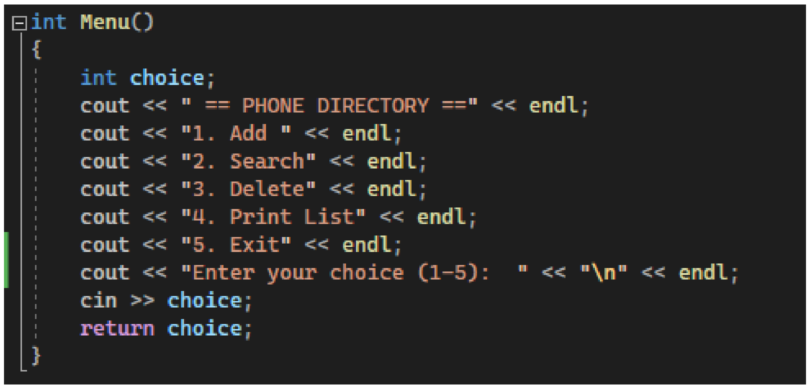
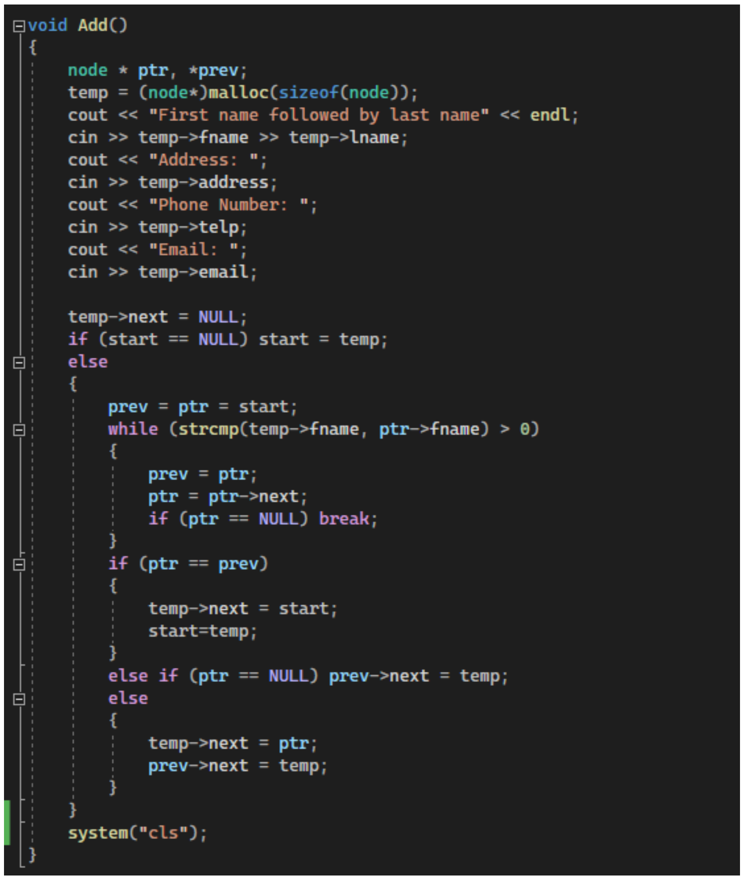
The big Oh for each operation:

1. The complexity of this set of code is o(1), since cout is a constant.



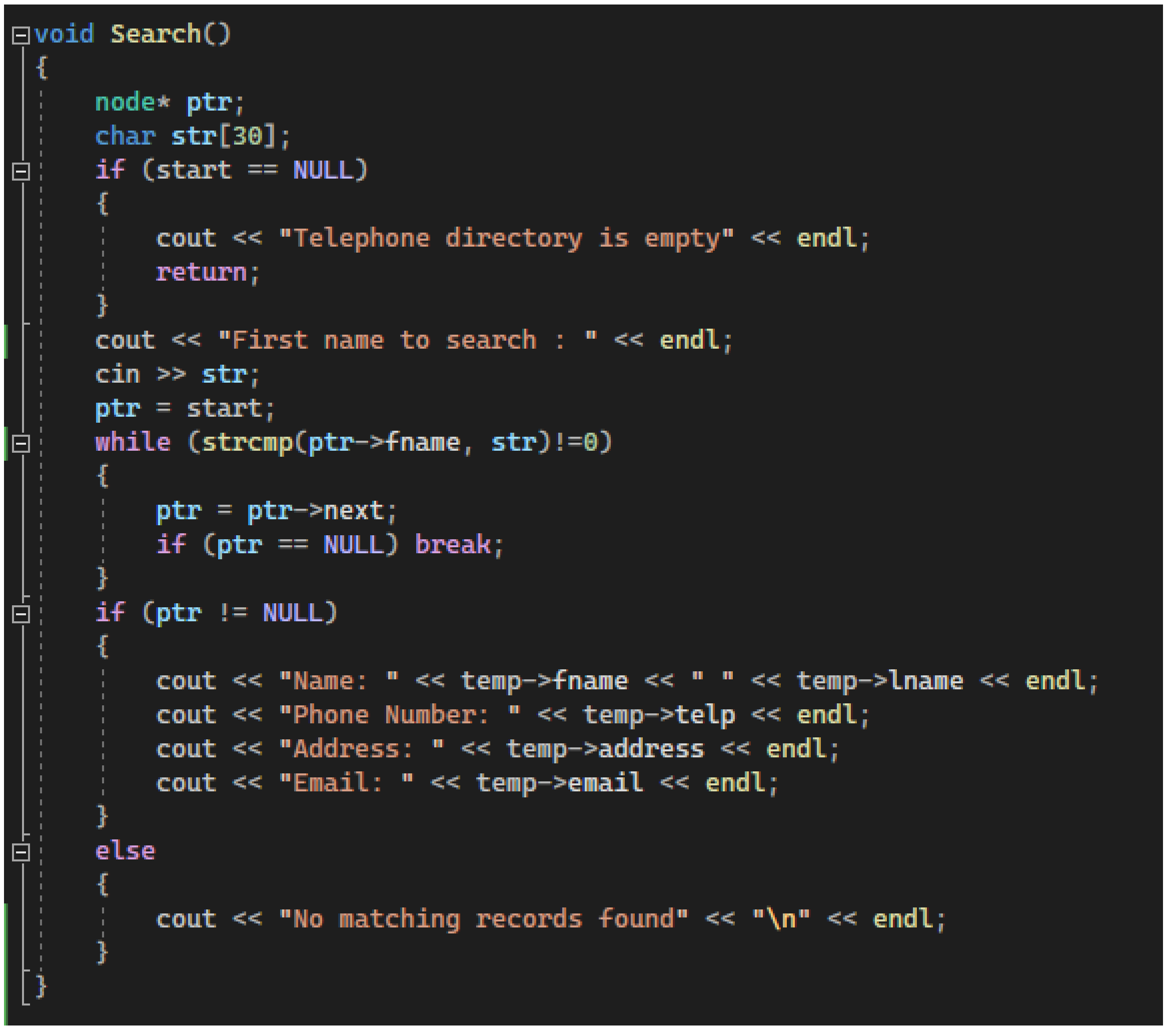
```
int Menu()
{
    int choice;
    cout << " == PHONE DIRECTORY ==" << endl;
    cout << "1. Add " << endl;
    cout << "2. Search" << endl;
    cout << "3. Delete" << endl;
    cout << "4. Print List" << endl;
    cout << "5. Exit" << endl;
    cout << "Enter your choice (1-5):  " << "\n" << endl;
    cin >> choice;
    return choice;
}
```

2. The complexity of this add function is O(n). Since memory allocation complexity is o(n) in single linked list and the rest of the code in the function is constant.



