 нужно исправить тот код что мне дали под тему, (разработка автоматизированной системы хранения и обработки информации товара в аптеке) на основании того, что есть. А также нужно (после исправления) как можно более подробное описание программного кода(по модулям)

1. #include <stdio.h>
2. #include <conio.h>
3. #include <stdlib.h>
5. typedef struct A {
6. struct A \*next;
7. char surname[32], name[32], thirdname[32], sex;
8. int age;
9. } element;
11. element \*AddElement() {
12. element \*B;
13. B = (element \*)malloc(sizeof(element));
14. printf("Vvedite FIO: ");
15. scanf("%32s %32s %32s", &(B->surname), &(B->name), &(B->thirdname));
16. printf("**\n**Vvedite vozrast: ");
17. scanf("%d", &(B->age));
18. printf("**\n**Vvedite pol (M/F): ");
19. scanf("**\n**%c", &(B->sex));
20. B->next = NULL;
21. return B;
22. }
24. element \*DelElement(element \*start) {
25. element \*newstart;
26. if(start == NULL)
27. return NULL;
28. newstart = start->next;
29. free(start);
30. return newstart;
31. }
33. void PrintQueue(element \*start) {
34. int k = 0;
35. element \*i;
36. i = start;
37. while(i != NULL) {
38. printf("%d: %s %s %s, age: %d, sex: %c**\n**", ++k, i->surname, i->name, i->thirdname, i->age, i->sex);
39. i = i->next;
40. }
41. }
42. void AddElementN(element \*start, int n) {
43. element \*el, \*n\_el;
44. int i;
45. n\_el = (element \*)malloc(sizeof(element));
46. printf("Vvedite FIO: ");
47. scanf("%32s %32s %32s", &(n\_el->surname), &(n\_el->name), &(n\_el->thirdname));
48. printf("**\n**Vvedite vozrast: ");
49. scanf("%d", &(n\_el->age));
50. printf("**\n**Vvedite pol (M/F): ");
51. scanf("**\n**%c", &(n\_el->sex));
52. el = start;
53. for(i = 0; i < (n-1); i++) {
54. if(el != NULL)
55. el = el->next;
56. else
57. return;
58. }
59. n\_el->next = el->next;
60. el->next = n\_el;
61. }
62. void DeleteElement(element \*start, int n) {
63. element \*d\_el, \*el;
64. int i;
65. el = start;
66. for(i = 0; i < (n-2); i++) {
67. if(el != NULL)
68. el = el->next;
69. else
70. return;
71. }
72. d\_el = el->next;
73. el->next = d\_el->next;
74. free(d\_el);
75. }
77. void loadFromText(char filename[32], short mode, element \*\*start, element \*\*end) {
78. element \*el;
79. FILE \*f;
80. if((f = fopen(filename, "rt")) == NULL) {
81. printf("Oshibka pri otkrytii faila");
82. exit(1);
83. }
84. if((mode == 0) || (\*start == NULL) || (\*end == NULL)) {
85. while(\*start != NULL)
86. \*start = DelElement(\*start);
87. rewind(f);
88. el = (element \*)malloc(sizeof(element));
89. fscanf(f, "**\n**%s**\n**%s**\n**%s**\n**%d**\n**%c**\n**", &(el->surname), &(el->name), &(el->thirdname), &(el->age), &(el->sex));
90. el->next = NULL;
91. \*start = \*end = el;
92. } else {
93. rewind(f);
94. }
95. while(!feof(f)) {
96. el = (element \*)malloc(sizeof(element));
97. fscanf(f, "**\n**%s**\n**%s**\n**%s**\n**%d**\n**%c**\n**", &(el->surname), &(el->name), &(el->thirdname), &(el->age), &(el->sex));
98. el->next = NULL;
99. (\*end)->next = el;
100. \*end = el;
101. }
102. fclose(f);
103. }
105. void loadFromBinary(char filename[32], short mode, element \*\*start, element \*\*end) {
106. element \*el;
107. FILE \*f;
108. if((f = fopen(filename, "rb")) == NULL) {
109. printf("Oshibka pri otkrytii faila");
110. exit(1);
111. }
112. if((mode == 0) || (\*start == NULL) || (\*end == NULL)) {
