**报告名称:贪吃蛇**

**班级：国06**

**学号：2252075**

**姓名：刘文飞**

**完成日期：2023.6.24**

1. **设计思路与功能描述：**
2. **设计思路：本题目旨在完成一个贪吃蛇小游戏**
3. **功能描述：**
4. **入门版**：要求玩家控制方向键来控制小蛇的前进方向，以使蛇吃掉面板上随机位置上的食物（位置随机，数量为 1-5 随机）。每次成功吃掉食物后小蛇体长将增加一点，得分增加。食物吃光则再次随机产生。当小蛇撞到边界或者蛇头与蛇身相撞时，蛇将挂掉，游戏结束。
5. **进阶版**：蛇挂掉后，此时蛇尸身改变显示颜色变成边界，再随机产生新的食物和蛇，游戏继续。直到剩余空间不足以生成新的蛇和食物为止。
6. **高级版**：蛇挂掉后，此时蛇尸身改变显示颜色变成食物，再随机产生新的食物和蛇，游戏继续。直到撞墙次数>5，或剩余空间不足以生成新的蛇和食物为止。
7. 游戏应自行实现菜单功能，要求菜单简洁美观，交互友好，功能明确。
8. 游戏需要有实时 UI，有实时信息显示
9. **遇到的问题和解决方法**
10. **整体的架构：**

**（1）：要有面向对象编程的思想，将复杂的游戏各种项目分成一个个功能类实现，大致分为游戏的control类，snake类，wall类，UI类，food类等**

**（2）：类的相互关系，私有变量的访问权限要搞明白**

**（3）：二维贪吃蛇的核心在于map[][]字符数组，要先在 map数组中实现对应变化再来实现图形化**

1. **蛇移动的处理：**
2. 一开始写snake类时，只有头尾坐标，在出现蛇身层叠时无法判断是如何重叠的，在此我采用了动态双链表来存储每一段蛇身坐标以及前后指针，确定了蛇身蛇头蛇尾的位置关系
3. **继续学习使用easyx工具：**

