

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ   ТЕХНІЧНИЙ   УНІВЕРСИТЕТ   УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** 3

з дисципліни “Основи програмування”

## тема “Структури даних, функції, вказівники і файлові потоки.”

|  |  |  |
| --- | --- | --- |
| Виконав  студентки I курсу  групи КП-61  Чумак Надії Русланівни  (*прізвище, ім’я, по батькові*)  варіант № 29 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Штрафні бали:   |  |  | | --- | --- | | **Термін здачі** | **Оформлення звіту** | |  |  | | Нараховані бали:   |  |  |  | | --- | --- | --- | | **Корект. програм (2 бала)** | **Відп. на теор. питання (1 бал)** | **Відп. на прогр. питання (2 бала)** | |  |  |  | | Сумарний бал:   |  | | --- | |  | |

Київ 2016

**Мета роботи**

Навчитися працювати зі структурами, файлами, динамічно виділяти пам’ять, а також вдосконалити та закріпити вміння роботи з вказівниками.

Навчитися оформлювати консольну програму для зручності роботи користувача.

**Постановка завдання**

#### Загальні вимоги до завдання:

Після входу у програму користувач має можливість зробити вибір:

* Створити новий масив даних
* Зчитати масив даних із файлу

Після цього масив даних можна модифікувати:

* Видалити дані із вказаної позиції у масиві
* Перезаписати дані у вказаній позиції на нововведені
* Перезаписати тільки обране поле даних елемента на вказаній позиції

Також дозволити користувачу виконувати спеціальні команди за варіантом.

Дати можливість користувачу зберегти зміни даних на файлову систему, запропонувавши йому ввести назву дискового файлу, у який збережуться дані.

#### **Завдання №1. Розробка структур і функцій**

1. Описати нову струкутуру даних типу **"Столиця країни"**. Визначити у структурі не менше 3-х полів різного типу, серед яких обов'язковими є: строка, ціле число, дробове число, **інша структура**.
2. Виконати декомпозицію завдання на окремі функції і описати їх прототипи у окремому header-файлі:
   1. Описати функції, одна з яких виконує запис даних із одної структури (через вказівник) у строку, та функцію, що виконує зворотнє перетворення.
   2. Описати функцію, що із тексту створює масив структур та зворотню функцію.
   3. Описати функцію, що видаляє структуру із масиву структур по індексу, функцію, що виконує перезапис структури по індексу даними із іншої структури.
   4. Описати функцію, що отримує вказівник на структуру, строку, що позначає поле у структурі та строку, що позначає нове значення вказаного поля у структурі і виконує перезапис поля заданої структури новим значенням.
   5. Описати функцію, що виконує дію над масивом вказівників на структури: **"Знайти всі столиці, у яких довгота більша за Х"**.
3. Реалізувати відповідні функції у окремому файлі з кодом.

#### **Завдання №2. Робота із файловими потоками**

1. Реалізувати функцію для зчитування даних із текстового файлу і перетворення їх у масив структур.
2. Реалізувати функцію для запису масиву структур у текстовий файл.

#### **Завдання №3. Тестування функцій**

1. Протестувати функції, що піддаються тестуванню, за допомогою assert() у окремій функції(-ях). Кожна функція має тестуватись мінімум на 5 тестових випадках.
2. Винести прототипи тестових функцій у окремий header-файл.
3. Винести тестові функції у окремий файл із кодом.

#### **Завдання №4. Розробка CUI-інтерфейсу**

1. Створити CUI - консольний інтерфейс користувача (просто взаємодія з користувачем через консоль) для використання описаного функціоналу.
2. Всі прототипи відповідних функцій CUI описати у окремому header-файлі, а їх реалізацію винести у окремий .c файл із кодом.
3. Тестові функції запускати тільки при заданні певного аргумента командного рядка.
4. У файлі із кодом функції main() не повинно бути жодних інших функцій.
5. Весь ввід користувача потрбно перевіряти на валідність щоби не допускати збоїв у роботі програми.

**Тексти коду програм**

**Завдання 1.**

|  |
| --- |
| Main.c |
| 1. #include "header.h" 2. #include "assert\_.h" 3. int main(int argc,char \*argv[]) { 4. if(argc > 1){ 5. test1(); 6. test2(); 7. test3(); 8. printf ("All asserts are passed\n"); 9. return EXIT\_SUCCESS; 10. } 11. srand((unsigned)time(NULL)); 12. menu(); 13. conReset(); 14. return 0; 15. } |

|  |
| --- |
| header.h |
| 1. #include <stdlib.h> 2. #include <stdio.h> 3. #include <ctype.h> 4. #include <string.h> 5. #include <pbconsole.h> 6. #include <progbase.h> 7. #include <time.h> 8. #include <limits.h> 9. int menu(void); 10. void clearLine(char c,int y,int x,int num,enum conAttribute\_e color); 11. void prebox(int x\_width\_name,int x\_height\_name,enum conAttribute\_e color); 12. void box(int box\_width,int box\_left,int box\_top,int box\_height,int y\_console\_height,enum conAttribute\_e color); 13. struct classification { 14. char platform [80]; 15. char numbers\_of\_player[80]; 16. }; 17. struct Computer\_Game { 18. char fullname[100]; 19. char genre[50]; 20. int year; 21. float rating; 22. struct classification games; 23. }; 24. struct Computer\_Game \*release; 25. void print\_array(struct Computer\_Game \*st, int count); 26. int print\_in\_text\_files(int count,struct Computer\_Game \*st); 27. int load (struct Computer\_Game \*st); 28. void prints(void); 29. int Computer\_Game\_REVERSE\_record(struct Computer\_Game \* st,int count ,char \*string,int checker); 30. void Computer\_Game\_record(struct Computer\_Game \* st,char \*string ,int count,int number); 31. void all\_structure\_ratingX(struct Computer\_Game \* st,int count,float number); 32. void text(struct Computer\_Game \* st,char \*text,int count); 33. void arrays\_of\_structers(struct Computer\_Game \* st,char \*text,int count); 34. void remove\_array(struct Computer\_Game \*st,int count,int number); 35. void rerecording\_1 (struct Computer\_Game \*st,int count,int number); 36. void rerecording(struct Computer\_Game \*st,int count,int number1,int number2); 37. void rerecording\_2(struct Computer\_Game \*st, int count,int number); 38. void input\_data(struct Computer\_Game \*st,int count); 39. void removing (struct Computer\_Game \*st,int count,int number); 40. int valid\_removing(int count); 41. int valid\_rerecord (int count); 42. int valid\_Game\_record(int count); 43. int valid\_recording1(int count); 44. int valid\_recording2(int count); 45. int valid\_Computer\_Reverse\_record (char \*string); 46. int valid\_Computer\_Reverse\_record1 (char \*string); 47. int valid\_Computer\_Reverse\_record2 (char \*string); 48. void print\_ratingX(struct Computer\_Game \* st,int count,float number); |

