МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ЕВФРОСИНИИ ПОЛОЦКОЙ»

Факультет информационных технологий

Кафедра технологий программирования

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «О-ОТПиСП»

Тема: «Конструкторы и деструкторы»

|  |  |
| --- | --- |
| Выполнил: | студент группы 23-ИТ-1  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Ю. Страпко |
| Проверил: | ассистент кафедры ТП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.В. Васильева |

|  |
| --- |
|  |

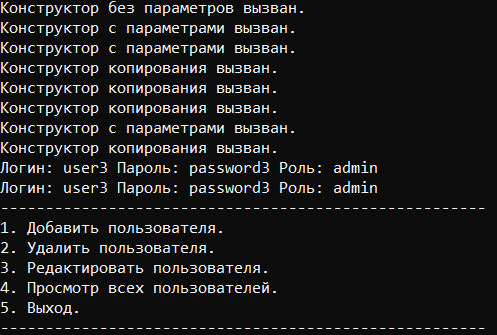
**Цель работы:**Научиться описывать конструкторы и деструкторы для класса, понимать различия между конструкторами. Понимать, в каких случаях могут быть вызваны конструкторы копирования, понимать время жизни объекта.

1. **Практическая часть**

Листинг 1.1 – Файл Program.cs

1. class User
2. {
3. private string login;
4. private string password;
5. private string role;
6. public User(string login, string password, string role)
7. {
8. Console.WriteLine("Конструктор с параметрами вызван.");
9. this.login = login;
10. this.password = password;
11. this.role = role;
12. }
13. public User()
14. {
15. Console.WriteLine("Конструктор без параметров вызван.");
16. }
17. public User(User user)
18. {
19. Console.WriteLine("Конструктор копирования вызван.");
20. this.login = user.login;
21. this.password = user.password;
22. this.role = user.role;
23. }
24. public void MoveFrom(User user)
25. {
26. Console.WriteLine("Метод перемещения вызван.");
27. this.login = user.login;
28. this.password = user.password;
29. this.role = user.role;
30. user.login = null;
31. user.password = null;
32. user.role = null;
33. }
34. ~User()
35. {
36. Console.WriteLine("Деструктор вызван.");
37. }
38. public void ShowInfo()
39. {
40. Console.WriteLine($"Логин: {login} Пароль: {password} Роль: {role}");
41. }
42. public string GetLogin() => login;
43. public string GetRole() => role;
44. public string GetPassword() => password;
45. public void ChangePassword(string newPassword)
46. {
47. password = newPassword;
48. }
49. public void ChangeRole(string newRole)
50. {
51. role = newRole;
52. }
53. }
54. class Application
55. {
56. private string name;
57. private int volume;
58. private List<User> users;
59. public Application(string name, int volume)
60. {
61. this.name = name;
62. this.volume = volume;
63. users = new List<User>();
64. }
65. public void Add(User user)
66. {
67. users.Add(user);
68. }
69. public void Remove(string login)
70. {
71. users.RemoveAll(u => u.GetLogin() == login);
72. }
73. public void RedactUser(string login, string newPassword, string ewRole)
74. {
75. foreach (var user in users)
76. {
77. if (user.GetLogin() == login)
78. {
79. user.ChangePassword(newPassword);
80. user.ChangeRole(newRole);
81. return;
82. }}}
83. public void CreateUser(string login, string password, string role)
84. {
85. User user = new User(login, password, role);
86. Add(user);
87. }
88. public void CreateCopies(User user, int count)
89. {
90. for (int i = 0; i < count; i++)
91. {
92. User copy = new User(user);
93. Add(copy);
94. }
95. }
96. public void ShowUser()
97. {
98. foreach (var user in users)
99. {
100. user.ShowInfo();
101. }
102. }
103. }
104. class Program
105. {
106. static void Main()
107. {
108. Application application = new Application("Управление пользователями", 200);
109. User user1 = new User();
110. User user2 = new User("user2", "password2", "admin");
111. application.Add(user2);
112. application.CreateUser("user3", "password3", "user");
113. application.CreateCopies(user2, 3);
114. User user3 = new User("user3", "password3", "admin");
115. User user4 = new User(user3);
116. user4.ShowInfo();
117. user3.ShowInfo();
118. while (true)
119. {
120. Console.WriteLine("--------------------------------");
121. Console.WriteLine("1. Добавить пользователя.");
122. Console.WriteLine("2. Удалить пользователя.");
123. Console.WriteLine("3. Редактировать пользователя.");
124. Console.WriteLine("4. Просмотр всех пользователей.");
125. Console.WriteLine("5. Выход.");
126. Console.WriteLine("-------------------------------");
127. string choise = Console.ReadLine();
128. switch (choise)
129. {
130. case "1":
131. Console.Write("Введите логин: ");
132. string login = Console.ReadLine();
133. if (string.IsNullOrWhiteSpace(login))
134. {
135. Console.WriteLine("Некорректный ввод.");
136. break;
137. }
138. Console.Write("Введите пароль: ");
139. string password = Console.ReadLine();
140. if (string.IsNullOrWhiteSpace(password))
141. {
142. Console.WriteLine("Некорректный ввод.");
143. break;
144. }
145. Console.Write("Введите роль: ");
146. string role = Console.ReadLine();
147. f (string.IsNullOrWhiteSpace(role))
148. {
149. Console.WriteLine("Некорректный ввод.");
150. break;
151. application.Add(new User(login, password, role));
152. break;
153. case "2":
154. Console.Write("Введите логин для удаления: ");
155. login = Console.ReadLine();
156. if (string.IsNullOrWhiteSpace(login))
157. {
158. Console.WriteLine("Некорректный ввод.");
159. break;
160. }
161. plication.Remove(login);
162. break;
163. case "3":
164. Console.Write("Введите логин для редактирования: ");
165. login = Console.ReadLine();
166. if (string.IsNullOrWhiteSpace(login))
167. {
168. Console.WriteLine("Некорректный ввод.");
169. break;
170. }
171. Console.Write("Введите новый пароль: ");
172. password = Console.ReadLine();
173. if (string.IsNullOrWhiteSpace(password))
174. {
175. Console.WriteLine("Некорректный ввод.");
176. break;
177. }
178. sole.Write("Введите новую роль: ");
179. role = Console.ReadLine();
180. if (string.IsNullOrWhiteSpace(role))
181. {
182. Console.WriteLine("Некорректный ввод.");
183. break;
184. }
185. application.RedactUser(login, password, role);
186. break;
187. case "4":
188. application.ShowUser();
189. break;
190. case "5":
191. return;
192. default:
193. Console.WriteLine("Некорректный ввод.");
194. break;
195. }}}}