**（1）：小黑子主题要依托easyx工具来实现**

1. **心得体会**
2. **贪吃蛇大作业不同于之前写的大作业，也是第一次在基本学习类与对象后用oop的思想进行编程，在编程过程中很明显感觉到oop思想的优势，一个是分类非常清晰，明确知道哪一个类负责哪一部分功能，二是采用类与对象后，代码可复用性增强，可扩展性增强，尤其是当类的功能越单一，体现地越明显**
3. **这次的大作业完成度不是那么高，问题出在自身，自己一开始以为是周日十八周末截止，6.23日晚上舍友说大作业要截止了，才惊慌发现已经完成不了，于是想6.24白天写完再补交，后来陈老师在群里发公告，感觉完蛋了。感谢星云助教给的补交机会（跪谢--）。虽说老师助教一再提醒不要赶ddl，自己还是没有听进去，不过这次通宵高程后一定会长记性。**
4. **源代码**
5. #define \_CRT\_SECURE\_NO\_WARNINGS
6. #pragma once
7. #include <conio.h>
8. #include<iostream>
9. #include<string>
10. #include <graphics.h>
11. #include <stdlib.h>
12. #include <ctime>
13. using namespace std;
14. const int size\_x = 1100;
15. const int size\_y = 730;
16. const int height = 38;
17. const int width = 38;
18. const int game\_x1 = 375;
19. const int game\_x2 = 1095;
20. const int game\_y1 = 5;
21. const int game\_y2 = 725;
22. struct body {
23. int x;
24. int y;
25. struct body\* next;
26. struct body\* previous;
27. };
28. class Snake {
29. private:
30. int lenth;
31. int life;
32. int score;
33. char direction;
34. struct body tail;
35. struct body head;
36. public:
37. Snake() {
38. lenth = 2;
39. life = 1;
40. score = 0;
41. direction = 'r';
42. head.x = 17;
43. head.y = 17;
44. tail.x = 16;
45. tail.y = 17;
46. head.previous = NULL;
47. head.next = &tail;
48. tail.next = NULL;
49. tail.previous = &head;
50. }
51. void renew() {
52. lenth = 2;
53. life--;
54. direction = 'r';
55. head.x = 17;
56. head.y = 17;
57. tail.x = 16;
58. tail.y = 17;
59. head.previous = NULL;
60. head.next = &tail;
61. tail.next = NULL;
62. tail.previous = &head;
63. }
64. int getlife() {
65. return life;
66. }
67. void setlife(int x) {
68. life = x;
69. }
70. struct body\* create\_body(int a, int b) {
71. struct body\* p = new body;
72. p->x = a;
73. p->y = b;
74. p->next = NULL;
75. p->previous = NULL;
76. return p;
77. }
78. void create(char map[][38]) {
79. map[head.x][head.y] = 'h';
80. map[tail.x][tail.y] = 't';
81. }
82. void move(char map[][38],int &judge) {
83. //先读键
84. if (\_kbhit()) {//如果有按键按下，则\_kbhit()函数返回真
85. char ch = \_getch();//使用\_getch()获取按下的键值
86. if (ch == -32) {
87. char c = \_getch();
88. switch (c) {
89. case 72:
90. direction = 'u';
91. break;
92. case 75:
93. direction = 'l';
94. break;
95. case 77:
96. direction = 'r';
97. break;
98. case 80:
99. direction = 'd';
100. }
101. }
102. switch (ch) {
103. case 119:
104. direction = 'u';
105. break;
106. case 97:
107. direction = 'l';
108. break;
109. case 100:
110. direction = 'r';
111. break;
112. case 115:
113. direction = 'd';
114. }
115. }
116. //根据方向运动，分为四种情况
117. if (direction == 'u') {
118. //三种情况分别是无食物，有食物，撞墙
119. if (map[head.x][head.y - 1] == '0') {
120. map[tail.x][tail.y] = '0';
121. struct body\* t = &tail;
122. while (t->previous != NULL) {
123. t->x = t->previous->x;
124. t->y = t->previous->y;
125. t = t->previous;
126. }/\*逆向遍历链表，除了头节点，其他坐标以及更改完毕\*/
127. map[head.x][head.y] = 's';
128. head.y = head.y - 1;
129. map[head.x][head.y] = 'h';
130. map[tail.x][tail.y] = 't';
131. }
132. else if ((map[head.x][head.y - 1] == 'b') || (map[head.x][head.y - 1] == 'c') || (map[head.x][head.y - 1] == 'y')) {
133. lenth++;
134. if (map[head.x][head.y - 1] == 'b')
135. score++;
