

Data Structures and Algorithms

Social Network Graph

Dated 2020-11-21

Question

Develop a social network website with thousands of users. With the use of data structure to efficiently represent and store the users' profiles. Keep the information about how these users are connected with each other.

Find out the friends for a given user. This search needs to be executed quickly as there are thousands of users.

Find the common friends for two given users. Propose a suitable data structure; implement the insert, delete, search operations and relevant algorithms to solve the problem. Analyse the time and space complexities of all the operations.

Solution

For the entire program visit: <https://github.com/psymbio/social-network-graph>

Initially, I thought of constructing two 2D matrices, one which would hold the information of all the users and the other that holds the data of how these people are connected with each other.

id	username	password	name	email
1	cornetto_tornado	severussnaps	Rick Roll	cornetto@yahoo.co.uk
2	xX-witchyPrincess-Xx	hhsj112@	Chandler Bing	
3	dancingsloths	nothing	Julian Assange	mumbo@jumbo.com

	0	1	2	3	4
0	1	0	1	0	
1	1	1	0	0	
2	1	1	0	0	
3	0	0	1	1	

But with thousands of users, the CSV file of the connections crashed my text-editor. Not only that, but if there was a 25% chance someone was your friend then about 50% of the space used by the graph goes to waste, which is a great deal if you're handling thousands of users.

Then I decided to make an adjacency-list representation of a graph. This would only deal with the ids of each user in the graph to show the connection, saving space.

Now to create the graph:

```
// structure to represent an user node
struct user_node
{
    int dest;
    struct user_node *next;
};

// structure to represent the list of users
struct user_list
{
    struct user_node *head;
};

// graph is an array of adjacency lists.
```