**2.Скриншоты результата**

****

**Рисунок 2.1 – Меню управления**

**3. Пояснение алгоритма**

Данный код представляет собой консольное приложение на C#, которое управляет пользователями. Вот краткое описание его работы:

Класс User:

Содержит три приватные поля: login, password, role. Конструктор принимает эти параметры и инициализирует поля. Методы: ShowInfo(): выводит информацию о пользователе. Геттеры для доступа к полям: GetLogin(), GetRole(), GetPassword(). ChangePassword(string newPassword): изменяет пароль. ChangeRole(string newRole): изменяет роль. Конструктор с параметрами: принимает логин, пароль и роль, инициализирует поля. Конструктор без параметров: создает пустого пользователя. Конструктор копирования: создает копию существующего пользователя. Деструктор: Вызывается при уничтожении объекта и выводит сообщение в консоль.

Класс Application:

Хранит название приложения, объем и список пользователей. Методы: Add(User user): добавляет пользователя в список. Remove(string login): удаляет пользователя по логину. RedactUser(string login, string newPassword, string newRole): редактирует информацию о пользователе. ShowUser(): выводит информацию о всех пользователях.

Класс Program:

Содержит метод Main(), который запускает главный цикл приложения. В цикле выводится меню с выбором действий (добавить, удалить, редактировать пользователей, просмотреть всех пользователей или выйти). В зависимости от выбора пользователя выполняются соответствующие действия, такие как ввод данных и вызов методов класса Application.

**Вывод:** в ходе данной лабораторной работы были изучено создание классов и объектов этих классов, вызов методов и обращение к данным через методы этих объектов, передача объектов других классов через методы, понимание влияния модификаторов доступа на видимость данных и методов.