node* del)
- function to delete node
- Receives address of pointer to the head node and the pointer to the node to be deleted as arguments
- Does not return any values

h) void delName(struct node* head_ref, char k_name[])
   - function to delete a record(node) w.r.t. city name
   - Receives pointer to the head node and name of city to be deleted as arguments
   - Does not return any values

i) void delCord(struct node* head_ref, float X, float Y)
   - function to delete a record(node) w.r.t. city coordinates
   - Receives pointer to the head node and coordinates(latitude and longitude) of the city to be deleted as arguments
   - Does not return any values

j) void print(struct node* head_ref)
   - Function to print city records within given distance from capital city
   - Receives pointer to the head node as argument
   - Does not return any values

4. IMPLEMENTATION

```c
/* Implementation of city database using Doubly Linked List*/

#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>

struct node  // doubly LL node defintion
{
    char name[50];  // name of city
    float x;        // x-latitude
    float y;        // y-longitude
    float dist;     // distance between city in node and capital city
    struct node* prev;  //pointer to previous node
    struct node* next;  //pointer to next node
};

float toRadians(const float degree) //function to convert city coordinates to radians
{
    float one_deg=(M_PI)/180;
    return (one_deg*degree);
}

float distance(float lat2,float long2) //function to calculate distance b/w two points usi
ng their coordinates
{
    float lat1=12.97, long1=77.59;

    //Convert the latitudes and longitudes from degree to radians.
    lat1=toRadians(lat1);
    long1=toRadians(long1);
    lat2=toRadians(lat2);
    long2=toRadians(long2);

    float dlat=lat2-lat1;
    float dlong=long2-long1;

    float ans= pow(sin(dlat / 2), 2) +  cos(lat1) * cos(lat2) * pow(sin(dlong / 2), 2);
    ans = 2 * asin(sqrt(ans));
    float r=6371;
    ans= ans*r;

    return ans;
}

void print(struct node* head_ref) //Function to print city records within given distance f
rom capital city
{
    struct node* curr=NULL;
    curr=head_ref;
    int flag=1,search=1;
    float d;
    printf("\n  Enter distance(in Kms): ");
```

```
    scanf("%f",&d);
    printf("\n  City records within %.2f kms of capital city Bengaluru:\n",d);
    while(flag!=0) //loop to
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  City records database is empty");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            if(search==1)
            {
                printf("\n   No city records found within the given distance\n");
            }
            flag=0;
        }
        else
        {
            curr=curr->next;
            if(d>=(curr-
>dist)) //prints record if dist is less than or equal to user specified distance
            {
                printf("\n=>| Name: %s \n  | Coordinate X: %.2f (Latitiude) \n  | Coordina
te Y: %.2f(Longitude) \n  | Distance from Capital: %f kms\n",curr->name,curr->x,curr-
>y,curr->dist);
                search=0;
            }
        }
    }

    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;
}

void append(struct node** head_ref) //Function to insert a new city record into the databa
se(LL)
{
    /* 1. allocate node */
    struct node* new_node = (struct node*)malloc(sizeof(struct node));

    struct node* last = *head_ref; /* used in step 5*/

    /* 2. put in the data  */
    printf("\n\nEnter city name:");
    scanf("%s",(new_node->name));

    printf("\nEnter Latitude(X):");
    scanf("%f",&(new_node->x));

    printf("\nEnter Longitude(Y):");
    scanf("%f",&(new_node->y));
```

```
    new_node->dist=distance(new_node->x,new_node->y);

    /* 3. This new node is going to be the last node, so
          make next of it as NULL*/
    new_node->next = NULL;

    /* 4. If the Linked List is empty, then make the new
          node as head */
    if (*head_ref == NULL) {
        new_node->prev = NULL;
        *head_ref = new_node;
        return;
    }

    /* 5. Else traverse till the last node */
    while (last->next != NULL)
        last = last->next;

    /* 6. Change the next of last node */
    last->next = new_node;

    /* 7. Make last node as previous of new node */
    new_node->prev = last;

    return;
}