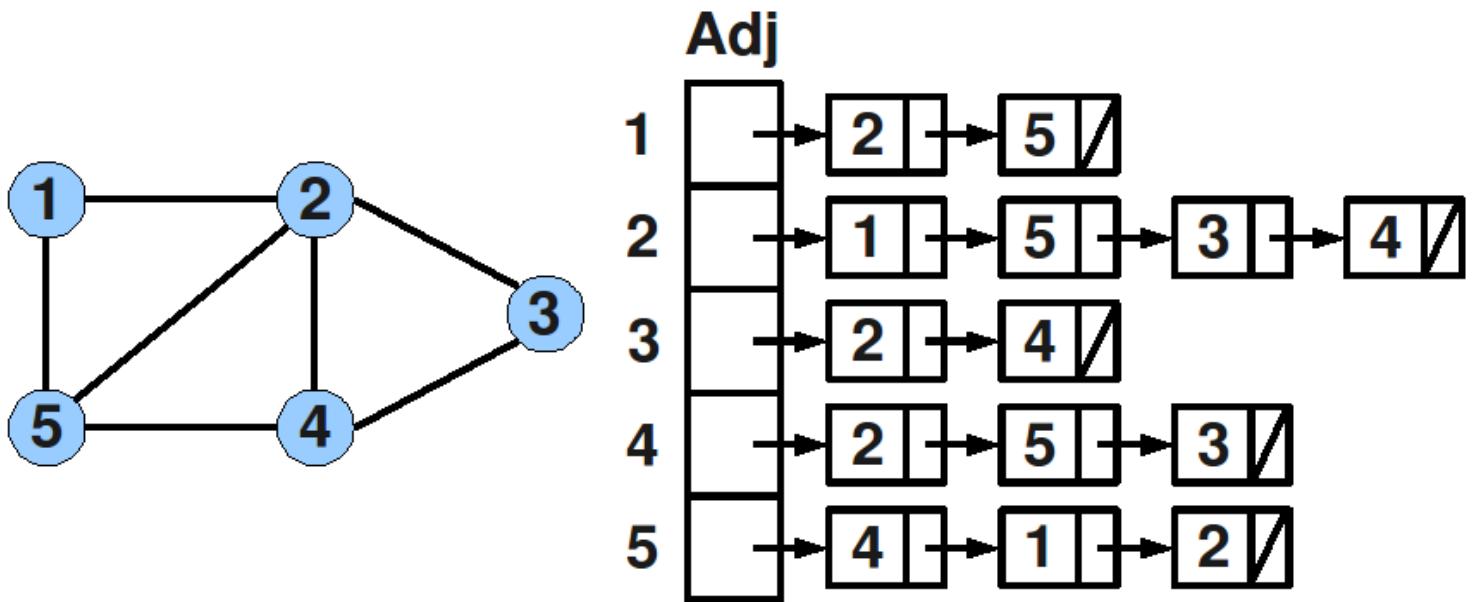


Figure 1: Adjacency Graph

```

struct Graph
{
    int V;
    struct user_list *array;
};

// A utility function to create a new adjacency list node
struct user_node *newuser_node(int dest)
{
    struct user_node *newNode = (struct user_node *)malloc(sizeof(struct user_node));
    newNode->dest = dest;
    newNode->next = NULL;
    return newNode;
}

// create a graph of V vertices
struct Graph *createGraph(int V)
{
    struct Graph *graph = (struct Graph *)malloc(sizeof(struct Graph));
    graph->V = V;
    graph->array = (struct user_list *)malloc(V * sizeof(struct user_list));
    int i;
    for (i = 0; i < V; ++i)
        graph->array[i].head = NULL;
    return graph;
}

```

Now that our graph has been implemented, let's carry out some functionality of the program.

1. Add a Friend

```

// add an edge to an undirected graph
void add_friend(struct Graph *graph, int src, int dest)
{
    // add edge from src to dest
    struct user_node *newNode = newuser_node(dest);
    newNode->next = graph->array[src].head;
    graph->array[src].head = newNode;
}

```

```

// add edge from dest to src
newNode = newuser_node(src);
newNode->next = graph->array[dest].head;
graph->array[dest].head = newNode;
}

```

This simply adds an edge between two user nodes of the graph and takes linear time $O(1)$.

2. Find a Node

```

int find_user(struct Graph *graph, int src)
{
    for (int v = 0; v < graph->V; ++v)
    {
        struct user_node *pCrawl = graph->array[v].head;
        while (pCrawl)
        {
            if (pCrawl->dest == src)
                return 1;
            pCrawl = pCrawl->next;
        }
    }
}

```

This takes $O(n^2)$ time complexity to find a user_node.

3. Remove a Friend

```

void remove_friend(struct Graph *graph, int a, int b)
{
    if (find_user(graph, src) && find_user(graph, dest))
    {
        struct user_node *aCrawl = graph->array[v].head;
        struct user_node *bCrawl = graph->array[v].head;
        while (aCrawl || bCrawl)
        {
            if (aCrawl->dest == b)
                free(graph->array);
            aCrawl = aCrawl->next;
            if (bCrawl->dest == a)
                free(graph->array);
            bCrawl = bCrawl->next;
        }
    }
}

```

This takes $O(n)$ time complexity.

4. Delete Your Account

```

void delete_user(struct Graph *graph, int a)
{
    if (find_user(graph, a))
    {
        for (int v = 0; v < graph->V; ++v)
        {
            struct user_node *aCrawl = graph->array[v].head;
            if (v == a)
            {
                free(graph->array);
            }
        }
    }
}

```

```

void delete_your_account(struct Graph *graph, MYSQL *conn)
{
    delete_user(graph, atoi(current_uid));
    sprintf(sql_statement, "DELETE FROM user_database where id=%d", atoi(current_uid));
    if (mysql_query(conn, sql_statement))
    {
        finish_with_error(conn);
    }
    printf("\nSo long, and thanks for all the fish.\n");
}

```

Deleting your account deletes the array of friends you had and your information in the database. Since find_user takes $O(n^2)$ so does this.