136. else if (map[head.x][head.y - 1] == 'c')
137. score = score + 2;
138. else if (map[head.x][head.y - 1] == 'y')
139. score = score + 3;
140. struct body\* p = create\_body(head.x, head.y);
141. p->previous = &head;
142. p->next = head.next;
143. map[head.x][head.y] = 's';
144. head.y = head.y - 1;
145. map[head.x][head.y] = 'h';
146. head.next->previous = p;
147. head.next = p;
148. //每次吃一个食物，在头后面插入一个节点，则只需要处理头和这一个节点的数据
149. }
150. else if ((map[head.x][head.y - 1] == 'w') || (map[head.x][head.y - 1] == 't') || (map[head.x][head.y - 1] == 's')) {
151. judge = 0;
152. }
153. }
154. else if (direction == 'd') {
155. if (map[head.x][head.y + 1] == '0') {
156. map[tail.x][tail.y] = '0';
157. struct body\* t = &tail;
158. while (t->previous != NULL) {
159. t->x = t->previous->x;
160. t->y = t->previous->y;
161. t = t->previous;
162. }/\*逆向遍历链表，除了头节点，其他坐标以及更改完毕\*/
163. map[head.x][head.y] = 's';
164. head.y = head.y + 1;
165. map[head.x][head.y] = 'h';
166. map[tail.x][tail.y] = 't';
167. }
168. else if ((map[head.x][head.y + 1] == 'b') || (map[head.x][head.y + 1] == 'c') || (map[head.x][head.y + 1] == 'y')) {
169. lenth++;
170. if (map[head.x][head.y + 1] == 'b')
171. score++;
172. else if (map[head.x][head.y + 1] == 'c')
173. score = score + 2;
174. else if (map[head.x][head.y + 1] == 'y')
175. score = score + 3;
176. struct body\* p = create\_body(head.x, head.y);
177. p->previous = &head;
178. p->next = head.next;
179. map[head.x][head.y] = 's';
180. head.y = head.y + 1;
181. map[head.x][head.y] = 'h';
182. head.next->previous = p;
183. head.next = p;
184. //每次吃一个食物，在头后面插入一个节点，则只需要处理头和这一个节点的数据
185. }
186. else if ((map[head.x][head.y + 1] == 'w') || (map[head.x][head.y + 1] == 't') || (map[head.x][head.y + 1] == 's')) {
187. judge = 0;
188. }
189. }
190. else if (direction == 'l') {
191. if (map[head.x - 1][head.y] == '0') {
192. map[tail.x][tail.y] = '0';
193. struct body\* t = &tail;
194. while (t->previous != NULL) {
195. t->x = t->previous->x;
196. t->y = t->previous->y;
197. t = t->previous;
198. }/\*逆向遍历链表，除了头节点，其他坐标以及更改完毕\*/
199. map[head.x][head.y] = 's';
200. head.x = head.x - 1;
201. map[head.x][head.y] = 'h';
202. map[tail.x][tail.y] = 't';
203. }
204. else if ((map[head.x - 1][head.y] == 'b') || (map[head.x - 1][head.y] == 'c') || (map[head.x - 1][head.y] == 'y')) {
205. lenth++;
206. if (map[head.x - 1][head.y] == 'b')
207. score++;
208. else if (map[head.x - 1][head.y] == 'c')
209. score = score + 2;
210. else if (map[head.x - 1][head.y] == 'y')
211. score = score + 3;
212. struct body\* p = create\_body(head.x, head.y);
213. p->previous = &head;
214. p->next = head.next;
215. map[head.x][head.y] = 's';
216. head.x = head.x - 1;
217. map[head.x][head.y] = 'h';
218. head.next->previous = p;
219. head.next = p;
220. //每次吃一个食物，在头后面插入一个节点，则只需要处理头和这一个节点的数据
221. }
222. else if ((map[head.x - 1][head.y] == 'w') || (map[head.x - 1][head.y] == 't') || (map[head.x - 1][head.y] == 's')) {
223. judge = 0;
224. }
225. }
226. else if (direction == 'r') {
227. if (map[head.x + 1][head.y] == '0') {
228. map[tail.x][tail.y] = '0';
229. struct body\* t = &tail;
230. while (t->previous != NULL) {