void display(struct node* head_ref) //function to display all records from database(LL)
{
    struct node* curr=NULL;
    curr=head_ref;
    int flag=1;
    while(flag!=0) //loop to print data in the list
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  No records in database");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            printf("\n  end\n");
            flag=0;
        }
        else
        {
            curr=curr->next;
            printf("\n=>| Name: %s \n  | Coordinate X: %.2f (Latitiude) \n  | Coordinate Y
: %.2f(Longitude) \n  | Distance from Capital: %f kms\n",curr->name,curr->x,curr->y,curr-
>dist);
        }


    }
```

```
    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;
}

void searchName(struct node* head_ref,char k_name[]) //Function to search record by name o
f the city
{
   struct node* curr=NULL;
   curr=head_ref;
   int flag=1,search=1;
    while(flag!=0 && search==1)
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  Database is empty");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            printf("\n  Record not found in the database\n");
            flag=0;
        }
        else
        {
            curr=curr->next;
            if(strcmp(k_name,curr->name)==0)
            {
                search=0;
                printf("\nRecord found!");
                printf("\n=>| Name: %s \n  | Coordinate X: %.2f (Latitiude) \n  | Coordina
te Y: %.2f(Longitude) \n  | Distance from Capital: %f kms\n",curr->name,curr->x,curr-
>y,curr->dist);
                break;
            }
        }


    }

    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;

}

void searchCord(struct node* head_ref,float X, float Y) //Function to search record by coo
rdinates of the city
{
   struct node* curr=NULL;
```

```
   curr=head_ref;
   int flag=1,search=1;
    while(flag!=0 && search==1) //loop to print data in the list
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  Database is empty");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            printf("\n  Record not found in the database\n");
            flag=0;
        }
        else
        {
            curr=curr->next;
            if(X==curr->x && Y==curr->y)
            {
                search=0;
                printf("\n  Record found!");
                printf("\n=>| Name: %s \n  | Coordinate X: %.2f (Latitiude) \n  | Coordina
te Y: %.2f(Longitude) \n  | Distance from Capital: %f kms\n",curr->name,curr->x,curr-
>y,curr->dist);
                break;
            }
        }
    }

    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;
}

void delNode(struct node** h_ref,struct node* del) //function to delete node
{
    if(*h_ref==NULL||del==NULL) //to check if list address or node to be
    return;                     //deleted exists or is it null

    if (*h_ref == del)   //when head node is to be deleted
        *h_ref = del->next;

    if (del->next != NULL) //when last node is to be deleted
        del->next->prev = del->prev;

    if (del->prev != NULL)
        del->prev->next = del->next;

    free(del);
}

void delName(struct node* head_ref,char k_name[]) //function to delete a record(node) w.r.
t. city name
```

```
{
   struct node* curr=NULL;
   curr=head_ref;
   int flag=1,search=1;
    while(flag!=0 && search==1) //loop to print data in the list
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  Database is empty");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            printf("\n  Record not found in the database\n");
            flag=0;
        }
        else
        {
            curr=curr->next;
            if(strcmp(k_name,curr->name)==0)
            {
                search=0;
                printf("\n  Record found!");
                delNode(&head_ref,curr);
                printf("\n  Record Deleted!");
                break;
            }
        }
    }
    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;

}

void delCord(struct node* head_ref,float X, float Y) //function to delete a record(node) w
.r.t. city coordinates
{
   struct node* curr=NULL;
   curr=head_ref;
   int flag=1,search=1;
    while(flag!=0 && search==1) //loop to print data in the list
    {
        if(curr->next==NULL && curr->prev==NULL)
        {
            printf("\n  Database is empty");
            flag=0;
        }
        else if(curr->next==NULL && curr->prev!=NULL)