|  |
| --- |
| structure.c |
| 1. #include "header.h" 2. enum { BUFFER\_SIZE = 100 }; 3. void prints(void) { 4. printf("1.Видалити дані із вказаної позиції у масиві\n"); 5. printf("2.Перезаписати дані у вказаній позиції на нововведені\n"); 6. printf("3.Перезаписати тільки обране поле даних елемента на вказаній позиції\n"); 7. printf("4.Записати дані із одної структури у строку\n"); 8. printf("5.Записати дані із строки в структуру\n"); 9. printf("6.Записати масив структур в текст\n"); 10. printf("7.Знайти всі ігри із рейтингом меншим за Х\n"); 11. printf("8.Перезапис структури по індексу даними із іншої структури\n"); 12. printf("9.Видалити структуру із масиву структур\n"); 13. printf("0.Натисніть ,щоб повернутися до головного меню\n"); 14. } 15. int valid\_Computer\_Reverse\_record1 (char \*string){ 16. int checker = 0; 17. int minCheck = 0; 18. for (int i = 0; string[i] != '\0'; i++) 19. { 20. if (string[i] == '.') 21. minCheck++; 22. if (isalpha (string[i]) == 0 && minCheck > 1) 23. checker = 1; 24. if (isalpha (string[i]) == 0 && string[i] != '.') 25. checker = 1; 26. } 27. return checker; 28. } 29. int valid\_Computer\_Reverse\_record2 (char \*string){ 30. int checker = 0; 31. for (int i = 0; string[i] != '\0'; i++) 32. { 33. if (isdigit (string[i]) ) 34. checker = 1; 35. } 36. return checker; 37. } 38. int valid\_Computer\_Reverse\_record (char \*string){ 39. int checker = 0; 40. int minCheck = 0; 41. if (string[0] == '\n') 42. checker = 1; 43. for (int i = 0; string[i] != '\0'; i++) 44. { 45. if (string[i] == '\n') 46. string[i] = '\0'; 47. } 48. for (int i = 0; string[i] != '\0'; i++) 49. { 50. if (string[i] == '-') 51. minCheck++; 52. if (isalpha (string[i]) == 0 && minCheck > 1) 53. checker = 1; 54. if (isalpha (string[i]) == 0 && string[i] != '-') 55. checker = 1; 56. } 57. return checker; 58. } 59. int Computer\_Game\_REVERSE\_record(struct Computer\_Game \* st,int count ,char \*string,int checker) { 60. if (checker == 0){ 61. sscanf(string," %s %s %d %f %s %s\n", 62. st->fullname, st->genre, &(st->year), &(st->rating), st->games.platform, st->games.numbers\_of\_player); 63. } else { 64. return 1; 65. } 66. return 0; 67. } 68. int valid\_Game\_record(int count){ 69. int number = 0; 70. int checker = 0; 71. int counter = 0; 72. while ( checker != 1) { 73. printf ("Введіть номер структури для запису в строку: "); 74. number = getInt(); 75. if (number > count || number <= 0) { 76. counter ++; 77. } else { 78. checker = 1; 79. } 80. if (counter > 0 && checker == 0){ 81. printf("Попробуйте ще раз\n"); 82. } 83. } 84. return number; 85. } 86. void Computer\_Game\_record(struct Computer\_Game \* st,char \*string ,int count,int number) { 87. for (int i = 0; i < count;i ++) { 88. if (i == (number - 1)){ 89. sprintf(string,"%s %s %d %0.02f %s %s", 90. st[i].fullname, st[i].genre, st[i].year,st[i].rating, st[i].games.platform, st[i].games.numbers\_of\_player); 91. printf("%s",string); 92. } 93. } 94. } 95. void arrays\_of\_structers(struct Computer\_Game \* st,char \*text,int count) { 96. for (int i = 0;i < count; i ++) { 97. sscanf(text,"%s %s %d %f %s %s", 98. st[i].fullname, st[i].genre, &st[i].year,&st[i].rating, st[i].games.platform, st[i].games.numbers\_of\_player); 99. } 100. printf("Успішно записано"); 101. } 102. void text(struct Computer\_Game \* st,char \*text,int count) { 103. for (int i = 0;i < count; i ++) { 104. sprintf(text,"Назва: %s Жанр: %s Рік: %d Рейтинг: %f Платформа: %s Кількість: %s\n", 105. st[i].fullname, st[i].genre, st[i].year,st[i].rating, st[i].games.platform, st[i].games.numbers\_of\_player); 106. printf("Назва: %s Жанр: %s Рік: %d Рейтинг: %f Платформа: %s Кількість: %s\n", 107. st[i].fullname, st[i].genre, st[i].year,st[i].rating, st[i].games.platform, st[i].games.numbers\_of\_player); 108. } 109. } 110. void print\_ratingX(struct Computer\_Game \* st,int count,float number){ 111. int counter = 0; 112. for (int i = 0;i < count;i ++) { 113. if (st[i].rating <= number){ 114. count ++; 115. printf("Назва гри: %s\n",st[i].fullname); 116. } 117. } 118. if (counter == 0){ 119. printf("Не знайдено"); 120. } 121. } 122. void all\_structure\_ratingX(struct Computer\_Game \* st,int count,float number){ 123. int counter = 0; 124. for (int i = 0;i < count;i ++) { 125. if (st[i].rating <= number){ 126. printf("Назва гри: %s\n",st[i].fullname); 127. counter ++; 128. } 129. } 130. if (counter == 0){ 131. printf("Не знайдено"); 132. } 133. } 134. int valid\_removing(int count) { 135. int number; 136. int checker = 0; 137. int counter = 0; 138. while ( checker != 1) { 139. printf("Введіть номер структури,який ви хочете видалити: "); 140. number = getInt(); 141. if (number > count || number <= 0) { 142. counter ++; 143. } else { 144. checker = 1; 145. } 146. if (counter > 0 && checker == 0){ 147. printf("Попробуйте ще раз\n"); 148. } 149. } 150. return number; 151. } 152. void remove\_array(struct Computer\_Game \*st,int count,int number){ 153. strcpy(st[number-1].fullname, "") ; 154. strcpy(st[number-1].genre, "") ; 155. st[number-1].year = 0; 156. st[number-1].rating = 0; 157. strcpy(st[number-1].games.platform, "") ; 158. strcpy(st[number-1].games.numbers\_of\_player, "") ; 159. } 160. int valid\_recording1(int count){ 161. int checker = 0; 162. int counter = 0; 163. int number1; 164. while ( checker != 1) { 165. printf("Введіть номер структури для запису даних: "); 166. number1 = getInt(); 167. if (number1 > count || number1 <= 0) { 168. counter ++; 169. } else { 170. checker = 1; 171. } 