231. t->x = t->previous->x;
232. t->y = t->previous->y;
233. t = t->previous;
234. }/\*逆向遍历链表，除了头节点，其他坐标以及更改完毕\*/
235. map[head.x][head.y] = 's';
236. head.x = head.x + 1;
237. map[head.x][head.y] = 'h';
238. map[tail.x][tail.y] = 't';
239. }
240. else if ((map[head.x + 1][head.y] == 'b') || (map[head.x + 1][head.y] == 'c') || (map[head.x + 1][head.y] == 'y')) {
241. lenth++;
242. if (map[head.x + 1][head.y] == 'b')
243. score++;
244. else if (map[head.x + 1][head.y] == 'c')
245. score = score + 2;
246. else if (map[head.x + 1][head.y] == 'y')
247. score = score + 3;
248. struct body\* p = create\_body(head.x, head.y);
249. p->previous = &head;
250. p->next = head.next;
251. map[head.x][head.y] = 's';
252. head.x = head.x + 1;
253. map[head.x][head.y] = 'h';
254. head.next->previous = p;
255. head.next = p;
256. //每次吃一个食物，在头后面插入一个节点，则只需要处理头和这一个节点的数据
257. }
258. else if ((map[head.x + 1][head.y] == 'w') || (map[head.x + 1][head.y] == 't') || (map[head.x + 1][head.y] == 's')) {
259. judge = 0;
260. }
261. }
262. }
263. void display() {
264. setfillcolor(RGB(135, 206, 235));
265. solidrectangle(220,85,360,435);
266. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
267. settextstyle(35, 25, "楷书");//设置字体高度，宽度，字型
268. settextcolor(RGB(255, 97, 0));
269. char buf1[10];
270. sprintf\_s(buf1, "%d",lenth); // 将 int 数据转换为字符串
271. char buf2[10];
272. sprintf\_s(buf2, "%d", life); // 将 int 数据转换为字符串
273. char buf3[10];
274. sprintf\_s(buf3, "%d", score); // 将 int 数据转换为字符串
275. outtextxy(220, 225, buf2);
276. outtextxy(220, 85, buf1);
277. outtextxy(220, 155, buf3);
278. }//蛇的移动
279. };
280. class Wall {
281. private:
282. public:
283. void create(char map[][38]) {
284. for (int i = 0; i < 38; i++) {
285. map[0][i] = 'w';
286. map[37][i] = 'w';
287. map[i][0] = 'w';
288. map[i][37] = 'w';
289. }
290. }//创造墙
291. void renew(char map[][38]) {
292. for (int i = 0; i < 38; i++) {
293. for (int j = 0; j < 38; j++) {
294. if ((map[i][j] == 's') || (map[i][j] == 't') || (map[i][j] == 'h')) {
295. map[i][j] = 'w';
296. }
297. }
298. }
299. }
300. };
301. class Food {
302. private:
303. int amount ;
304. char version[3];
305. public:
306. Food() {
307. amount = 15;
308. version[0] = 'b';//篮球
309. version[1] = 'c';//只因
310. version[2] = 'y';//背带裤
311. }
312. void create(char map[][38]) {
313. int count = 0;
314. while (count < amount) {
315. int a = rand();
316. int b = rand();
317. int c = rand();
318. a = a % 38;
319. b = b % 38;
320. c = c % 3;
321. if (map[a][b] == '0') {
322. map[a][b] = version[c];
323. count++;
324. }
325. }
326. }
327. void renew(char map[][38]) {
328. int count = 0;
329. for (int i = 0; i < 38; i++) {
330. for (int j = 0; j < 38; j++) {
331. if ((map[i][j] == 'b') || (map[i][j] == 'c') || (map[i][j] == 'y'))
332. count++;
333. }
334. }
335. while (count < amount) {
336. int a = rand();
337. int b = rand();
338. int c = rand();
339. a = a % 38;
340. b = b % 38;
341. c = c % 3;
342. if (map[a][b] == '0') {
343. map[a][b] = version[c];
344. count++;
345. }
346. }
347. }//游戏进行中刷新
348. void renew\_(char map[][38]) {
349. for (int i = 0; i < 38; i++) {
350. for (int j = 0; j < 38; j++) {