5. Print Entire Graph

```

void print_friend_graph(struct Graph *graph)
{
    int v;
    for (v = 0; v < graph->V; ++v)
    {
        struct user_node *pCrawl = graph->array[v].head;
        printf("\nFriends of %d are:\n", v);
        while (pCrawl)
        {
            printf("-> %d", pCrawl->dest);
            pCrawl = pCrawl->next;
        }
        printf("\n");
    }
}

```

This prints the entire graph in $O(n^2)$ however this is only a back-end function that normal users cannot run.

6. Print the Friends of a User

```

void print_userid_friends(struct Graph *graph, int v, MYSQL *conn, MYSQL_ROW row)
{
    int status = 0;
    struct user_node *pCrawl = graph->array[v].head;
    while (pCrawl)
    {
        sprintf(sql_statement, "SELECT username, first_name, last_name
        FROM user_database WHERE id=%d", pCrawl->dest);
        if (mysql_query(conn, sql_statement))
        {
            finish_with_error(conn);
        }
        MYSQL_RES *result = mysql_store_result(conn);
        while ((row = mysql_fetch_row(result)) != NULL)
        {
            printf("Username: %s\t", row[0]);
            printf("Name: %s %s\n", row[1], row[2]);
        }
        pCrawl = pCrawl->next;
    }
}

```

For a user with id = v this prints all the information of the friends of that user. This takes $O(n)$ time.

7. Print Common Friends of Two Users

```

void find_common_friends(struct Graph *graph, int a, int b, MYSQL *conn, MYSQL_ROW row)
{

```

```

printf("here 1");
struct user_node *aCrawl = graph->array[a].head;
struct user_node *bCrawl = graph->array[b].head;
printf("here 1");
while (aCrawl || bCrawl)
{
    if (aCrawl->dest == bCrawl->dest)
    {
        sprintf(sql_statement, "SELECT username, first_name, last_name FROM
user_database WHERE id=%d", aCrawl->dest);
        if (mysql_query(conn, sql_statement))
        {
            finish_with_error(conn);
        }
        MYSQL_RES *result = mysql_store_result(conn);
        while ((row = mysql_fetch_row(result)) != NULL)
        {
            printf("Username: %s\t", row[0]);
            printf("Name: %s %s\n", row[1], row[2]);
        }
    }
    aCrawl = aCrawl->next;
    bCrawl = bCrawl->next;
}
}

void print_common_friends(struct Graph *graph, MySQL *conn, MySQL_ROW row)
{
    char user_a[50];
    char user_b[50];
    char user_a_uid[50];
    char user_b_uid[50];
    printf("Enter the first user: ");
    scanf("%s", user_a);
    printf("Enter the second user: ");
    scanf("%s", user_b);
    sprintf(sql_statement, "SELECT id FROM user_database
WHERE username='%s' OR username='%s'", user_a, user_b);
    if (mysql_query(conn, sql_statement) != 0)
    {
        printf("Query failed with the following message:\n");
        printf("%s\n", mysql_error(conn));
        exit(1);
    }
    MYSQL_RES *result = mysql_store_result(conn);
    while ((row = mysql_fetch_row(result)) != NULL)
    {
        strcpy(user_a_uid, row[0]);
        strcpy(user_b_uid, row[1]);
    }
    printf("%s", user_a_uid);
    printf("%s", user_b_uid);
    printf("The common friends of %s and %s are:\n", user_a, user_b);
    find_common_friends(graph, atoi(user_a_uid), atoi(user_b_uid), conn, row);
}

```

8. Print Friend Recommendations

```

void print_friend_recommendations(struct Graph *graph, int v, MySQL *conn, MySQL_ROW row)
{
    struct user_node *pCrawl = graph->array[v].head;
    while (pCrawl)

```

```

{
    if (pCrawl->dest != v)
    {
        sprintf(sql_statement, "SELECT username, first_name, last_name FROM user_database WHERE id=%d", pCrawl->dest);
        if (mysql_query(conn, sql_statement))
        {
            finish_with_error(conn);
        }
        MYSQL_RES *result = mysql_store_result(conn);
        while ((row = mysql_fetch_row(result)) != NULL)
        {
            printf("Recommendation from username: %s\t", row[0]);
            printf("Name: %s %s\n", row[1], row[2]);
            print_userid_friends(graph, pCrawl->dest, conn, row);
            printf("\n\n\n");
        }
    }
    pCrawl = pCrawl->next;
}
}

```

This takes $O(n^k)$ where k is the number of friends user with id = v has.