172. if (counter > 0 && checker == 0){ 173. printf("Попробуйте ще раз\n"); 174. } 175. } 176. return number1; 177. } 178. int valid\_recording2(int count){ 179. int checker = 0; 180. int counter = 0; 181. int number2; 182. while ( checker != 1) { 183. printf("Введіть номер структури з якої ви хочете перезаписати дані: "); 184. number2 = getInt(); 185. if (number2 > count || number2 <= 0) { 186. counter ++; 187. } else { 188. checker = 1; 189. } 190. if (counter > 0 && checker == 0){ 191. printf("Попробуйте ще раз\n"); 192. } 193. } 194. return number2; 195. } 196. void rerecording(struct Computer\_Game \*st,int count,int number1,int number2){ 197. strcpy(st[number1-1].fullname,st[number2-1].fullname ) ; 198. strcpy(st[number1-1].genre, st[number2-1].genre) ; 199. st[number1-1].year = st[number2-1].year; 200. st[number1-1].rating = st[number2-1].rating; 201. strcpy(st[number1-1].games.platform, st[number2-1].games.platform) ; 202. strcpy(st[number1-1].games.numbers\_of\_player, st[number2-1].games.numbers\_of\_player) ; 203. } 204. void removing (struct Computer\_Game \*st,int count,int number) { 205. char down; 206. // int box\_width = 87; 207. printf("1.Видалити дані з поля /назва/\n"); 208. printf("2.Видалити дані з поля /жанр/ \n"); 209. printf("3.Видалити дані з поля /рік/\n"); 210. printf("4.Видалити дані з поля /рейтинг/\n"); 211. printf("5.Видалити дані з поля /платформа/\n"); 212. printf("6.Видалити дані з поля /кількість гравців/\n"); 213. printf("Ваш вибір: \n"); 214. down = conGetChar(); 215. switch(down) { 216. case '1':{ strcpy(st[number-1].fullname, "");} break; 217. case '2':{ strcpy(st[number-1].genre, "");}break; 218. case '3':{ st[number-1].year = 0;}break; 219. case '4':{ st[number-1].rating = 0;}break; 220. case '5':{ strcpy(st[number-1].games.platform, ""); }break; 221. case '6':{ strcpy(st[number-1].games.numbers\_of\_player, "");}break; 222. } 223. } 224. void rerecording\_1 (struct Computer\_Game \*st,int count,int number) { 225. char kyDown; 226. int maximum\_counter = 100; 227. //int box\_width = 87; 228. char buffer[BUFFER\_SIZE]; 229. for ( int i= 0;i < count;i++ ){ 230. if (i == number - 1){ 231. printf("1.Перезаписати дані з поля /назва/\n"); 232. printf("2.Перезаписати дані з поля /жанр/ \n"); 233. printf("3.Перезаписати дані з поля /рік/\n"); 234. printf("4.Перезаписати дані з поля /рейтинг/\n"); 235. printf("5.Перезаписати дані з поля /платформа/\n"); 236. printf("6.Перезаписати дані з поля /кількість гравців/\n"); 237. printf("Ваш вибір: \n"); 238. kyDown = conGetChar(); 239. switch(kyDown) { 240. case '1':{ 241. fgets(st[i].fullname, maximum\_counter, stdin); 242. st[i].fullname[strlen(st[i].fullname) - 1] = '\0'; 243. } break; 244. case '2':{ 245. fgets(st[i].genre, maximum\_counter, stdin); 246. st[i].genre[strlen(st[i].genre) - 1] = '\0'; 247. }break; 248. case '3':{ 249. st[i].year = 0; 250. fgets(buffer,maximum\_counter,stdin); 251. sscanf(buffer,"%d",&(st + i)->year); 252. }break; 253. case '4':{ 254. st[i].rating = 0; 255. fgets(buffer,maximum\_counter,stdin); 256. sscanf(buffer,"%f",&(st + i)->rating); 257. }break; 258. case '5':{ 259. fgets(st[i].games.platform, maximum\_counter, stdin); 260. st[i].games.platform[strlen(st[i].games.platform) - 1] = '\0'; 261. }break; 262. case '6':{ 263. fgets(st[i].games.numbers\_of\_player, maximum\_counter, stdin); 264. st[i].games.numbers\_of\_player[strlen(st[i].games.numbers\_of\_player) - 1] = '\0'; 265. }break; 266. } 267. } 268. } 269. } 270. void print\_array(struct Computer\_Game \*st, int count){ 271. for(int i = 0;i < count; i++){ 272. printf("Назва: %s ",st[i].fullname); 273. printf(" Жанр: %s ",st[i].genre); 274. printf(" Рік: %d ",st[i].year); 275. printf(" Рейтинг: %0.02f ",st[i].rating); 276. printf(" Платформа: %s ",st[i].games.platform); 277. printf(" Кількість: %s\n",st[i].games.numbers\_of\_player); 278. } 279. } 280. int valid\_rerecord (int count){ 281. int number; 282. int checker = 0; 283. int counter = 0; 284. while ( checker != 1) { 285. printf("Введіть номер структури,яку ви хочете перезаписати: "); 286. number = getInt(); 287. if (number > count || number <= 0) { 288. counter ++; 289. } else { 290. checker = 1; 291. } 292. if (counter > 0 && checker == 0){ 293. printf("Попробуйте ще раз\n"); 294. } 295. } 296. return number; 297. } 298. void rerecording\_2(struct Computer\_Game \*st, int count,int number) { 299. int maximum\_counter = 100; 300. char buffer[BUFFER\_SIZE]; 301. for ( int i= 0;i < count;i++ ){ 302. if (i == number - 1){ 303. printf("Введіть назву гри: "); 304. fgets((st + i)->fullname,maximum\_counter,stdin); 305. (st + i)->fullname[strlen((st + i)->fullname) -1] = '\0'; 306. printf("Введіть жанр гри: "); 307. fgets((st + i)->genre,maximum\_counter,stdin); 308. (st + i)->genre[strlen((st + i)->genre) -1] = '\0'; 309. printf("Введіть рік створення гри: "); 310. fgets(buffer,maximum\_counter,stdin); 311. sscanf(buffer,"%d",&(st + i)->year); 312. printf("Введіть рейтинг гри : "); 313. fgets(buffer,maximum\_counter,stdin); 314. sscanf(buffer,"%f",&(st + i)->rating); 315. printf("Введіть ігрову платформу гри : "); 316. fgets((st + i)->games.platform,maximum\_counter,stdin); 317. (st + i)->games.platform[strlen((st + i)->games.platform) -1] = '\0'; 318. printf("Введіть кількість гравців гри : "); 319. fgets((st + i)->games.numbers\_of\_player,maximum\_counter,stdin); 320. (st + i)->games.numbers\_of\_player[strlen((st + i)→ >games.numbers\_of\_player) -1] = '\0'; 321. } 322. } 323. } |