351. if ((map[i][j] == 's') || (map[i][j] == 't') || (map[i][j] == 'h')) {
352. int c = rand();
353. c = c % 3;
354. map[i][j] = version[c];
355. }
356. }
357. }
358. }
359. };
360. class UI {
361. private:
362. clock\_t start, end;
363. int version;
364. public:
365. UI() {
366. start = 0;
367. end = 0;
368. version = 0;
369. }
370. void start\_() {
371. start = clock();
372. }
373. void time\_display() {
374. end = clock();
375. double duration;
376. duration = (double)(end - start) / CLOCKS\_PER\_SEC;
377. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
378. settextstyle(35, 25, "楷书");//设置字体高度，宽度，字型
379. settextcolor(RGB(255, 97, 0));
380. char buf1[10];
381. sprintf\_s(buf1, "%d", (int)duration); // 将 int 数据转换为字符串
382. strcat(buf1, "s");
383. outtextxy(220, 295, buf1);
384. }
385. int get\_version() {
386. return version;
387. }
388. void menu1();
389. void menu2();
390. void main\_page();
391. //void display();
392. //void renew();
393. };
394. class Control {
395. private:
396. Snake snake;
397. Food food;
398. Wall wall;
399. UI ui;
400. char map[height][width];
401. char map2[height][width];
402. public:
403. Control() {
404. for (int i = 0; i < 38; i++) {
405. for (int j = 0; j < 38; j++) {
406. map[i][j] = '0';
407. }
408. }// 初始化地图
409. wall.create(map);
410. snake.create(map);
411. food.create(map);
412. for (int i = 0; i < 38; i++) {
413. for (int j = 0; j < 38; j++) {
414. map2[i][j] = '1';
415. }
416. }// 初始化地图
417. }
418. void display() {
419. snake.display();
420. ui.time\_display();
421. }
422. void over() {
423. char arr1[] = "游戏结束";
424. settextstyle(90, 45, "隶书");//设置字体高度，宽度，字型
425. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
426. settextcolor(RED);//红字
427. outtextxy(350, 300, arr1);
428. /\*一些数据的处理\*/
429. }//结束
430. void run() {
431. ui.start\_();
432. int judge = 1;
433. int version = ui.get\_version();
434. pointing();
435. if (version == 1) {
436. while (1) {
437. Sleep(300);
438. snake.move(map,judge);
439. food.renew(map);
440. pointing();
441. display();
442. if (judge == 0) {
443. over();
444. Sleep(2000);
445. break;
446. }
447. }
448. }
449. else if (version == 2) {
450. snake.setlife(5);
451. while (1) {
452. Sleep(300);
453. snake.move(map, judge);
454. food.renew(map);
455. pointing();
456. display();
457. if (judge == 0) {
458. wall.renew(map);
459. snake.renew();
460. judge = 1;
461. if (snake.getlife()==0) {
462. over();
463. Sleep(2000);
464. break;
465. }
466. }
467. }
468. }
469. else if (version == 3) {
470. snake.setlife(5);
471. while (1) {
472. Sleep(300);
473. snake.move(map, judge);
474. food.renew(map);
475. pointing();
476. display();
477. if (judge == 0) {
478. food.renew\_(map);
479. snake.renew();
480. judge = 1;
481. if (snake.getlife() == 0) {
482. over();
483. Sleep(2000);
484. break;
485. }
486. }
487. }
488. }
489. }
490. void menu() {
491. initgraph(size\_x, size\_y);
492. ui.menu1();
493. }
494. void pointing() {
495. IMAGE img1;
496. IMAGE img2;
497. IMAGE img3;
498. loadimage(&img1, "卡通篮球.jpg", 20, 20);//相对路径
499. loadimage(&img2, "只因.jpg", 20, 20);//相对路径
500. loadimage(&img3, "背带裤.jpg", 20, 20);//相对路径
501. for (int i = 1; i < 37;i++) {
502. for (int j = 1; j < 37; j++) {