        {
            printf("\n  Record not found in the database\n");
            flag=0;
        }
```

```
        else
        {
            curr=curr->next;
            if(X==curr->x && Y==curr->y)
            {
                search=0;
                printf("\n  Record found!");
                delNode(&head_ref,curr);
                printf("\n  Record Deleted!");
                break;
            }
        }
    }

    while(curr->prev!=NULL) //loop to bring current node pointer back to head node
    {
        curr=curr->prev;
    }
    flag=1;
    return;
}


int main()
{
    struct node* head=NULL; // node pointer declaration, pointing to Null
    head=(struct node*)malloc(sizeof(struct node));  //allocating memory and returning a p
ointer to it
    struct node* curr=NULL;
    curr=head;

    head->prev=NULL;
    head->next=NULL;

    int o; // variable to store option number

    printf("\nWelcome to City Database!\n (Capital City: Bengaluru)");

    while(o!=6)  // while loop keeps running until user enters 6
    {
        printf("\n_____________________________________________________________________\n"
        "\nChoose any option[1-6]:\n"
        " 1. Enter new records\n"
        " 2. Delete records\n"
        " 3. Display all records\n"
        " 4. Search for a record\n"
        " 5. Display records within a distance from a given city\n"
        " 6. Quit\n");
        printf("\nEntered Option:");
        scanf("%d",&o);

        if(o==1)  //function to enter new records
        {
            char ch='y';
            while(ch=='y'||ch=='Y')
            {
```

```
            append(&head);
            printf("\n  Enter more records? [y/n]: ");
            scanf(" %c",&ch);
        }
    }

    else if(o==2)  //function to delete records
    {
        char city[50];
        float x1,y1;
        char ch='y';
        int p;
        while(ch=='y'||ch=='Y')
        {
         printf("\n Search record to be deleted by: \n"
         "  1. City Name\n"
         "  2. City Coordinates\n"
         "  Your choice[1 or 2]:");
         scanf(" %d",&p);
         if(p==1)
         {
             printf("\n   Enter name of the city to be deleted:");
             scanf("%s",city);
             delName(head,city);
         }

         else if(p==2)
         {
             printf("\n   Enter coordinates of the city to be deleted:");
             printf("\n    Enter Latitude(X):");
             scanf(" %f",&x1);
             printf(" \n    Enter Longitude(Y): ");
             scanf("%f",&y1);
             delCord(head,x1,y1);
         }


         else
         {
             printf("\nInvalid option!");
         }

         printf("\nContinue Deleting? [y/n]: ");
         scanf(" %c",&ch);
        }
    }

    else if(o==3) //funtion to Display all records
    { display(head);}

    else if(o==4) //Search for a record
    {
        char city[50];
        float x1,y1;
        char ch='y';
        int p;
```

```
            while(ch=='y'||ch=='Y')
            {
             printf("\n Search by: \n"
             "  1. City Name\n"
             "  2. City Coordinates\n"
             "  Your choice[1 or 2]:");
             scanf(" %d",&p);
             if(p==1)
             {
                 printf("\n   Enter name of the city to be searched:");
                 scanf(" %s",city);
                 searchName(head,city);
             }

             else if(p==2)
             {
                 printf("\n   Enter Latitude(X):");
                 scanf(" %f",&x1);
                 printf(" \n   Enter Longitude(Y): ");
                 scanf("%f",&y1);
                 searchCord(head,x1,y1);
             }

             else
             {
                 printf("\n  Invalid option!");
             }

             printf("\n  Continue Searching? [y/n]\n");
             scanf(" %c",&ch);
            }
        }

        else if(o==5) // Display records within a distance from a given city
        {
            char ch='y';
            while(ch=='y'||ch=='Y')
            {
                print(head);
                printf("\n Enter another distance? [y/n]: ");
                scanf(" %c",&ch);
            }

        }

        else if(o==6) //funtion to exit the program
        {break;}

        else
        {
            printf("Invalid Input! Enter again");
        }

    }
    return 0;
}
```