**Завдання 3.**

|  |
| --- |
| Asserts.h |
| 1. #include "assert.h" 2. #include "math.h" 3. void test1(void); 4. void test2(void); 5. void test3(void); |

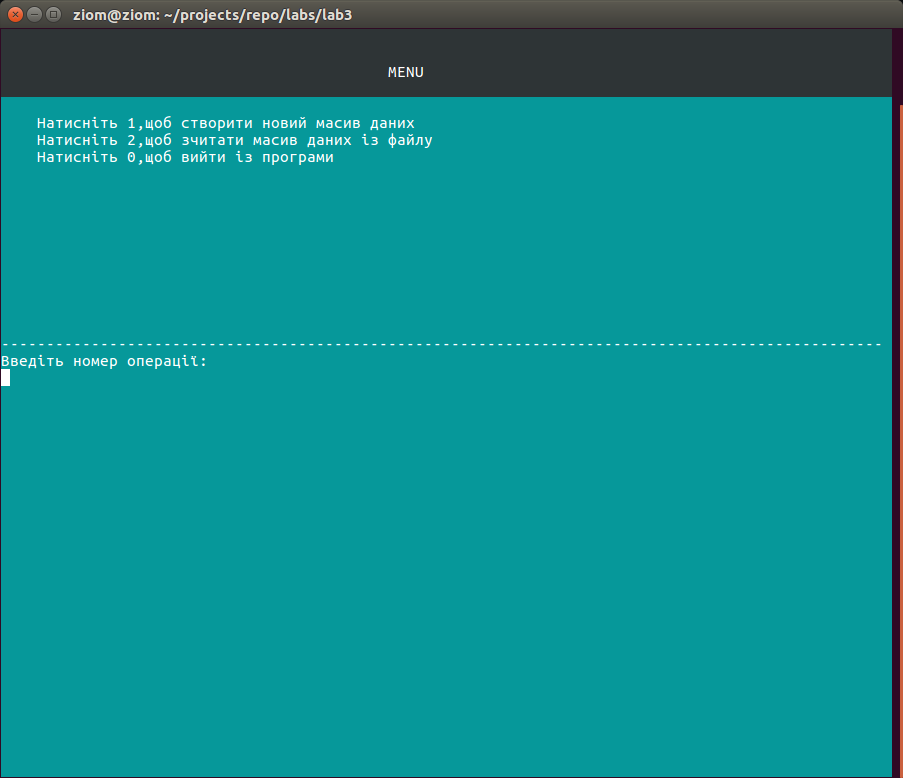
|  |
| --- |
| Asserts.c |
| 1. #include "assert\_.h" 2. #include "header.h" 3. void test1(void){ 4. struct Computer\_Game test[2] = { 5. {"Game1","dsfsd",1231,12.3,{"android","one"}}, 6. }; 7. rerecording(test,2,2,1); 8. assert(strcmp(test[0].fullname,test[1].fullname)== 0); 9. assert(strcmp(test[0].genre,test[1].genre)== 0); 10. assert(test[0].year == test[1].year); 11. assert(strcmp(test[0].games.platform,test[1].games.platform)== 0); 12. assert(strcmp(test[0].games.numbers\_of\_player,test[1].games.numbers\_of\_player)== 0); 13. } 14. void test2(void){ 15. struct Computer\_Game test[2] = { 16. {"Game2","honka",1998,24.3,{"MacOS","two"}}, 17. }; 18. remove\_array(test,2,1); 19. assert(strcmp(test[0].fullname,"")== 0); 20. assert(strcmp(test[0].genre,"")== 0); 21. assert(test[0].year == 0); 22. assert(strcmp(test[0].games.platform,"")== 0); 23. assert(strcmp(test[0].games.numbers\_of\_player,"")== 0); 24. } 25. void test3(void){ 26. char string1 []= "ghghj"; 27. char string2 []= "1232"; 28. char string3 []= "1game"; 29. char string4 []= "G2ame"; 30. char string5 []= "Need for Speed"; 31. assert(valid\_Computer\_Reverse\_record2 (string1) == 0); 32. assert(valid\_Computer\_Reverse\_record2 (string2) == 1); 33. assert(valid\_Computer\_Reverse\_record2 (string3) == 1); 34. assert(valid\_Computer\_Reverse\_record2 (string4) == 1); 35. assert(valid\_Computer\_Reverse\_record2 (string5) == 0); 36. } |

