Zoe Veale

005850386

CSE 461

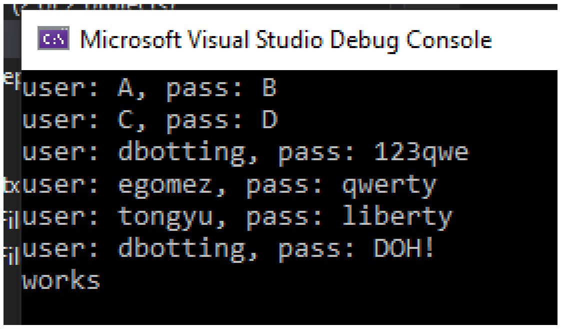
```
#include "passwordFile.h"

int main(int argc, char* argv) {

  PasswordFile passfile("password.txt");
  passfile.AddPassword("dbotting", "123qwe");
  passfile.AddPassword("egomez", "qwerty");
  passfile.AddPassword("tongyu", "liberty");
  passfile.AddPassword("dbotting", "DOH!");
  passfile.PrintFile();

  if (passfile.CheckPassword("dbotting", "123qwe")) {
    printf("works\n");
  }

  return 0;
}
```



```
user: A, pass: B
user: C, pass: D
user: dbotting, pass: 123qwe
user: egomez, pass: qwerty
user: tongyu, pass: liberty
user: dbotting, pass: DOH!
works
```

```
#ifndef PASSWORDFILE_H
#define PASSWORDFILE_H

#include <fstream>
#include <string>
#include <iostream>
#include <vector>
#include <iterator>
#include <algorithm>

class PasswordFile {
public:

  PasswordFile(std::string filename);// opens the file and reads the names/passwords in
the vectors user and password.
  void AddPassword(std::string newuser, std::string newpassword); //this adds a new
user/password to the vectors and writes the vectors to the file filename
  bool CheckPassword(std::string user, std::string passwd); // returns true if user
exists and password matches
  static void NewSalt(int ns);
  void PrintFile();

private:

  std::string filename; // the file that contains password information
  std::vector<std::string> user; // the list of usernames
  std::vector<std::string> password; // the list of passwords
  void SynchFile();// writes the user / password vectors to the password file
  static int salt;

  std::string Encrypt(std::string s);
  //std::string Decrypt(std::string s);
};

#endif // !PASSWORDFILE_H
```

```
#include "passwordFile.h"

int PasswordFile::salt;

PasswordFile::PasswordFile(std::string filename) {
  salt = 1;

  std::fstream inFile;
  this->filename = filename;
  inFile.open(this->filename, std::ios::in | std::ios::out);

  std::string inData;


  while (!inFile.eof()) {
    inFile >> inData;
    user.push_back(inData);
    inFile >> inData;
    password.push_back(inData);

  }
  user.pop_back();
  password.pop_back();
  inFile.close();
}

void PasswordFile::AddPassword(std::string newuser, std::string newpassword) {

  user.push_back(newuser);
  password.push_back(newpassword);
  SynchFile();
}

bool PasswordFile::CheckPassword(std::string user, std::string passwd) {
  for (int i = 0; i < user.size(); i++) {
    if (this->user[i].c_str() == user && this->password[i].c_str() == passwd) {
      return true;
    }
  }
  return false;
}

void PasswordFile::PrintFile() {
  for (int i = 0; i < user.size(); i++) {
    printf("user: %s, pass: %s\n", user[i].c_str(), password[i].c_str());
  }
}

void PasswordFile::NewSalt(int ns) {
  salt = ns;
}

std::string PasswordFile::Encrypt(std::string s) {
  for (int i = 0; i < s.size(); i++) {
    s[i] += salt;
  }
  return s;
}
```

```
void PasswordFile::SynchFile() {
  std::fstream inFile;
  inFile.open(filename, std::ios::out);

  for (int i = 0; i < user.size(); i++) {
    inFile << user[i].c_str() << " ";
    inFile << password[i].c_str() << " \n";
  }
  inFile.close();
}
```