503. if (map[i][j] != map2[i][j]) {
504. if (map[i][j] == 'w') {
505. setfillcolor(BLUE);
506. solidrectangle(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, game\_x1 + (i - 1) \* 20 + 20, game\_y1 + (j - 1) \* 20 + 20);
507. }
508. else if (map[i][j] == 'b') {
509. putimage(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, &img1);
510. }
511. else if (map[i][j] == 'c') {
512. putimage(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, &img2);
513. }
514. else if (map[i][j] == 'y') {
515. putimage(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, &img3);
516. }
517. else if (map[i][j] == 't') {
518. setfillcolor(GREEN);
519. solidrectangle(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, game\_x1 + (i - 1) \* 20 + 20, game\_y1 + (j - 1) \* 20 + 20);
520. }
521. else if (map[i][j] == 'h') {
522. setfillcolor(RED);
523. solidrectangle(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, game\_x1 + (i - 1) \* 20 + 20, game\_y1 + (j - 1) \* 20 + 20);
524. }
525. else if (map[i][j] == 's') {
526. setfillcolor(GREEN);
527. solidrectangle(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, game\_x1 + (i - 1) \* 20 + 20, game\_y1 + (j - 1) \* 20 + 20);
528. }
529. else if (map[i][j] == '0') {
530. setfillcolor(RGB(135, 206, 235));
531. solidrectangle(game\_x1 + (i - 1) \* 20, game\_y1 + (j - 1) \* 20, game\_x1 + (i - 1) \* 20 + 20, game\_y1 + (j - 1) \* 20 + 20);
532. }
533. map2[i][j] = map[i][j];
534. }
535. }
536. }
537. }
538. };
539. void UI::menu1() {
540. IMAGE img1;
541. IMAGE img2;
542. loadimage(&img1, "剪刀手.jpg", size\_x, size\_y);//相对路径
543. loadimage(&img2, "卡通.jpg", 300, 300);//相对路径
544. putimage(0, 0, &img1);
545. putimage(750, 400, &img2);
546. settextstyle(90, 45, "隶书");//设置字体高度，宽度，字型
547. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
548. settextcolor(RED);//红字
549. //文字居中
550. char arr1[] = "只 因 贪 吃 蛇";
551. char arr2[] = "开始游戏";
552. char arr3[] = "历史记录";
553. int x1 = (size\_x - textwidth(arr1)) / 2;//textwidth求字符串所占像素长度
554. int y1 = (size\_y - textheight(arr1)) / 2;//textheight求字符串所占像素高度
555. int x2 = (size\_x - textwidth(arr2)) / 2;//textwidth求字符串所占像素长度
556. int y2 = (size\_y - textheight(arr2)) / 2;//textheight求字符串所占像素高度
557. int x3 = (size\_x - textwidth(arr3)) / 2;//textwidth求字符串所占像素长度
558. int y3 = (size\_y - textheight(arr3)) / 2;//textheight求字符串所占像素高度
559. outtextxy(x1, y1 - 150, arr1);
560. settextstyle(70, 35, "隶书");//设置字体高度，宽度，字型
561. settextcolor(BROWN);//红字
562. outtextxy(x2, y2 + 100, arr2);
563. outtextxy(x3, y3 + 250, arr3);
564. MOUSEMSG m;
565. int judge = 0;
566. while (1) {
567. m = GetMouseMsg();
568. if (m.uMsg == WM\_LBUTTONDOWN)
569. {
570. if ((m.x > 370) && (m.x < 650) && (m.y > 435) && (m.y < 485)) {
571. judge = 1;
572. break;
573. }
574. else if ((m.x > 370) && (m.x < 650) && (m.y > 685) && (m.y < 735)) {
575. judge = 2;
576. break;
577. }
578. }
579. }
580. if (judge == 1) {
581. menu2();
582. }
583. else if (judge == 2) {
584. //menu3();
585. }
586. }
587. void UI::menu2() {
588. IMAGE img1;
589. IMAGE img2;
590. IMAGE img3;
591. IMAGE img4;
592. loadimage(&img1, "跳.jpg", size\_x, size\_y);//相对路径
593. loadimage(&img2, "电击小子.jpg", 200, 200);//相对路径
