Assignment 9 Programming

Name : - Jeevan Rajpurohit

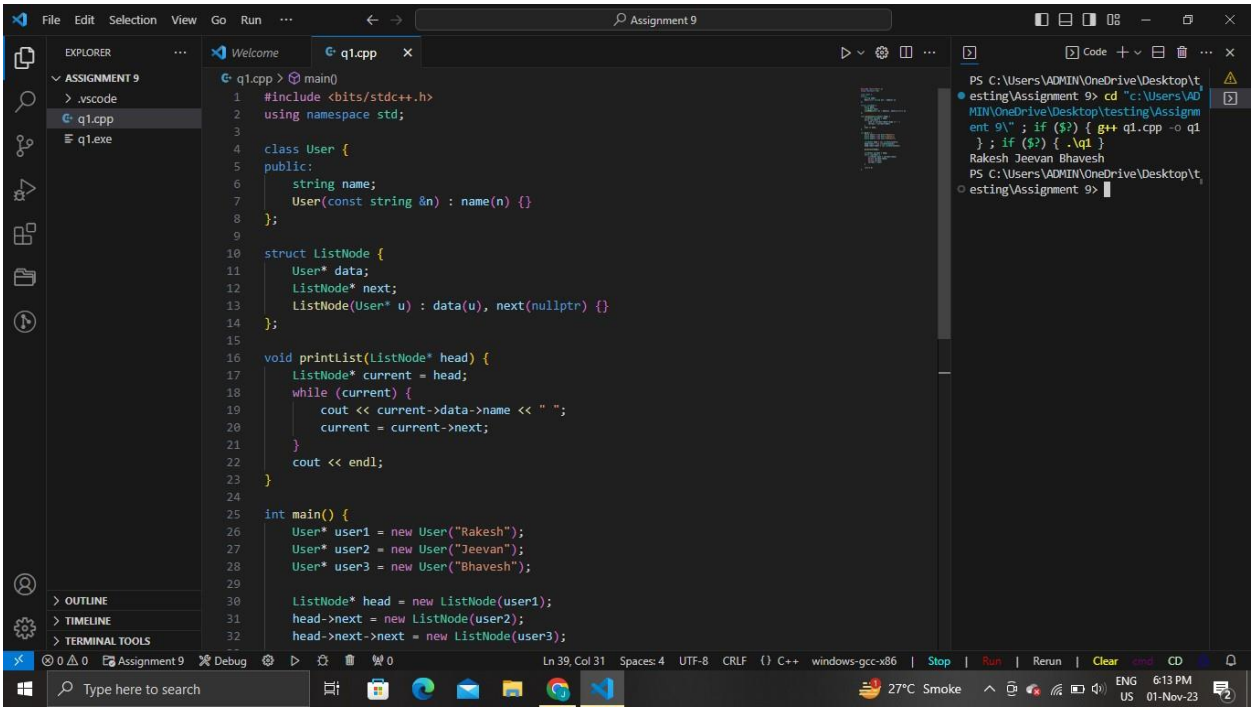
Student ID-202312090

1. Use templates to write generic code that can store user objects in a linked list. The linked list node should have pointer to the user objects ie they should not store the user objects inside linked list nodes. Use unit tests to test your template library code for all scenarios including possible edge cases.

Ans->
```cpp
#include <bits/stdc++.h>

using namespace std;

class User {
public:
    string name;
    User(const string &n) : name(n) {}
};

struct ListNode {
    User* data;
    ListNode* next;
    ListNode(User* u) : data(u), next(nullptr) {}
};

void printList(ListNode* head) {
    ListNode* current = head;
    while (current) {
        cout << current->data->name << " ";
        current = current->next;
    }
    cout << endl;
}

int main() {
    User* user1 = new User("Rakesh");
    User* user2 = new User("Jeevan");
    User* user3 = new User("Bhavesh");

    ListNode* head = new ListNode(user1);
    head->next = new ListNode(user2);
    head->next->next = new ListNode(user3);
```

```
    printList(head);

    ListNode* current = head;
    while (current) {
        ListNode* next = current->next;
        delete current->data;
        delete current;
        current = next;
    }

    return 0;
}
```

