ENSF 614 – Fall 2021

Lab 7 – Tuesday, November 16

Student Name: Aastha Patel and Bhavyai Gupta

Submission date: November 16, 2021

**Exercise C – Developing Singleton Pattern in C++**

**Client\_A.h**

|  |
| --- |
| /\*   \* File Name:               Client\_A.h   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "User.h"  #include "LoginServer.h"  #ifndef CLIENT\_A\_H  #define CLIENT\_A\_H  class Client\_A{  public:      Client\_A();      //  PROMISES: constructor of Client\_A object and initializes its data member        Client\_A(const Client\_A& src);      //  REQUIRES: source to refer Client\_A object      //  PROMISES: constructor of new Client\_A object and sets its instance data member to LoginServer      Client\_A& operator = (const Client\_A& rhs);      //  REQUIRES: rhs will refer to a Client\_A's object      //  PROMISES: copy and assign the data members of the rhs object to the Client\_A        void add(string username, string password);      //  REQUIRES: username and password of Client\_A      //  PROMISES: add a new user of Client\_A to list      User\* validate(string username, string password);      //  PROMISES: returns a pointer based on passed arguments    private:      LoginServer\* instance;  };  #endif |

**Client\_A.cpp**

|  |
| --- |
| /\*   \* File Name:               Client\_A.cpp   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "User.h"  #include "Client\_A.h"  #include <iostream>  using namespace std;  Client\_A::Client\_A() {      instance = LoginServer::getInstance();  }  Client\_A::Client\_A(const Client\_A& src) {      instance = LoginServer::getInstance();  }  Client\_A& Client\_A::operator = (const Client\_A& rhs) {      if (this != &rhs) {          instance = LoginServer::getInstance();      }      return \*this;  }  void Client\_A::add(string username, string password) {      instance->add(username, password);  }  User\* Client\_A::validate(string username, string password) {      User\* foundUser = instance->validate(username, password);      return foundUser;  } |

**Client\_B.h**

|  |
| --- |
| /\*   \* File Name:               Client\_B.h   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "User.h"  #include "LoginServer.h"  #ifndef CLIENT\_B\_H  #define CLIENT\_B\_H  class Client\_B {  public:      Client\_B();      //  PROMISES: constructor of Client\_B object and initializes its data member      Client\_B(const Client\_B& src);      //  REQUIRES: source to refer Client\_B object      //  PROMISES: constructor of new Client\_B object and sets its instance data member to LoginServer      Client\_B& operator = (const Client\_B& rhs);      //  REQUIRES: rhs will refer to a Client\_B's object      //  PROMISES: copy and assign the data members of the rhs object to the Client\_B      void add(string username, string password);      //  REQUIRES: username and password of Client\_B      //  PROMISES: add a new user of Client\_B to list      User\* validate(string username, string password);      //  PROMISES: returns a pointer based on passed arguments  private:      LoginServer\* instance;  };  #endif |

**Client\_B.cpp**

|  |
| --- |
| /\*   \* File Name:               Client\_B.cpp   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "User.h"  #include "Client\_B.h"  #include <iostream>  using namespace std;  Client\_B::Client\_B() {      instance = LoginServer::getInstance();  }  Client\_B::Client\_B(const Client\_B& src) {      instance = LoginServer::getInstance();  }  Client\_B& Client\_B::operator = (const Client\_B& rhs) {      if (this != &rhs) {          instance = LoginServer::getInstance();      }      return \*this;  }  void Client\_B::add(string username, string password) {      instance->add(username, password);  }  User\* Client\_B::validate(string username, string password) {      User\* foundUser = instance->validate(username, password);      return foundUser;  } |

**LoginServer.h**

|  |
| --- |
| /\*   \* File Name:               LoginServer.h   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include <vector>  #include "User.h"  #ifndef LOGIN\_SERVER\_H  #define LOGIN\_SERVER\_H  class LoginServer{    public:      static LoginServer\* getInstance();      //  PROMISES: returns single instance of LoginServer      void add(string username, string password);      //  REQUIRES: username and password      //  PROMISES: add new users as per the given arguments      User\* validate(string username, string password);      //  REQUIRES: username and password      //  PROMISES: returns pointer as per the arguments    private:      LoginServer();      //  PROMISES: constructor to create new LoginServer object      LoginServer(const LoginServer& src);      //  REQUIRES: source refrence to other object      //  PROMISES: create copy of object      LoginServer& operator = (const LoginServer& rhs);      //  REQUIRES: rhs reference to refer object of LoginServer      //  PROMISES: copy and assign the data members of the rhs object to the LoginServer        vector<User> users;      static LoginServer\* instance;  };  #endif |