9. Insert a New Login

```

int sign_up(MYSQL *conn, MYSQL_ROW row)
{
    char uid[50];
    if (mysql_query(conn, "SELECT id FROM user_database
ORDER BY ID DESC LIMIT 1") != 0)
    {
        printf("Query failed with the following message:\n");
        printf("%s\n", mysql_error(conn));
        exit(1);
    }
    MYSQL_RES *result = mysql_store_result(conn);
    while ((row = mysql_fetch_row(result)) != NULL)
    {
        strcpy(uid, row[0]);
    }
    char *new_uid = string_test(uid);
    char username[50], password[50], first_name[50], last_name[50], gender[50];
    char dob[50], phone_number[50], email[50], lives_in[50], college[50];
    printf("Enter your username: ");
    scanf("%s", username);
    printf("Enter your password: ");
    scanf("%s", password);
    printf("Enter your first name: ");
    scanf("%s", first_name);
    printf("Enter your last name: ");
    scanf("%s", last_name);
    printf("Enter your gender(Male/Female): ");
    scanf("%s", gender);
    printf("Enter your date of birth(YYYY-MM-DD): ");
    scanf("%s", dob);
    printf("Enter your mobile number(XXX-XXX-XXXX): ");
    scanf("%s", phone_number);
    printf("Enter your email: ");
    scanf("%s", email);
    printf("Enter your address: ");
    scanf("%s", lives_in);
    printf("Enter your college: ");

```

```

scanf("%s", college);
sprintf(sql_statement, "INSERT INTO user_database(id, username, password,
first_name, last_name, gender, dob, since, phone_number,
email, lives_in, college) VALUES('%s', '%s', '%s', '%s',
'%s', '%s', '2020-11-29', '%s', '%s', '%s', '%s')", new_uid,
username, password,
first_name, last_name, gender, dob, phone_number, email, lives_in, college);
if (mysql_query(conn, sql_statement) != 0)
{
    printf("Query failed with the following message:\n");
    printf("%s\n", mysql_error(conn));
}
else
{
    printf("Great! you've signed up. The program shall exit now,
next time log in with your new credentials.\n");
    exit(0);
}
}

```

This takes constant time.

Output

For the entire program visit: <https://github.com/psymbio/social-network-graph>

Create an account

```

Activities Terminal Sat Nov 21 4:10 PM
vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project$ ./run
1. Enter 1 to log in with existing account.
2. Enter 2 to sign up.
3. Enter 3 to exit.
Enter your choice: 2
Enter your username: galantis
Enter your password: atlantis
Enter your first name: Joe
Enter your last name: Biden
Enter your gender(Male/Female): Male
Enter your date of birth(YYYY-MM-DD): 1987-02-23
Enter your mobile number(XXX-XXX-XXXX): 273-473-6284
Enter your email: joebiden@gmail.com
Enter your address: Washington
Enter your college: EdinburghUni
Great! you've signed up. The program shall exit now, next time log in with your new credentials.
vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project$ vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project$ 