Output-



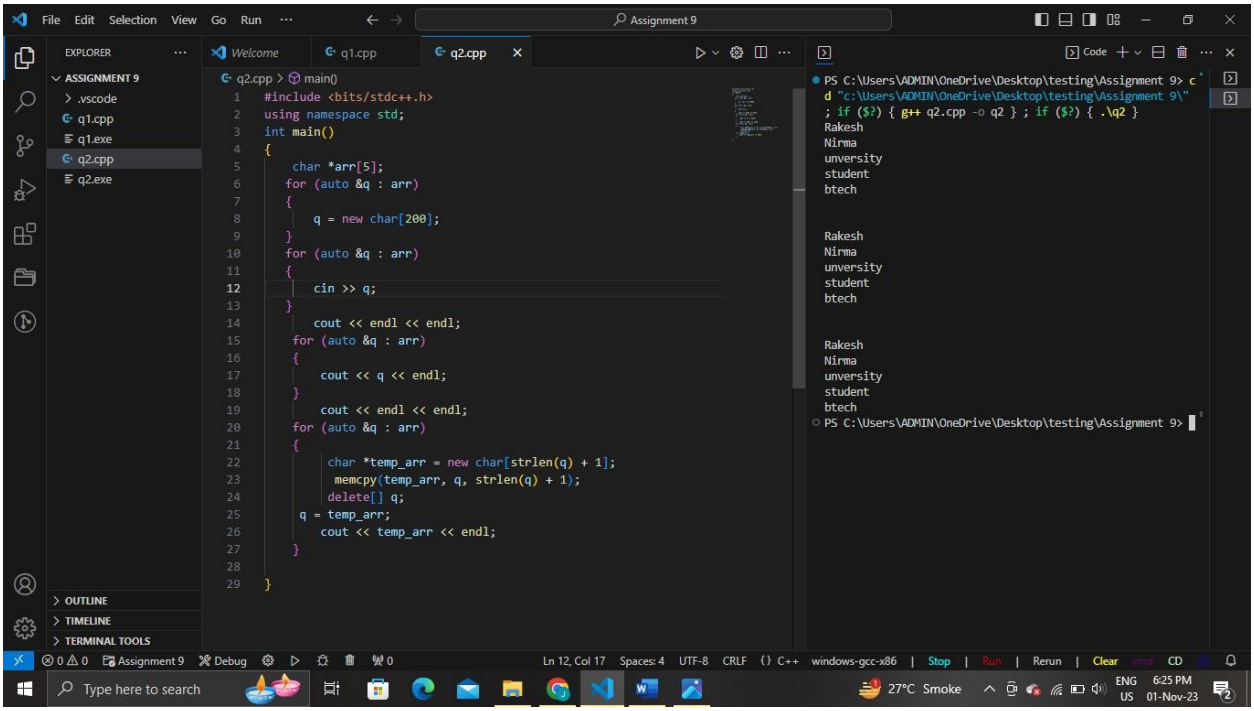
2).

Read data from file that has 100+ records to store the information in the linked list. You can get data from online resources, like top 100 contries by GDP or, 100 cities and data related to them. Top 100 noble prize winners - year, subject and name of scientist. Read this data and use the template code above to store the information in the linked list. Assume that data is in random order.

```
Ans->#include <bits/stdc++.h>
```

```
using namespace std;
int main()
{
    char *arr[5];
   for (auto &q : arr)
   {
       q = new char[200];
   }
   for (auto &q : arr)
   {
       cin >> q;
   }
       cout << endl << endl;
    for (auto &q : arr)
    {
        cout << q << endl;
    }
        cout << endl << endl;
    for (auto &q : arr)
    {
         char *temp_arr = new char[strlen(q) + 1];
          memcpy(temp_arr, q, strlen(q) + 1);
         delete[] q;
     q = temp_arr;
        cout << temp_arr << endl;
    }

}
```

Output:-



3).

Now, we want to have another linked list on top of this linked list to provide to users sorted data. The data can be sorted using any field of the object that has been stored in the first linked list. Use callback (ie std::function) to provide a comparison function to the user of this template library, and create a linked list of nodes that are pointing to first linked list nodes such that iterating through this 2nd list would give the data in sorted order. You can implement any sorting algorithm for this.

Use free store as much as possible (and therefore, ensure that the memory is deleted properly)

Ans->

```
#include <bits/stdc++.h>

using namespace std;
struct Tempt
{
    char *name;
    int age;
    char *game;
};
int main()
{
   unique_ptr<Tempt> tempt{new Tempt()};
```

```
    char *input{new char[100]};
        cin >> input;
   tempt->name = new char[strlen(input) + 1];
        strcpy(tempt->name, input);
            cin >> tempt->age;
            cin >> input;
   tempt->game = new char[strlen(input) + 1];
        strcpy(tempt->game, input);
            cout << " ************* " << endl;
            cout << tempt->name << endl;
            cout << tempt->age << endl;
            cout << tempt->game << endl;
    delete[] tempt->name;
    delete[] tempt->game;
}
```

Output:-