**Завдання 4.**

|  |
| --- |
| Menu.c |
| 1. #include "header.h" 2. enum { BUFFER\_SIZE = 100 }; 3. void clearLine(char c,int y,int x,int num,enum conAttribute\_e color) { 4. int i,j; 5. conSetAttr(color); 6. for (j = 0; j < num; j ++) { 7. for (i = 0; i < x; i ++) { 8. conMove(y+j,1+i); 9. putchar(c); 10. } 11. } 12. } 13. void prebox(int x\_width\_name,int x\_height\_name,enum conAttribute\_e color) { 14. conSetAttr(color); 15. int x,y; 16. for (x = 0;x < x\_width\_name;x ++) { 17. for (y = 0; y < x\_height\_name; y ++) { 18. conMove(y, x); 19. putchar(' '); 20. } 21. } 22. } 23. void box(int box\_width,int box\_left,int box\_top,int box\_height,int y\_console\_height,enum conAttribute\_e color) { 24. conSetAttr(color); 25. int x,y; 26. for ( x = box\_left;x < box\_left + box\_width; x++) { 27. for (y = box\_top + box\_height ; y < box\_top + box\_height + y\_console\_height; y++) { 28. conMove(y, x); 29. putchar(' '); 30. } 31. } 32. } 33. void input\_data(struct Computer\_Game \*st,int count) { 34. int i; 35. int maximum\_counter = 100; 36. int box\_width = 87; 37. clearLine(' ',6,box\_width-1,4,BG\_CYAN); 38. for(i = 0; i < count; i++) { 39. conMove(6,1); 40. char buffer[100]; 41. printf("Введіть назву %d гри: ",i + 1); 42. fgets((st + i)->fullname,maximum\_counter,stdin); 43. (st + i)->fullname[strlen((st + i)->fullname) -1] = '\0'; 44. printf("Введіть жанр %d гри: ",i + 1); 45. fgets((st + i)->genre,maximum\_counter,stdin); 46. (st + i)->genre[strlen((st + i)->genre) -1] = '\0'; 47. printf("Введіть рік створення %d гри: ",i + 1); 48. fgets(buffer,maximum\_counter,stdin); 49. sscanf(buffer,"%d",&(st + i)->year); 50. printf("Введіть рейтинг %d гри : ",i + 1); 51. fgets(buffer,maximum\_counter,stdin); 52. sscanf(buffer,"%f",&(st + i)->rating); 53. printf("Введіть ігрову платформу %d гри : ",i + 1); 54. fgets((st + i)->games.platform,maximum\_counter,stdin); 55. (st + i)->games.platform[strlen((st + i)->games.platform) -1] = '\0'; 56. printf("Введіть кількість гравців %d гри : ",i + 1); 57. fgets((st + i)->games.numbers\_of\_player,maximum\_counter,stdin); 58. (st + i)->games.numbers\_of\_player[strlen((st + i)->games.numbers\_of\_player) -1] = '\0'; 59. clearLine(' ',6,box\_width-1,10,BG\_CYAN); 60. } 61. } 62. int menu (void) { 63. char buffer[BUFFER\_SIZE]; 64. char keyDown ; 65. int box\_left = 0; 66. int box\_top = 5; 67. int box\_width = 100; 68. int box\_height = 0; 69. int x\_width\_name = 100; 70. int x\_height\_name = 5; 71. int y\_console\_height = 40; 72. conClear(); 73. prebox(x\_width\_name,x\_height\_name,BG\_BLACK); 74. conMove(3,44); 75. printf("MENU\n"); 76. conReset(); 77. conMove(12,0); 78. box(box\_width, box\_left, box\_top,box\_height, y\_console\_height,BG\_CYAN); 79. struct Computer\_Game \*release; 80. release = (struct Computer\_Game\*)malloc(8\*sizeof(struct Computer\_Game)); 81. if (NULL == release) { 82. printf("Can't alloc memory"); 83. return EXIT\_FAILURE; 84. } 85. conMove(6,5); 86. printf("Натисніть 1,щоб створити новий масив даних\n"); 87. conMove(7,5); 88. printf("Натисніть 2,щоб зчитати масив даних із файлу\n"); 89. conMove(8,5); 90. printf("Натисніть 0,щоб вийти із програми \n"); 91. while ( keyDown != '0') { 92. conMove(19,1); 93. printf("--------------------------------------------------------------------------------------------------"); 94. conMove(20,1); 95. puts ("Введіть номер операції: "); 96. keyDown = conGetChar(); 97. switch (keyDown) { 98. case '1': { 99. clearLine(' ',5,box\_width-1,6,BG\_CYAN); 100. conMove(6,1); 101. int count; 102. int number6 = 5; 103. printf ("\tВи можете охарактеризувати не більше 5 ігор\n"); 104. printf ("\tВиберіть кількість ігор ,які ви бажаєте охарактеризувати: "); 105. count = getInt(); 106. if (count > number6 || count <= 0) { 107. printf("Попробуйте ще раз\n"); 108. count = getInt(); 109. } 110. input\_data(release,count); 111. clearLine(' ',5,box\_width-1,20,BG\_CYAN); 112. conMove(6,1); 113. prints(); 114. conMove(3,44); 115. printf("Task 1\n"); 116. char Down; 117. while(Down != '0') { 118. conMove(16,1); 119. printf("--------------------------------------------------------------------------------------------------"); 120. conMove(17,1); 121. puts ("Введіть номер операції: "); 122. Down = conGetChar(); 123. switch(Down){ 124. case '1':{ 125. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 126. conMove(19,1); 127. // valid\_removing(count); 128. int number = valid\_removing(count); 129. removing (release, count,number); 130. print\_array(release, count); 131. }break; 132. case '2':{ 133. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 134. conMove(19,1); 135. int number = valid\_rerecord(count); 136. rerecording\_2(release, count,number); 137. print\_array(release, count); 138. }break; 139. case '3':{ 140. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 141. conMove(19,1); 142. int number = valid\_rerecord(count); 143. rerecording\_1 (release, count,number); 144. print\_array(release, count); 145. }break; 146. case '4':{ 147. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 148. conMove(19,1); 149. int number = valid\_Game\_record( count); 150. Computer\_Game\_record(release,buffer ,count,number); 151. }break; 152. case '5':{ 153. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 154. conMove(19,1); 155. int checker = valid\_Computer\_Reverse\_record (buffer); 156. Computer\_Game\_REVERSE\_record(release, count ,buffer,checker); 157. printf("Успішно записано"); 158. }break; 159. case '6':{ 160. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 161. conMove(19,1); 162. text(release,buffer, count); 163. // arrays\_of\_structers(release,buffer, count); 164. }break; 165. case '7':{ 166. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 167. conMove(19,1); 168. float number1 = 0; 169. printf("Введіть рейтинг: "); 170. number1 = getDouble(); 171. all\_structure\_ratingX(release,count,number1); 172. // print\_ratingX(release,count,number1); 173. } break; 174. case '8': { 175. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 176. conMove(19,1); 177. int number1 = valid\_recording1(count); 178. int number2 = valid\_recording2(count); 179. rerecording(release, count,number1,number2); 180. print\_array(release, count); 181. }break; 182. case '9':{ 183. clearLine(' ',18,box\_width-1,30,BG\_CYAN); 184. conMove(19,1); 185. // valid\_removing(count); 186. int number = valid\_removing(count); 187. remove\_array(release,count,number); 188. print\_array(release, count); 189. }break; 190. } 191. } 192. clearLine(' ',5,box\_width-1,30,BG\_CYAN); 193. conMove(6,1); 194. printf("Не забудьте зберегти дані\n"); 195. printf("Введіть назву текстового файлу для збереження\n"); 196. print\_in\_text\_files( count,release); 197. clearLine(' ',5,box\_width-1,5,BG\_CYAN); 198. conMove(6,1); 199. printf("Натисніть 2 ,щоб зчитати массив з файлу\n"); 200. printf("Натисніть 0,щоб вийти з програми\n"); 201. } 202. break; 203. case '2': { 204. // int count = 0; 205. clearLine(' ',5,box\_width-1,10,BG\_CYAN); 206. conMove(5,1); 207. load (release); 208. print\_array(release, 3); 209. conMove(8,1); 210. printf("--------------------------------------------------------------------------------------------------"); 211. conMove(9,1); 212. prints(); 213. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 214. conMove(3,44); 215. printf("Task 2\n"); 216. char kDown; 217. while(kDown != '0') { 218. conMove(18,1); 219. printf("--------------------------------------------------------------------------------------------------"); 220. conMove(19,1); 221. puts ("Введіть номер операції: "); 222. kDown = conGetChar(); 223. switch(kDown){ 224. case '1':{ 225. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 226. conMove(21,1); 227. // valid\_removing(3); 228. int number = valid\_removing(3); 229. removing (release, 3,number); 230. print\_array(release, 3); 231. }break; 232. case '2':{ 233. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 234. conMove(21,1); 235. int number = valid\_rerecord(3); 236. rerecording\_2(release, 3,number); 237. print\_array(release, 3); 238. }break; 239. case '3':{ 240. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 241. conMove(21,1); 242. int number = valid\_rerecord(3); 243. rerecording\_1 (release, 3,number); 244. print\_array(release, 3); 245. }break; 246. case '4':{ 247. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 248. conMove(21,1); 249. int number = valid\_Game\_record( 3); 250. Computer\_Game\_record(release,buffer ,3, number); 251. }break; 252. case '5':{ 253. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 254. conMove(21,1); 255. int checker = valid\_Computer\_Reverse\_record (buffer); 256. Computer\_Game\_REVERSE\_record(release,3 ,buffer,checker); 257. }break; 258. case '6':{ 259. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 260. conMove(21,1); 261. text(release,buffer, 3); 262. }break; 263. case '7':{ 264. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 265. conMove(21,1); 266. int number; 267. printf("Введіть рейтинг: "); 268. number = getDouble(); 269. all\_structure\_ratingX(release,3,number); 270. // print\_ratingX(release,3,number); 271. } break; 272. case '8': { 273. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 274. conMove(21,1); 275. int number1 = valid\_recording1(3); 276. int number2 = valid\_recording2(3); 277. rerecording(release, 3,number1,number2); 278. print\_array(release, 3); 279. }break; 280. case '9':{ 281. clearLine(' ',19,box\_width-1,30,BG\_CYAN); 282. conMove(21,1); 283. // valid\_removing(3); 284. int number = valid\_removing(3); 285. remove\_array(release,3,number); 286. print\_array(release, 3); 287. }break; 288. } 289. } 290. clearLine(' ',5,box\_width-1,30,BG\_CYAN); 291. conMove(6,1); 292. printf("Не забудьте зберегти дані\n"); 293. printf("Введіть назву текстового файлу для збереження\n"); 294. print\_in\_text\_files( 3,release); 295. clearLine(' ',5,box\_width-1,5,BG\_CYAN); 296. conMove(6,1); 297. printf("Натисніть 2 ,щоб зчитати массив з файлу\n"); 298. printf("Натисніть 0,щоб вийти з програми\n"); 299. } 300. break; 301. } 302. } 303. box(box\_width, box\_left, 0, box\_height,y\_console\_height,BG\_CYAN); 304. conMove(22,40); 305. printf("Кінець\n"); 306. conMove(50,1); 307. free(release); 308. return 0; 309. } |