5. RESULTS

CASE-1: Insertion of records into the database

```
Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:1


Enter city name:Mangalore

Enter Latitude(X):12.91

Enter Longitude(Y):74.85

  Enter more records? [y/n]: y


Enter city name:Udupi

Enter Latitude(X):13.34

Enter Longitude(Y):74.74

  Enter more records? [y/n]: y


Enter city name:Mysore

Enter Latitude(X):12.29

Enter Longitude(Y):76.63

  Enter more records? [y/n]: n
```

CASE-2: Printing all records present in the database

```
_____________________________________________________________

Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:3

=>| Name: Mangalore
  | Coordinate X: 12.91 (Latitiude)
  | Coordinate Y: 74.85(Longitude)
  | Distance from Capital: 297.009552 kms

=>| Name: Udupi
  | Coordinate X: 13.34 (Latitiude)
  | Coordinate Y: 74.74(Longitude)
  | Distance from Capital: 311.316681 kms

=>| Name: Mysore
  | Coordinate X: 12.29 (Latitiude)
  | Coordinate Y: 76.63(Longitude)
  | Distance from Capital: 128.713593 kms

  end


_____________________________________________________________

Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit
```

CASE-3: Searching for a record in the database

```
Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:4

 Search by:
  1. City Name
  2. City Coordinates
  Your choice[1 or 2]:1

   Enter name of the city to be searched:Mysore

Record found!
=>| Name: Mysore
  | Coordinate X: 12.29 (Latitiude)
  | Coordinate Y: 76.63(Longitude)
  | Distance from Capital: 128.713593 kms

  Continue Searching? [y/n]
y

 Search by:
  1. City Name
  2. City Coordinates
  Your choice[1 or 2]:2

   Enter  Latitude(X):13.34

   Enter Longitude(Y): 74.74
```

```
 Search by:
  1. City Name
  2. City Coordinates
  Your choice[1 or 2]:2

   Enter  Latitude(X):13.34

   Enter Longitude(Y): 74.74

  Record found!
=>| Name: Udupi
  | Coordinate X: 13.34 (Latitiude)
  | Coordinate Y: 74.74(Longitude)
  | Distance from Capital: 311.316681 kms

  Continue Searching? [y/n]
n


--------------------------------------------------------------------------------

Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:
```

CASE-4: Deletion of records from the database

```
Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:2

 Search record to be deleted by:
  1. City Name
  2. City Coordinates
  Your choice[1 or 2]:1

   Enter name of the city to be deleted:Mysore

  Record found!
  Record Deleted!
Continue Deleting? [y/n]: y

 Search record to be deleted by:
  1. City Name
  2. City Coordinates
  Your choice[1 or 2]:2

   Enter coordinates of the city to be deleted:
    Enter Latitude(X):12.29

    Enter Longitude(Y): 76.63

  Record not found in the database

Continue Deleting? [y/n]: n
```

After deletion of record:

```
Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:3

=>| Name: Mangalore
  | Coordinate X: 12.91 (Latitiude)
  | Coordinate Y: 74.85(Longitude)
  | Distance from Capital: 297.009552 kms

=>| Name: Udupi
  | Coordinate X: 13.34 (Latitiude)
  | Coordinate Y: 74.74(Longitude)
  | Distance from Capital: 311.316681 kms

  end


Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:
```

CASE-5: Print all records within a given distance from capital city

```
Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:5

  Enter distance(in Kms): 300

  City records within 300.00 kms of capital city Bengaluru:

=>| Name: Mangalore
  | Coordinate X: 12.91 (Latitiude)
  | Coordinate Y: 74.85(Longitude)
  | Distance from Capital: 297.009552 kms

 Enter another distance? [y/n]: y

  Enter distance(in Kms): 350

  City records within 350.00 kms of capital city Bengaluru:

=>| Name: Mangalore
  | Coordinate X: 12.91 (Latitiude)
  | Coordinate Y: 74.85(Longitude)
  | Distance from Capital: 297.009552 kms

=>| Name: Udupi
  | Coordinate X: 13.34 (Latitiude)
  | Coordinate Y: 74.74(Longitude)
  | Distance from Capital: 311.316681 kms
```

```
  Enter distance(in Kms): 250

  City records within 250.00 kms of capital city Bengaluru:

   No city records found within the given distance

 Enter another distance? [y/n]: n

---

Choose any option[1-6]:
 1. Enter new records
 2. Delete records
 3. Display all records
 4. Search for a record
 5. Display records within a distance from a given city
 6. Quit

Entered Option:
```

6. CONCLUSION

City Database was implemented successfully using Doubly Linked List data structure. Each database record contains the name of the city (a string of arbitrary length) and the coordinates of the city expressed as integer x(latitude) and y(longitude) coordinates. The database allows records to be inserted and deleted by name or coordinate, and searched by name or coordinate. The program allows an additional operation to print all records within a given distance of a capital city(Bengaluru).

Complexity of operations:

a)Insertion of records into the database

Worst Case: O(n)
Best case: O(1)

b)Deletion of records from the database by

- Name of the city
- Coordinates of the city

Worst Case: O(n)
Best case: O(1)

c)Searching for a record in the database by

- Name of the city
- Coordinates of the city

Worst Case: O(n)
Best case: O(1)

d) print all records within a given distance from capital city

Worst Case: O(n)
Best case: O(n)

e)Printing all records present in the database

Worst Case: O(n)
Best case: O(1)

7. References

Coordinates of Cities: www.google.com
Data Structure: https://www.geeksforgeeks.org/doubly-linked-list/