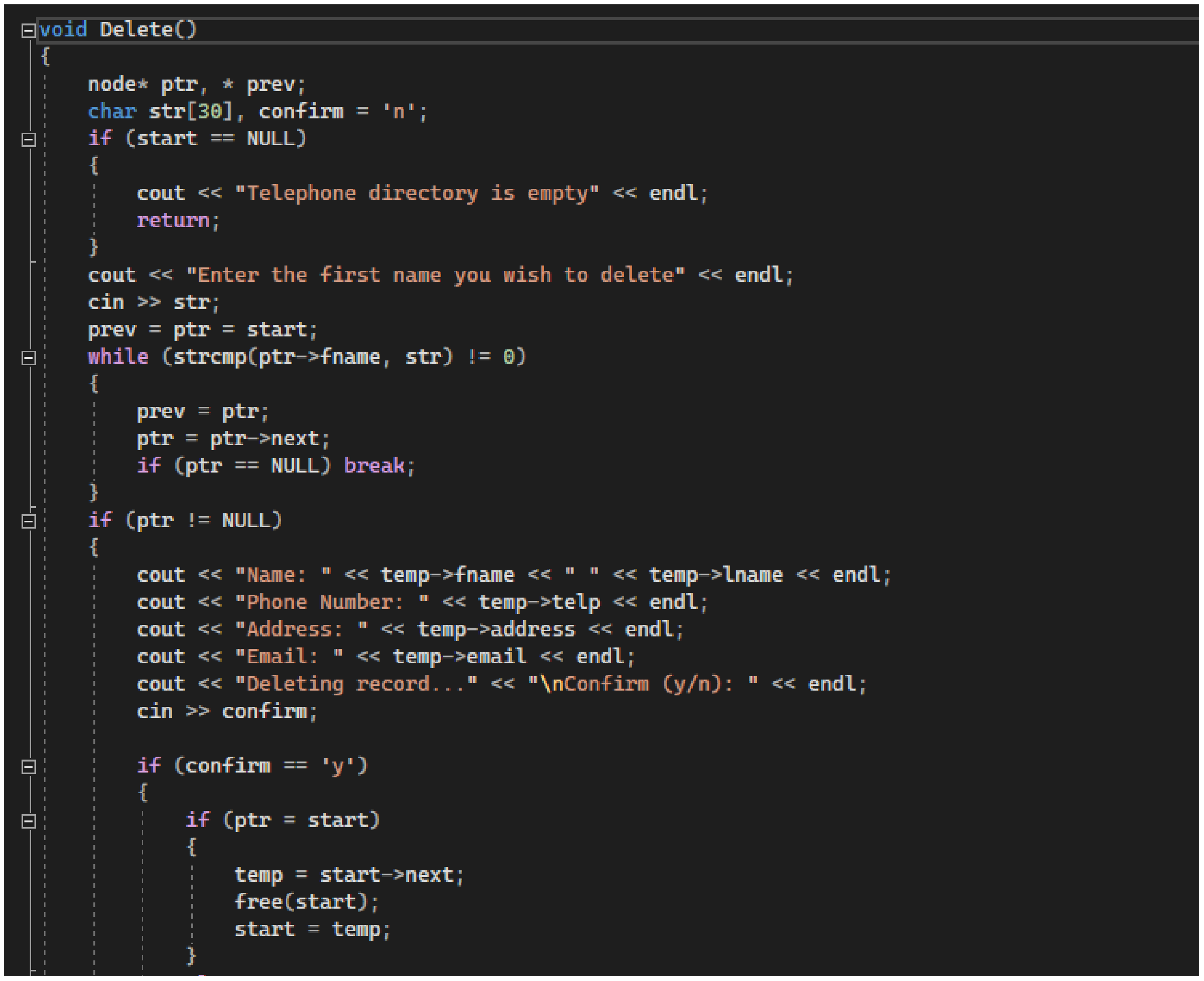
```
void Add()
{
    node * ptr, *prev;
    temp = (node*)malloc(sizeof(node));
    cout << "First name followed by last name" << endl;
    cin >> temp->fname >> temp->lname;
    cout << "Address: ";
    cin >> temp->address;
    cout << "Phone Number: ";
    cin >> temp->telp;
    cout << "Email: ";
    cin >> temp->email;

    temp->next = NULL;
    if (start == NULL) start = temp;
    else
    {
        prev = ptr = start;
        while (strcmp(temp->fname, ptr->fname) > 0)
        {
            prev = ptr;
            ptr = ptr->next;
            if (ptr == NULL) break;
        }
        if (ptr == prev)
        {
            temp->next = start;
            start=temp;
        }
        else if (ptr == NULL) prev->next = temp;
        else
        {
            temp->next = ptr;
            prev->next = temp;
        }
    }
    system("cls");
}
```