```

View Your Friends

Activities Terminal Sat Nov 21 4:13 PM vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project\$./run

```
1. Enter 1 to log in with existing account.  
2. Enter 2 to sign up.  
3. Enter 3 to exit.  
Enter your choice: 1  
  
Enter your username: galantis  
Enter the password:  
Login successful!  
  
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 1  
  
Your friends are:  
Username: vgriffeymd Name: Valencia Griffey  
Username: erandalss9s Name: Ethelin Randals  
Username: mstirzaker3x Name: Mandi Stirzaker  
Username: blahertyhh Name: Baldwin Laherty  
Username: btimsfu Name: Benedicto Tims  
Username: ymackartancw Name: Yard MacKartan  
Username: sboss3h Name: Salvador Boss  
Username: ghoxeyc5 Name: Gianna Hoxey  
Username: mmarcinkowskii2 Name: Matt Marcinkowski  
Username: uedisql Name: Udale Edis  
Username: wstrussoa Name: Wendel Struss  
Username: rjakeman9q Name: Reidar Jakeman  
Username: dreekenn8 Name: Deanne Reeken  
Username: ktownsend4d Name: Katy Townsend  
Username: jkornaliklike2 Name: Jourdan Kornalik  
Username: ndymickcz Name: Nathalia Dymick  
Username: pfeltham7y Name: Phineas Feltham  
Username: hparadesit Name: Hardy Parades  
Username: mmactruestrieg9 Name: Mirabelle MacTrustrie  
Username: atrolovenzia Name: Ahra Trolovenzia
```

View Friend Recommendations

Activities Terminal Sat Nov 21 4:16 PM vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project\$

```
Login successful!  
  
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 2  
Recommendation from username: rkemittyg Name: Ramsay Kemitt  
Username: username Name: First Last  
Username: mmelchiorlf Name: Moselle Melchior  
Username: dfrickeyae Name: Dunc Frickey  
Username: vnieassah Name: Veronique Nieass  
Username: skenny48 Name: Sholom Kenny  
Username: ygreenerbpb Name: Yurik Greener  
Username: tfenich83 Name: Tadd Fenich  
Username: amizenat Name: Alessandro Mizen  
Username: amorrill94 Name: Anthiathia Morrill  
  
Recommendation from username: bstockwell9h Name: Bryon Stockwell  
Username: username Name: First Last  
Username: ctorricellaag Name: Christean Torricella  
Username: galantis Name: Joe Biden  
Username: bpieruccipn Name: Bron Pierucci  
Username: pconnochie7 Name: Pearline Connochie  
Username: eabeaurb Name: Elle Abeau  
Username: aharcencp Name: Arly Harce  
Username: mivanovic6z Name: Minna Ivanovic  
Username: aspat1k Name: Alva Spat  
Username: mthake7n Name: Morty Thake  
Username: bvaudre41 Name: Brig Vaudre  
  
Recommendation from username: aoloughanekc Name: Ari O'Loughane
```

Make New Friends

Activities Terminal Sat Nov 21 4:17 PM vanillaskies@pop-os: ~/projects/computer-science/data-structures-and-algorithms/final-project

```
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 3  
Enter a username to be friends with: fprimo4k  
You've made a new friend!  
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 1  
Your friends are:  
Username: fprimo4k Name: Ferd Primo  
Username: rkemittyg Name: Ramsay Kemitt  
Username: bstockwell9h Name: Bryon Stockwell  
Username: aolaghanekc Name: Ari O'Loughane  
Username: pcollicott9e Name: Pepillo Collicott  
Username: lkiehnltb Name: Lanita Kiehnlt  
Username: nvampouilleer Name: Nettle Vampouille  
Username: shurlldl Name: Stearne Hurl  
Username: ctorricellaag Name: Christean Torricella  
Username: lpeeble9y Name: Liza Peebles  
Username: esummerside9m Name: Edlin Summerside
```

View Another User's Information and Friends

Activities Terminal Sat Nov 21 4:18 PM vanillaskies@pop-os: ~/projects/computer-science/data-structures-and-algorithms/final-project

```
Enter your choice: 1  
Your friends are:  
Username: fprimo4k Name: Ferd Primo  
Username: rkemittyg Name: Ramsay Kemitt  
Username: bstockwell9h Name: Bryon Stockwell  
Username: aolaghanekc Name: Ari O'Loughane  
Username: pcollicott9e Name: Pepillo Collicott  
Username: lkiehnltb Name: Lanita Kiehnlt  
Username: nvampouilleer Name: Nettle Vampouille  
Username: shurlldl Name: Stearne Hurl  
Username: ctorricellaag Name: Christean Torricella  
Username: lpeeble9y Name: Liza Peebles  
Username: esummerside9m Name: Edlin Summerside  
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 5  
Enter a username to find friends of: lpeeble9y  
lpeeble9y's friends are:  
Username: kolumneyra Name: Karoline O'Lunney  
Username: username Name: First Last  
1. Enter 1 to view your friends.  
2. Enter 2 to view friend recommendations.  
3. Enter 3 to make a new friend.  
4. Enter 4 to view a user's information.  
5. Enter 5 to view a user's friends.  
6. Enter 6 to view common friends of two users.  
7. Enter 7 to delete your account.  
8. Enter 8 to exit.  
Enter your choice: 
```