113. while(\*start != NULL)
114. \*start = DelElement(\*start);
115. rewind(f);
116. el = (element \*)malloc(sizeof(element));
117. if(fread(&el, sizeof(element), 1, f) == 0) exit(1);
118. el->next = NULL;
119. \*start = \*end = el;
120. } else {
121. rewind(f);
122. }
123. while(!feof(f)) {
124. el = (element \*)malloc(sizeof(element));
125. if(fread(&el, sizeof(element), 1, f) == 1) {
126. el->next = NULL;
127. (\*end)->next = el;
128. \*end = el;
129. } else {
130. free(el);
131. }
132. }
133. fclose(f);
134. }
135. void saveToText(char filename[32], short mode, element \*start) {
136. FILE \*f;
137. element \*el;
138. if(mode == 0)
139. f = fopen(filename, "wt");
140. else
141. f = fopen(filename, "at");
142. el = start;
143. while(el != NULL) {
144. fprintf(f, "**\n**%s**\n**%s**\n**%s**\n**%d**\n**%c**\n**", el->surname, el->name, el->thirdname, el->age, el->sex);
145. el = el->next;
146. }
147. fclose(f);
148. }
150. void saveToBinary(char filename[32], short mode, element \*start) {
151. FILE \*f;
152. element \*el;
153. if(mode == 0)
154. f = fopen(filename, "wb");
155. else
156. f = fopen(filename, "ab");
157. el = start;
158. while(el != NULL) {
159. fwrite(&el, sizeof(element), 1, f);
160. el = el->next;
161. }
162. fclose(f);
163. }
164. void main() {
165. int act = 0;
166. short mode;
167. char filename[32];
168. clrscr();
169. element \*start, \*end, \*tmp;
170. start = end = NULL;
171. while(act < 10) {
172. printf("1 | New El**\t**2 | Del El**\n**3 | Print queue**\t** **\**
173. 4 | New Element mezhdu 3 i 4**\n**5 | Del 2nd element**\t** **\**
174. 6 | Load from text**\n**7 | Save to text**\t\**
175. 8 | Load from binary**\n**9 | Save to binary**\t\**
176. 10 | Exit**\n\n**Deistvie: ");
177. scanf("%d", &act);
178. printf("**\n**");
179. switch(act) {
180. case 1:
181. tmp = AddElement();
182. if(start == NULL)
183. start = end = tmp;
184. else {
185. end->next = tmp;
186. end = tmp;
187. }
188. **break**;
189. case 2:
190. start = DelElement(start);
191. if(start == NULL)
192. end = start;
193. **break**;
194. case 3:
195. PrintQueue(start);
196. **break**;
197. case 4:
198. AddElementN(start, 3);
199. **break**;
200. case 5:
201. DeleteElement(start, 2);
202. **break**;
203. case 6:
204. printf("Vvedite imya faila: ");
205. scanf("%s", &filename);
206. printf("0 | Perezapisat' ochered'**\t**1 | Dobavit' v konets ocheredi**\n**Vyberite: ");
207. scanf("%d", &mode);
208. loadFromText(filename, mode, &start, &end);
209. **break**;
210. case 7:
211. printf("Vvedite imya faila: ");
212. scanf("%s", &filename);
213. printf("0 | Perezapisat' file**\t**1 | Dobavit' v konets file**\n**Vyberite: ");
214. scanf("%d", &mode);
215. saveToText(filename, mode, start);
216. **break**;
217. case 8:
218. printf("Vvedite imya faila: ");
219. scanf("%s", &filename);
220. printf("0 | Perezapisat' ochered'**\t**1 | Dobavit' v konets ocheredi**\n**Vyberite: ");
221. scanf("%d", &mode);
222. loadFromBinary(filename, mode, &start, &end);
223. **break**;
224. case 9:
225. printf("Vvedite imya faila: ");
226. scanf("%s", &filename);
227. printf("0 | Perezapisat' file**\t**1 | Dobavit' v konets file**\n**Vyberite: ");
228. scanf("%d", &mode);
229. saveToBinary(filename, mode, start);
230. **break**;
231. }
232. printf("**\n**");
233. }
234. }