# Лабораторная работа №4

**Тестирование ПО методом «стеклянного (белого) ящика»**

1. **Цель работы**
   1. Тестирование программного кода по методологии белого ящика;
   2. Исправление обнаруженных ошибок.
2. **Литература**
   1. Фленов М. Е. Библия C#. — 5-е изд., перераб. и доп. / М. Е. Фленов — СПб.: БХВ-Петербург, 2022. — 464 с.
   2. Куликов, С. C. Тестирование программного обеспечения. Базовый курс : практ. пособие. / С. С. Куликов. — Минск: Четыре четверти, 2020. — 294 с
3. **Подготовка к работе**
   1. Повторить теоретический материал (см. п.2).
   2. Изучить описание лабораторной работы.
4. **Основное оборудование**
   1. Персональный компьютер.
5. **Задание**
   1. Создайте проект консольного приложения C# и скопируйте в него код из приложения п.9.1;
   2. Выполнить тестирование приложения методом белого ящика, задокументировать и классифицировать все обнаруженные ошибки;
   3. Исправить обнаруженные ошибки, выполнить повторное тестирование;
   4. Составить отчет по проделанной работе.
6. **Порядок выполнения работы**
   1. Повторить теоретический материал п. 3.1;
   2. Выполнить тестирование ПО п. 5.1-5.3;
   3. Ответить на контрольные вопросы п. 8;
   4. Заполнить отчет п. 7.
7. **Содержание отчета**
   1. Титульный лист;
   2. Цель работы;
   3. Протокол тестирования;
   4. Ответы на контрольные вопросы п. 6.3;
   5. Вывод по проделанной работе.
8. **Контрольные вопросы**
   1. Что такое метод тестирования белого ящика и чем он отличается от метода черного ящика?
   2. Перечислите достоинства и недостатки тестирования методом белого ящика?
9. **Приложение**
   1. **Код программы**

|  |
| --- |
| namespace ErrorsApp  {  using System;  using System.Collections.Generic;  internal class Program  {  static Dictionary<string, string> userCredentials = new Dictionary<string, string>();  static List<User> users = new List<User>();  static void Main(string[] args)  {  bool exit = false;  bool isAuthenticated = false;  while (!exit)  {  Console.WriteLine("\nМеню:");  if (!isAuthenticated)  {  Console.WriteLine("1. Авторизоваться");  Console.WriteLine("2. Зарегистрироваться");  Console.WriteLine("3. Выйти из программы");  }  else  {  Console.WriteLine("1. Добавить пользователяя");  Console.WriteLine("2. Удалить пользователя");  Console.WriteLine("2. Найти пользователя по имени");  Console.WriteLine("4. Вывести всех пользователей");  Console.WriteLine("5. Выйти из учетной записи");  Console.WriteLine("6. Выйти из программы");  }    Console.Write("Выберите опцию: ");  string choice = Console.ReadLine();  if (!isAuthenticated)  {  switch (choice)  {  case "1":  isAuthenticated = Authorize();  break;  case "2":  Register();  break;  case "3":  exit = true;  break;  default:  Console.WriteLine("Неверный выбор. Попробуйте снова.");  break;  }  }  else  {  switch (choice)  {  case "1":  AddUser();  break;  case "2":  RemoveUser();  break;  case "3":  FindUser();  break;  case "4":  DisplayUsers();  break;  case "5":  isAuthenticated = false;  Console.WriteLine("Вы вышли из учетной записи.");  break;  case "6":  exit = true;  break;  default:  Console.WriteLine("Неверный выбор. Попробуйте снова.");  break;  }  }  }  }  static bool Authorize()  {  Console.WriteLine("Введите имя пользователя:");  string username = Console.ReadLine();  Console.WriteLine("Введите пароль:");  string password = Console.ReadLine();  if (userCredentials[username] == password)  {  Console.WriteLine("Успешная авторизация!");  return true;  }  else  {  Console.WriteLine("Неверное имя пользователя или пароль.");  return false;  }  }  static void Register()  {  Console.WriteLine("Введите имя пользователя для регистрации:");  string username = Console.ReadLine();  if (string.IsNullOrEmpty(username))  {  Console.WriteLine("Имя пользователя не может быть пустым.");  }  if (userCredentials.ContainsKey(username))  {  Console.WriteLine("Пользователь с таким именем уже существует.");  }  Console.WriteLine("Введите пароль:");  string password = Console.ReadLine();  if (password.Length < 8)  {  Console.WriteLine("Пароль слишком короткий. Минимум 8 символов.");  }  userCredentials.Add(username, password);  Console.WriteLine("Пользователь успешно зарегистрирован.");  }  static void AddUser()  {  Console.WriteLine("Введите имя пользователя:");  string name = Console.ReadLine();  Console.WriteLine("Введите возраст пользователя:");  int age = Convert.ToInt32(Console.ReadLine());  if (age < 0)  {  Console.WriteLine("Возраст не может быть отрицательным.");  }  users.Add(new User(name, age));  Console.WriteLine("Пользователь добавлен.");  }  static void RemoveUser()  {  Console.WriteLine("Введите имя пользователя для удаления:");  string name = Console.ReadLine();  User userToRemove = users.Find(u => u.Name == name);  if (userToRemove != null)  {  users.Remove(userToRemove);  Console.WriteLine("Пользователь удален.");  }  else  {  Console.WriteLine("Пользователь не найден.");  }  }  static void FindUser()  {  Console.WriteLine("Введите имя пользователя для поиска:");  string name = Console.ReadLine();  User userFound = users.Find(u => u.Name == name);  if (userFound != null)  {  Console.WriteLine("Найден пользователь: {userFound.Name}, возраст {userFound.age}");  }  else  {  Console.WriteLine("Пользователь не найден.");  }  }  static void DisplayUsers()  {  if (users.Count == 0)  {  Console.WriteLine("Список пользователей пуст.");  return  }  for (int i = 0; i <= users.Count; i++)  {  Console.WriteLine($"Имя: {users[i].Name}, Возраст: {users[i].Age}");  }  }  }  class User  {  public string Name;  public int Age;  public User(string name, int age)  {  Name = name;  Age = age;  }  }  } |

* 1. **Протокол тестирования**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание ошибки | Тип ошибки | Местоположение ошибки | Некорректный код | Предложения по исправлению |
|  |  |  |  |  |
|  |  |  |  |  |