3. The complexity of this set of code is o(n) from the while loop.

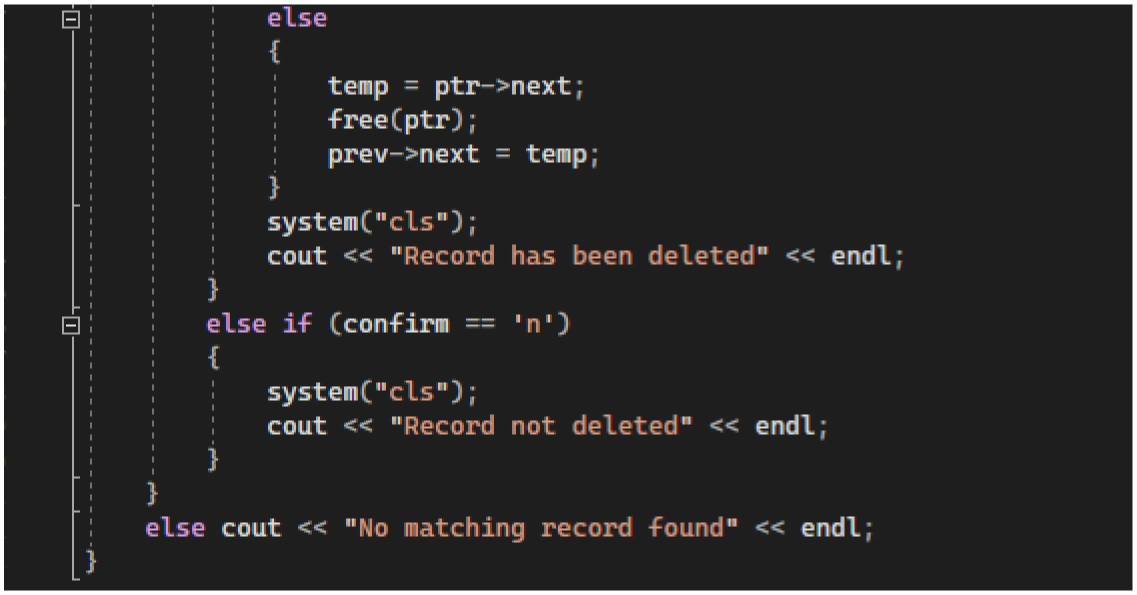


```
void Search()
{
    node* ptr;
    char str[30];
    if (start == NULL)
    {
        cout << "Telephone directory is empty" << endl;
        return;
    }
    cout << "First name to search : " << endl;
    cin >> str;
    ptr = start;
    while (strcmp(ptr->fname, str)!=0)
    {
        ptr = ptr->next;
        if (ptr == NULL) break;
    }
    if (ptr != NULL)
    {
        cout << "Name: " << temp->fname << " " << temp->lname << endl;
        cout << "Phone Number: " << temp->telp << endl;
        cout << "Address: " << temp->address << endl;
        cout << "Email: " << temp->email << endl;
    }
    else
    {
        cout << "No matching records found" << "\n" << endl;
    }
}
```

4. The complexity of the code below is o(n) which is a worst case scenario, single linked list usually has o(1) as it complexity

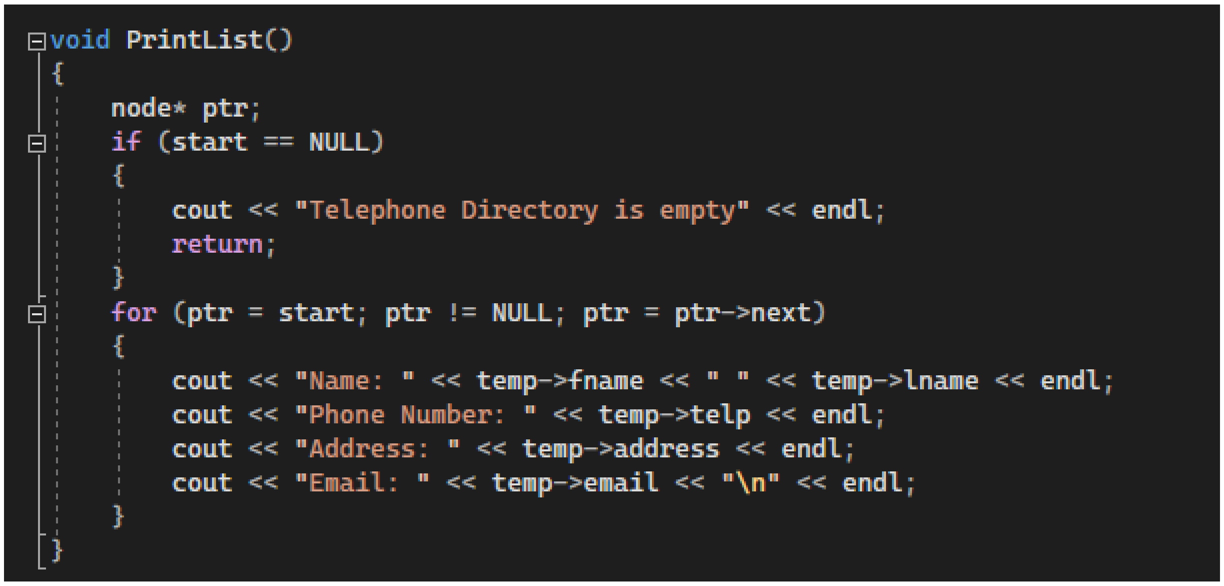


```cpp
void Delete()
{
    node* ptr, * prev;
    char str[30], confirm = 'n';
    if (start == NULL)
    {
        cout << "Telephone directory is empty" << endl;
        return;
    }
    cout << "Enter the first name you wish to delete" << endl;
    cin >> str;
    prev = ptr = start;
    while (strcmp(ptr->fname, str) != 0)
    {
        prev = ptr;
        ptr = ptr->next;
        if (ptr == NULL) break;
    }
    if (ptr != NULL)
    {
        cout << "Name: " << temp->fname << " " << temp->lname << endl;
        cout << "Phone Number: " << temp->telp << endl;
        cout << "Address: " << temp->address << endl;
        cout << "Email: " << temp->email << endl;
        cout << "Deleting record..." << "\nConfirm (y/n): " << endl;
        cin >> confirm;

        if (confirm == 'y')
        {
            if (ptr = start)
            {
                temp = start->next;
                free(start);
                start = temp;
            }
```