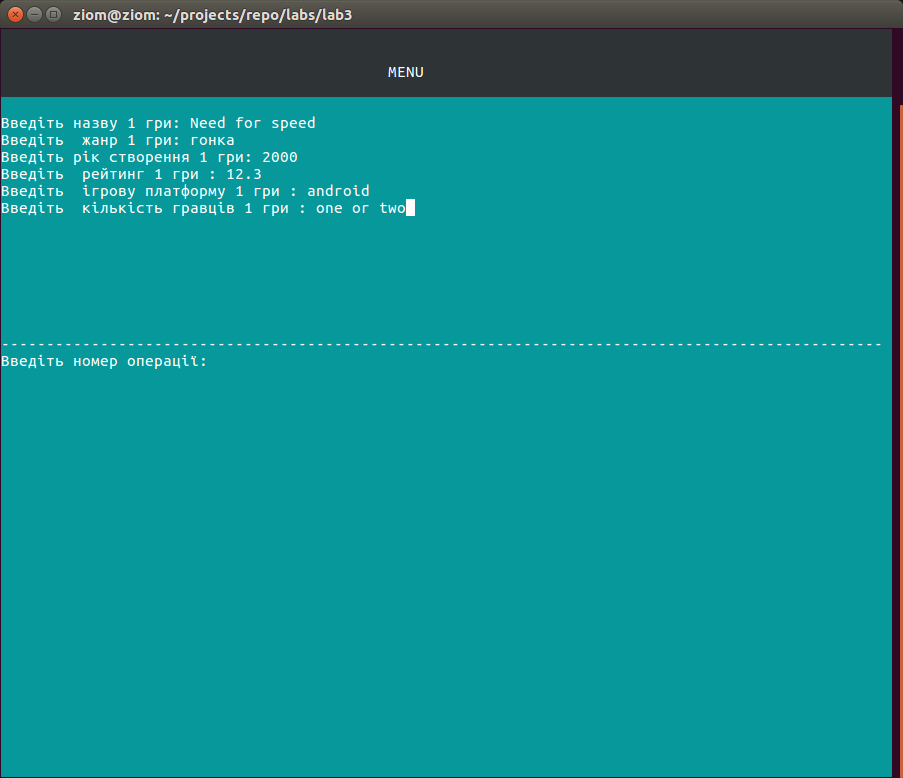
**Приклади результатів**

**Головне меню:**

****

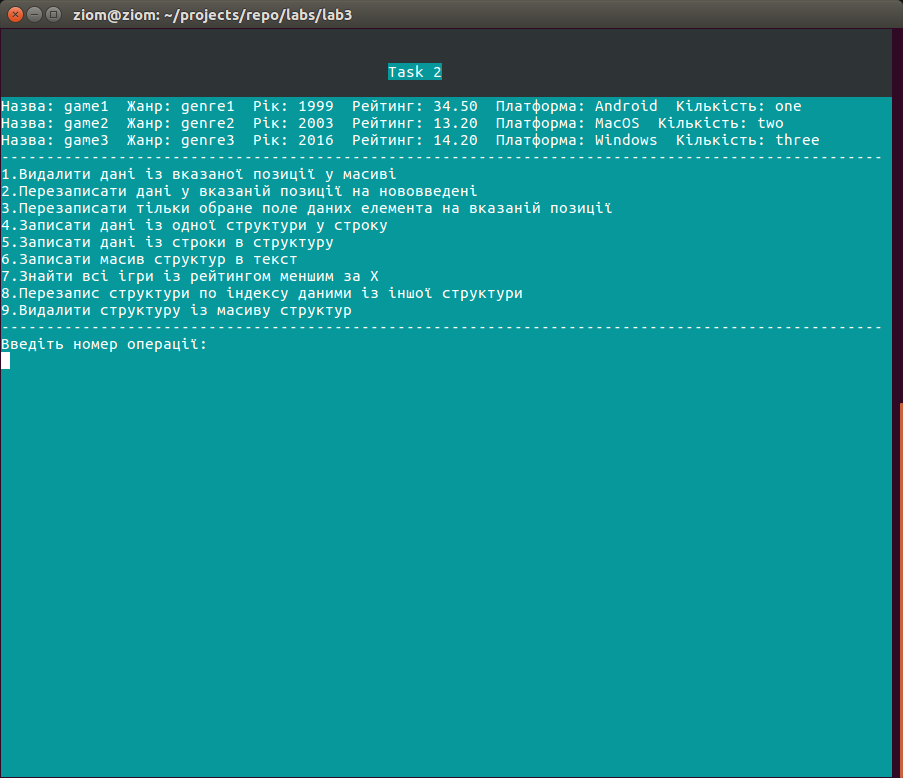
**1. Створення масиву структур:**

А) За умови корректного вводу.

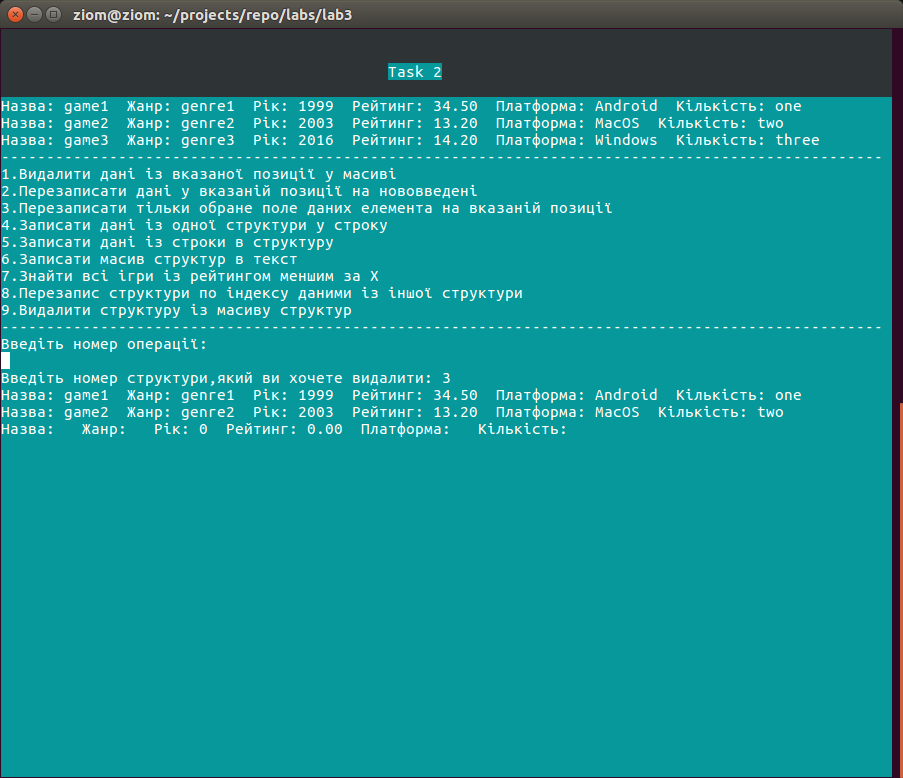


2. Робота із прочитаним з файлу масивом структур:

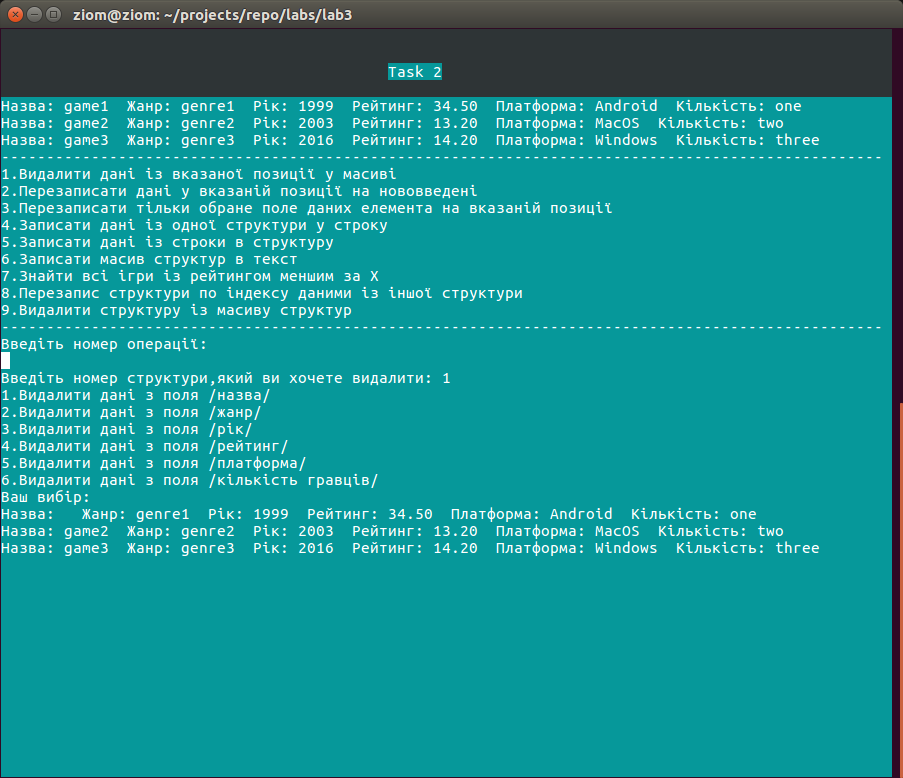
А) За умови коректного вводу.