View Common Friends

Activities Terminal vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project

Login successful!

```

1. Enter 1 to view your friends.
2. Enter 2 to view friend recommendations.
3. Enter 3 to make a new friend.
4. Enter 4 to view a user's information.
5. Enter 5 to view a user's friends.
6. Enter 6 to view common friends of two users.
7. Enter 7 to delete your account.
8. Enter 8 to exit.
Enter your choice: 2
Recommendation from username: rkemittyg Name: Ramsay Kemitt
Username: username Name: First Last
Username: mmelchiorlf Name: Moselle Melchior
Username: dfrickeyae Name: Dunc Frickey
Username: vnieassah = rand Name: Veronique Nieass
Username: skenny48 int j = Name: Sholom Kenny
Username: yggreenerb Name: Yurik Greener
Username: tfenich83dd friend Name: Tadd Fenich
Username: amizenat / print Name: Alessandro Mizenat
Username: amorrill94 Name: Anthiathia Morrill
83 // print friend(graph);
84 // delete social network(graph);
85 // print the adjacency list representation of the above graph
86 Recommendation from username: bstockwell9h Name: Bryon Stockwell
Username: username Name: First Last
Username: ctorricellaag Name: Christean Torricella
Username: galantis Name: Joe Biden recommendations.\n";
Username: bpieruccipm Name: Bron Pierucci
Username: pconnochie7 Name: Pearline Connochie
Username: eabeurb Name: Elle Abeau friends of two users.\n";
Username: aharcenepp Name: Arly Harcecount.\n";
Username: mivanovic62 Name: Minna Ivanovic
Username: aspat1k Name: Alva Spat
Username: mthake7th (menu) Name: Morty Thake
Username: bvaudre41 Name: Brig Vaudre
102 case 1:
103     printf("\nYour friends are: \n");
104     print_userid_friends(graph, atoi(current_uid), conn, row);
105     getchar();
106     setchar();

```

Recommendation from username: aoloughanekc Name: Ari O'Loughane

Username: Username Name: First Last

Delete Your Account

Activities Terminal vanillaskies@pop-os:~/projects/computer-science/data-structures-and-algorithms/final-project

Enter the password:

The password and username you've entered don't match.
Press any enter to continue.

Enter your username: username
Enter the password:

The password and username you've entered don't match.
Press any enter to continue.

Enter your username: galantis
Enter the password:
Login successful!

1. Enter 1 to view your friends.
2. Enter 2 to view friend recommendations.
3. Enter 3 to make a new friend.
4. Enter 4 to view a user's information.
5. Enter 5 to view a user's friends.
6. Enter 6 to view common friends of two users.
7. Enter 7 to delete your account.
8. Enter 8 to exit.

Enter your choice: 4

Enter a username to view information of: klund2n
Username: klund2n Name: Kaitlynn Lund
Born in: 1954-10-07 User since: 2005-02-21
Lives in: 491 Muir Road College: College of St. Elizabeth Email: Klund2n@springer.com Phone number: 514-300-2009

1. Enter 1 to view your friends.
2. Enter 2 to view friend recommendations.
3. Enter 3 to make a new friend.
4. Enter 4 to view a user's information.
5. Enter 5 to view a user's friends.
6. Enter 6 to view common friends of two users.
7. Enter 7 to delete your account.
8. Enter 8 to exit.

Enter your choice: []