594. loadimage(&img3, "皮卡丘.jpg", 200, 200);//相对路径
595. loadimage(&img4, "中分头.jpg", 200, 200);//相对路径
596. putimage(0, 0, &img1);
597. putimage(100, 50, &img2);
598. putimage(100, 275, &img3);
599. putimage(100, 500, &img4);
600. settextstyle(90, 45, "隶书");//设置字体高度，宽度，字型
601. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
602. settextcolor(RED);//红字
603. //文字居中
604. char arr1[] = "入 门 版";
605. char arr2[] = "进 阶 版";
606. char arr3[] = "高 级 版";
607. outtextxy(350, 100, arr1);
608. outtextxy(350, 325, arr2);
609. outtextxy(350, 550, arr3);
610. MOUSEMSG m;
611. while (1) {
612. m = GetMouseMsg();
613. if (m.uMsg == WM\_LBUTTONDOWN)
614. {
615. if ((m.x > 100) && (m.x < 820) && (m.y > 60) && (m.y < 260)) {
616. version = 1;
617. break;
618. }
619. else if ((m.x > 100) && (m.x < 820) && (m.y > 280) && (m.y < 480)) {
620. version = 2;
621. break;
622. }
623. else if ((m.x > 100) && (m.x < 820) && (m.y > 500) && (m.y < 700)) {
624. version = 3;
625. break;
626. }
627. }
628. }
629. main\_page();
630. }
631. void UI::main\_page() {
632. setbkcolor(RGB(135, 206, 235));
633. // 用背景色清空屏幕
634. cleardevice();
635. const int ui\_x1 = 5;
636. const int ui\_x2 = 370;
637. const int ui\_y1 = 5;
638. const int ui\_y2 = 440;
639. const int ui\_y3 = 445;
640. const int ui\_y4 = 725;
641. setfillcolor(BLUE);//设置填充颜色-黄色
642. solidrectangle(0, 0, size\_x, game\_y1);
643. solidrectangle(0, game\_y2, size\_x, size\_y);
644. solidrectangle(0, 0, ui\_x1, ui\_y4);
645. solidrectangle(ui\_x2, ui\_y1, game\_x1, ui\_y4);
646. solidrectangle(game\_x2, ui\_y1, size\_x, ui\_y4);
647. solidrectangle(ui\_x1, ui\_y2, ui\_x2, ui\_y3);
648. char arr1[] = "入门版";
649. char arr2[] = "进阶版";
650. char arr3[] = "高级版";
651. char arr4[] = "长度";
652. char arr5[] = "得分";
653. char arr6[] = "生命";
654. char arr7[] = "时间";
655. char arr8[] = "最高分";
656. char arr9[] = "食物说明";
657. char arr10[] = "篮球 +1";
658. char arr11[] = "只因 +2";
659. char arr12[] = "背带裤 +3";
660. setbkmode(TRANSPARENT);//设置字体背景透明，默认不透明
661. settextstyle(55, 40, "隶书");//设置字体高度，宽度，字型
662. settextcolor(RGB(255, 97, 0));
663. if (version == 1) {
664. outtextxy(20, 5, arr1);
665. }
666. else if (version == 2) {
667. outtextxy(20, 5, arr2);
668. }
669. else if (version == 3) {
670. outtextxy(20, 5, arr3);
671. }
672. settextstyle(35, 25, "楷书");//设置字体高度，宽度，字型
673. outtextxy(20, 85, arr4);
674. outtextxy(20, 155, arr5);
675. outtextxy(20, 225, arr6);
676. outtextxy(20, 295, arr7);
677. outtextxy(20, 365, arr8);
678. outtextxy(20, 460, arr9);
679. outtextxy(20, 530, arr10);
680. outtextxy(20, 600, arr11);
681. outtextxy(20, 670, arr12);
682. IMAGE img1;
683. IMAGE img2;
684. IMAGE img3;
685. loadimage(&img1, "卡通篮球.jpg", 40, 40);//相对路径
686. loadimage(&img2, "只因.jpg", 40, 40);//相对路径
687. loadimage(&img3, "背带裤.jpg", 40, 40);//相对路径
688. putimage(220, 530, &img1);
689. putimage(220, 600, &img2);
690. putimage(220, 670, &img3);
691. }

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

#include<string>

#include<stdio.h>

#include <graphics.h>

#include <stdlib.h>

#include"snake.h"

using namespace std;

int main() {

Control con;

con.menu();

con.run();

return 0;

1. }