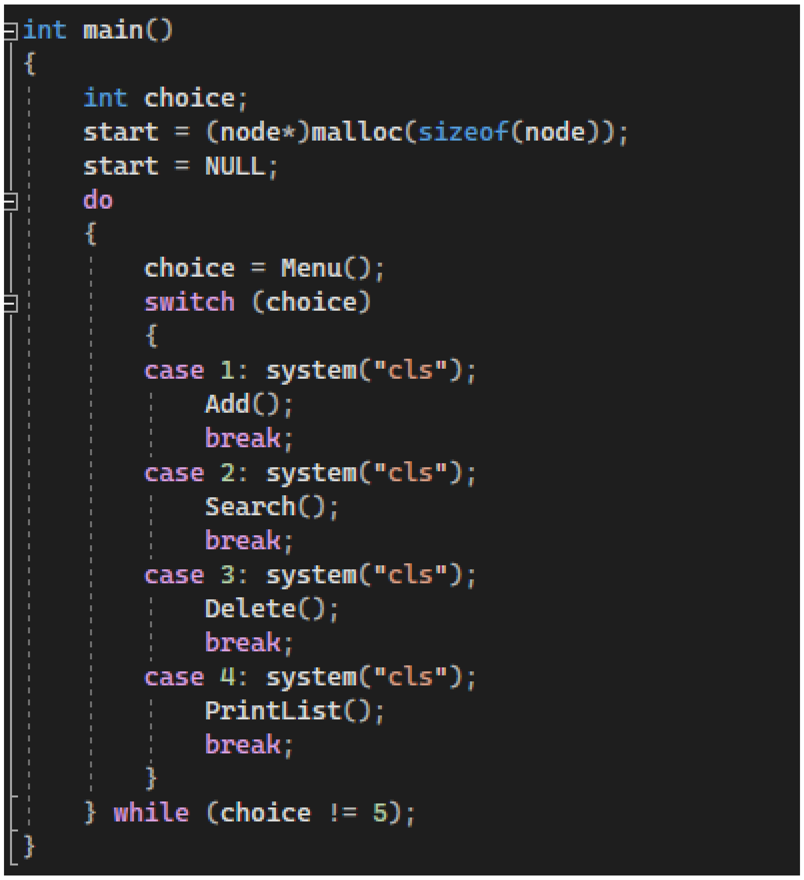
```cpp
            else
            {
                temp = ptr->next;
                free(ptr);
                prev->next = temp;
            }
            system("cls");
            cout << "Record has been deleted" << endl;
        }
        else if (confirm == 'n')
        {
            system("cls");
            cout << "Record not deleted" << endl;
        }
    }
    else cout << "No matching record found" << endl;
}
```

5. The complexity of this is o(1) since everything here is constant, even though it has a for loop inside of it.



```
void PrintList()
{
    node* ptr;
    if (start == NULL)
    {
        cout << "Telephone Directory is empty" << endl;
        return;
    }
    for (ptr = start; ptr != NULL; ptr = ptr->next)
    {
        cout << "Name: " << temp->fname << " " << temp->lname << endl;
        cout << "Phone Number: " << temp->telp << endl;
        cout << "Address: " << temp->address << endl;
        cout << "Email: " << temp->email << "\n" << endl;
    }
}
```

6. The complexity of the main function is o(n)



```
int main()
{
    int choice;
    start = (node*)malloc(sizeof(node));
    start = NULL;
    do
    {
        choice = Menu();
        switch (choice)
        {
        case 1: system("cls");
            Add();
            break;
        case 2: system("cls");
            Search();
            break;
        case 3: system("cls");
            Delete();
            break;
        case 4: system("cls");
            PrintList();
            break;
        }
    } while (choice != 5);
}
```