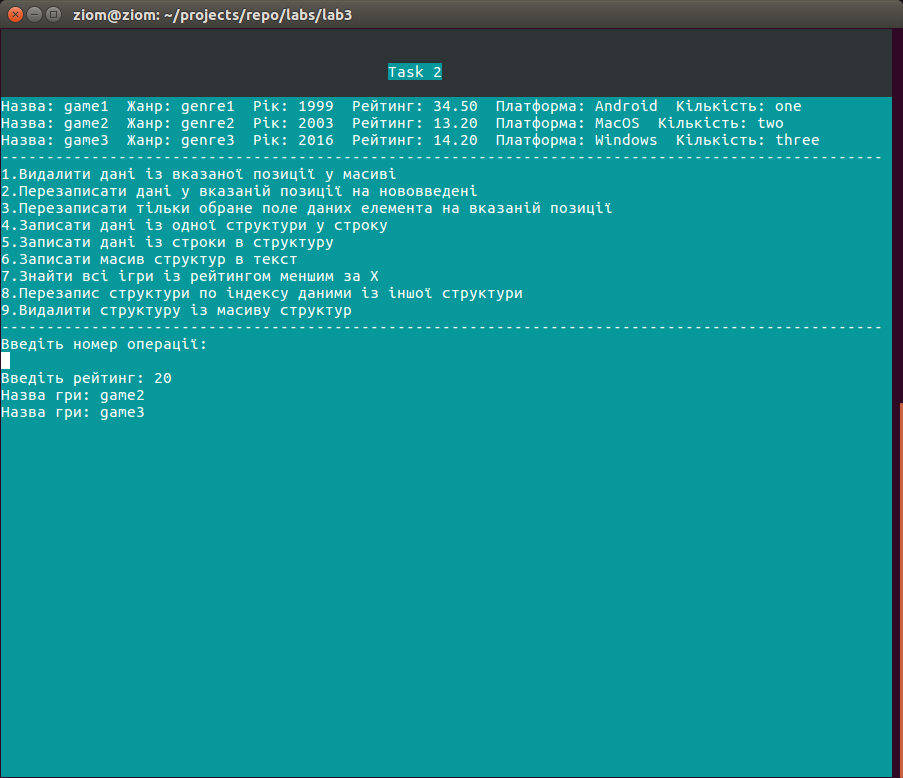
* + Видалення масиву за заданим індексом



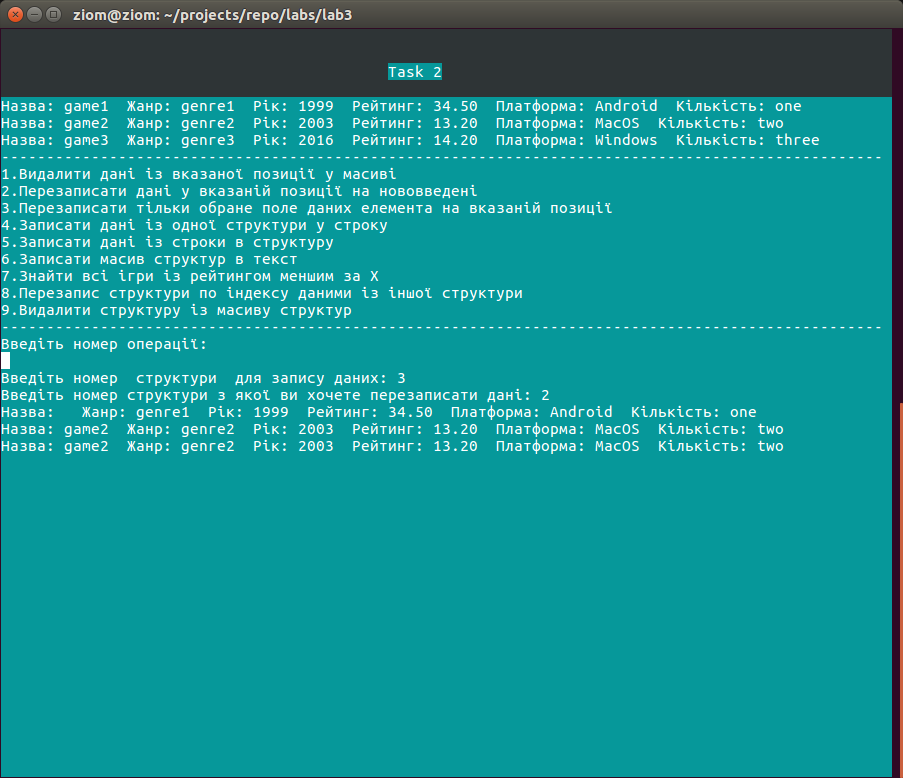
Видалити дані із вказаної позиції у масиві



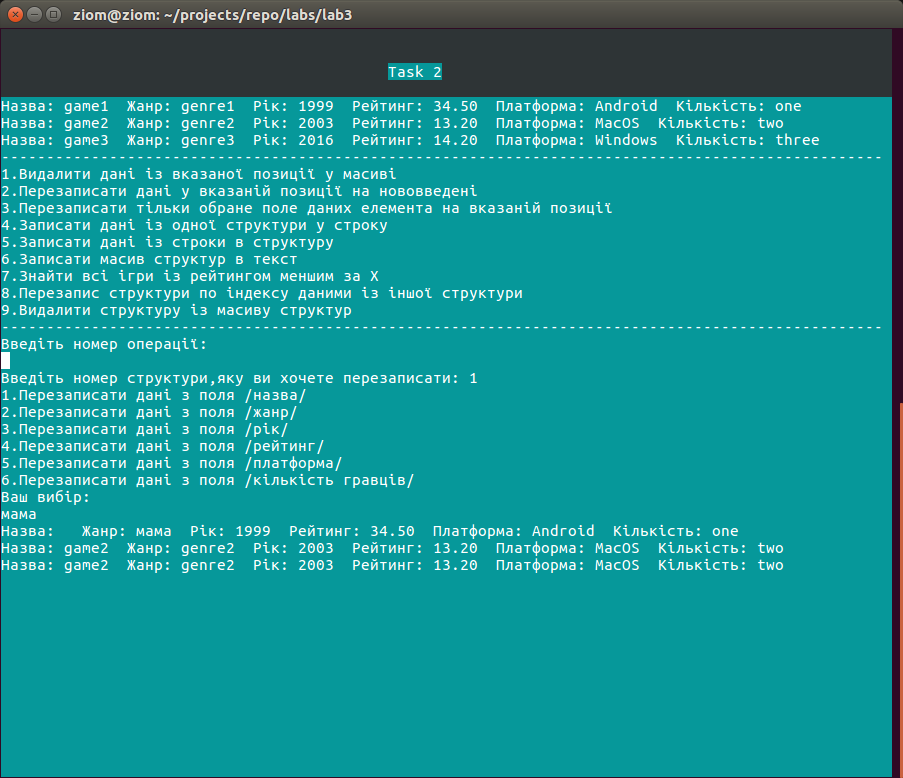
* + Знайти всі ігри з рейтингом меншим за Х