**LoginServer.cpp**

|  |
| --- |
| /\*   \* File Name:               LoginServer.cpp   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "LoginServer.h"  #include "User.h"  #include <iostream>  #include <string>  using namespace std;  LoginServer\* LoginServer::instance = 0;  LoginServer::LoginServer() {  }  LoginServer::LoginServer(const LoginServer& src) {      instance = LoginServer::getInstance();      users = vector<User> (users);  }  LoginServer& LoginServer::operator=(const LoginServer& rhs) {      if (this != &rhs) {          instance = LoginServer::getInstance();          users = vector<User> (users);      }      return \*this;  }  LoginServer\* LoginServer::getInstance(){      if (instance == NULL) {          instance = new LoginServer;      }      return instance;  }  void LoginServer::add(string username, string password) {      struct User user;      user.password = password;      user.username = username;      for (int i = 0; i < (int) users.size(); i++) {          if (users.at(i).username.compare(username) == 0) {              cout << "Username already exists" << endl;              return;          }      }      users.push\_back(user);      cout << "User successfully added!" << endl;  }  User\* LoginServer::validate(string username, string password) {      for (int i = 0; i < (int) users.size(); i++) {          if (users.at(i).username.compare(username) == 0 && users.at(i).password.compare(password) == 0){              return &users.at(i);          }      }      return 0;  } |

**User.h**

|  |
| --- |
| /\*   \* File Name:               User.h   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include <string>  using namespace std;  #ifndef USER\_H  #define USER\_H  struct User {      string username;      string password;  };  #endif |

**Main.cpp**

|  |
| --- |
| /\*   \* File Name:               main.cpp   \* Course:                  ENSF 614 - Fall 2021   \* Lab # and Assignment #:  Lab 7 Exercise C   \* Lab section:             C01   \* Completed by:            Aastha Patel, Bhavyai Gupta   \* Submission Date:   \*/  #include "Client\_A.h"  #include "Client\_B.h"  #include <iostream>  using namespace std;  int main() {      Client\_A ca;      cout << "Created a new Client\_A object called ca ..." << endl;      cout << "adding two usernames, Jack and Judy, by client ca ..." << endl;      ca.add("Jack", "apple5000");      ca.add("Judy", "orange$1234");      Client\_B cb;      cout << "Created a new Client\_B object called cb ... " << endl;      cout << "Adding two usernames called Jim and Josh, by client cb ..." << endl;      cb.add("Jim", "brooks$2017");      cb.add("Josh", "mypass2000");      cout << "Now adding another username called Jim by client ca.\n";      cout << "It must be avoided because a similar username already exits ..." << endl;      ca.add("Jim", "brooks$2017");      cout << "Another attempt to add username called Jim, but this time by client cb,\n";      cout << "with a different password\n";      cout << "It must be avoided again ..." << endl;      cb.add("Jim", "br$2017");      cout << "Now client cb validates existence of username Jack and his password: " << endl;      if( User \*u = cb.validate("Jack", "apple5000"))          cout << "Found: username: " << u->username << " and the password is: " << u->password <<  endl;      else          cout << "Username or password NOT found" << endl;      cout << "Now client ca validates existence of username Jack with a wrong password. " << endl;      if( User \*u = ca.validate("Jack", "apple4000"))          cout << "Found: username is: " << u->username << " and password is: " << u->password <<  endl;      else          cout << "Username or password NOT found" << endl;      cout << "Trying to make a new Client\_A object which is a copy of client ca:" << endl;      Client\_A ca2 = ca;      cout << "Adding a usernames called Tim by client ca2 ..." << endl;      cb.add("Tim", "blue\_sky");      cout << "Make a new Client\_A object called ca3:" << endl;      Client\_A ca3;      cout << "Make ca3 a copy of ca2:" << endl;      ca3 = ca2;      cout << "Now client ca3 validates existence of username Tim and his password: " << endl;      if( User \*u = ca3.validate("Tim", "blue\_sky"))          cout << "Found: username: " << u->username << " and the password is: " << u->password <<  endl;      else          cout << " Tim NOT found" << endl;  #if 0      cout << "Lets now make a couple of objects of LoginServer by main funciton:" << endl;      LoginServer x;      LoginServer y = x;      x = y;      cout << "Now LoginServer x validates existence of username Tim and his password: " << endl;      if( User \*u = y.validate("Tim", "blue\_sky"))          cout << "Found: username: " << u->username << " and the password is: " << u->password <<  endl;      else          cout << "Tim NOT found" << endl;  #endif      return 0;  } |

**Output**

|  |
| --- |
|  |

**Ex-C Part II**

* Program was not able to create LoginServer object.
* The reason is singleton object requires constructor should be private. And the new object can only be created only if there is no previous object was created of the same class.