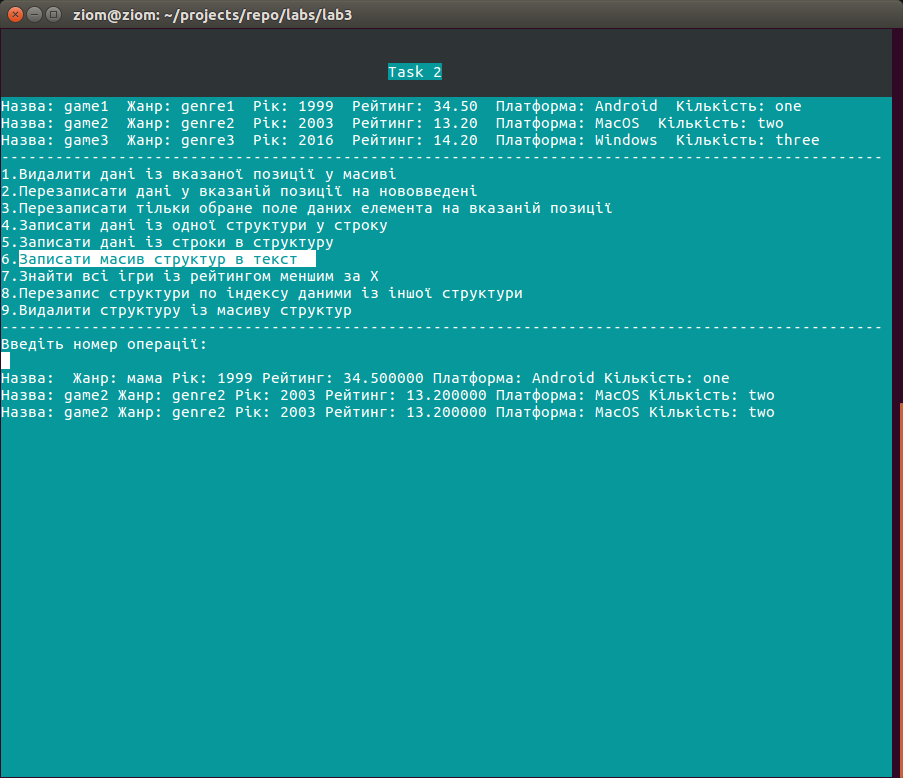
- Перезапис структури іншою структурою:



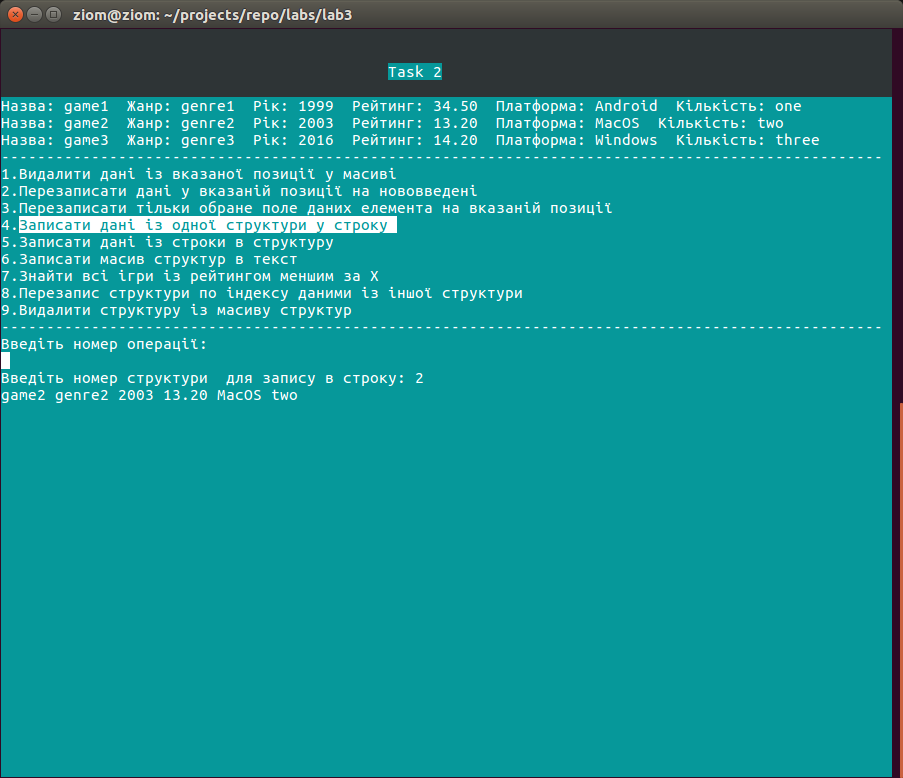
* + Перезапис поля структури масиву:



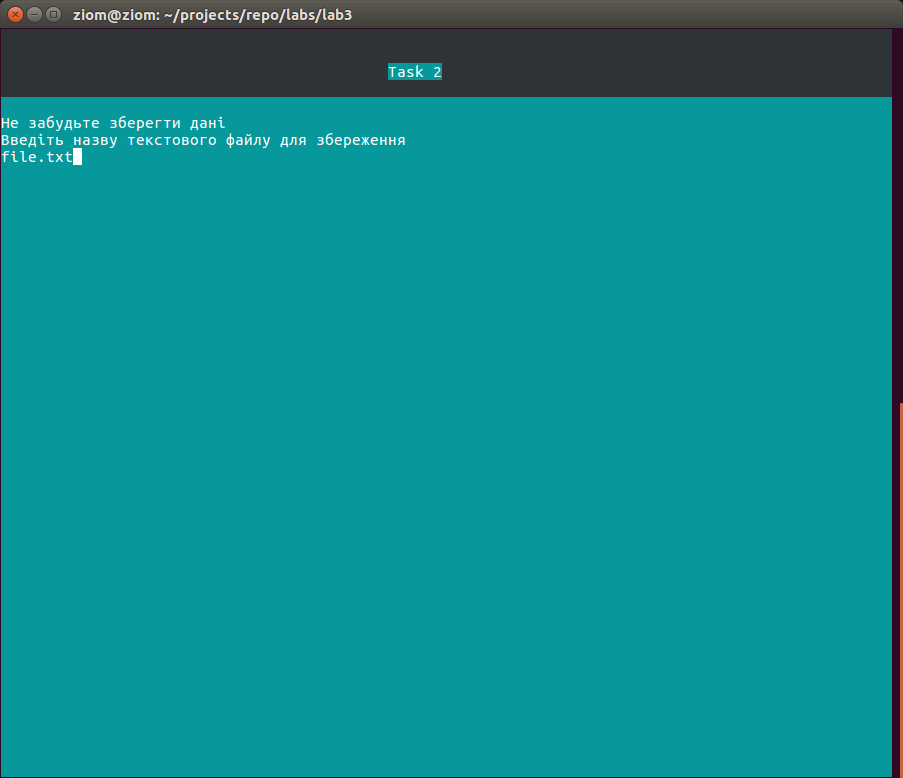
Записати масив структур в текст



Записати дані із одної структури у строку



* + Збереження масиву структур у файлі:



**Висновки**

Виконуючи дану лабораторну роботу, мною було застосовані на практиці знання, що стосувалися динамічного виділення пям’яті. Я навчилася працювати зі структурами та файлами, вдосконалив вміння роботи з вказівниками, та мною було вдосконалено власне вміння створення CUI, - вміння оформлювати консольну програму для зручності роботи користувача.

Компіляція всього коду відбувалася за допомогою компілятора gcc.