```
2
 3
                  SNAKE XENZTA
 4
                  -----
 5
 6
         This is a basic snake game which is commonly
 7
         found in old nokia phones. This game has a
 8
         snake the player has to move with the arrow
keys. There will be food particles generated
 9
10
         on the screen from time to time. When the
11
         snake eats them, its length increases. The
12
         game ends when the snake bits its own body.
13
         The player's objective is to get the highest
14
         possible score by making the snake as long as
15
         possible.
16
17
         Features of this game:
18
         1. 3 difficulty levels: Easy, Medium, Hard
19
             Harder the difficulty level, faster will
             be the movement of the snake.
20
21
         2. The player can choose 3 different screen
22
            sizes: Small, Medium, Large
23
24
25
         Created by: Anirudh Panigrahi
26
                  XI-E
27
                  Remal Public School, Sector-3
28
                  Rohini, New Delhi
29
                  Roll no.-11534
30
31
32
33
     #include <iostream.h>
34
     #include <conio.h>
35
     #include <dos.h>
36
     #include <stdlib.h>
37
38
     #define ARR_X 200
39
     #define ARR_Y 200
                         //dimensions of pos_key matrix
40
41
     char pos key[ARR X][ARR Y]; //matrix corresponding to every position on the screen
42
43
     void keystroke();
                                   //function which receives and maps keystrokes and decides the corresponding
     movement
     void adjustxy(int&, int&); //helps in swapping screen sides when snake reaches one edge of screen
void movehead(int, int); //prints the head of the snake
44
45
     void movetail(char [ARR_X][ARR_Y]); //replaces the last element of the snake with " "
46
47
     void addfood();
                                  //adds food particles in the game
48
                                   //checks if snake has eaten food or not
     int checkfood();
49
     int checkdie();
                                   //checks if snake has bitten itself or not
50
                                   //terminates game if checkdie() is true
     void printgameover();
51
     char getdir();
                                   //converts arrow key input to alphabets
     void screen();
void frame();
52
                                   //sets all screen parameters at the beginning of the program
53
                                   //prints the borders of the screen
54
     int pause();
                                   //manages pause option
55
                                               //helps in option selection in the game menus with arrow keys
     int select(int , int , int [], int);
56
57
                          //maximum horizontal coordinate of screen
     int MAX X:
58
     int MIN_X;
                          //minimum horizontal coordinate of screen
59
     int MAX_Y;
                          //maximum vertical coordinate of screen
60
     int MIN Y;
                          //minimum vertical coordinate of screen
                          //MAX X for small screen
61
     int max xs;
                          //MAX_X for medium screen
//MAX_X for large screen
62
     int max_xm;
63
     int max_xl;
                          //MAX_X for small screen
64
     int max_ys;
                         //MAX_X for medium screen
65
     int max_ym;
                          //MAX_X for large screen
66
     int max_yl;
                              //head coordinates
67
     int x, y;
     int x1, y1;
                              //tail coordinates
68
69
                              //food coordinates
     int x2, y2;
70
     int snaklen;
                          //records length of the snake(to show score at end)
71
     int frame_width = 150; //sets default game speed
72
                              //records the no. of times the game is played
     int counter = 0;
73
74
     void main()
75
     {
76
         if(counter==0)
          //FUNCTIONS TO RUN ONLY AT THE START OF THE PROGRAM//
77
78
79
              randomize();
                              //to seed the random() function in addfood()
80
              screen();
```

1

```
81
      //////WELCOME SCREEN////////
82
 83
          int choice;
 84
          char choice2;
 85
          flag3:
86
          snaklen = 1;
87
          for(int l = 0; l < ARR_X; ++l)
88
89
              for(int m = 0; m < ARR_Y; ++m)
 90
              {
91
                   pos_{key[l][m] = 'k';}
                                                //to reset pos key at start of every new game
92
93
94
          clrscr();
                      frame();
 95
          _setcursortype(_NOCURSOR);
96
          //////MAIN MENU/
97
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 - 6);
98
          textcolor(YELLOW);
          cprintf("**************");
99
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 - 4);
100
          cprintf("****");
101
102
          textcolor(MAGENTA);
          cprintf("SNAKE XENZIA");
103
104
          textcolor(YELLOW);
          cprintf("****");
105
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 - 2);
106
          cprintf("*******
107
          textcolor(WHITE);
108
109
          gotoxy((MAX X - MIN X + 1)/2 - 10, (MAX Y - MIN Y + 1)/2);
          cout<<"1. Play";
110
111
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 + 1);
          cout<<"2. Controls"
112
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 + 2);
113
          cout<<"3. Options"
114
          gotoxy((MAX_X - MIN_X + 1)/2 - 10, (MAX_Y - MIN_Y + 1)/2 + 3);
115
116
          cout<<"4. Exit";
          ///////END OF MAIN MENU///////
117
118
          int pos_gap1[12] = \{1, 1, 1, 1, -1\};
119
          choice = select((MAX_X - MIN_X + 1)/2 - 11, (MAX_Y - MIN_Y + 1)/2, pos_gap1, 4);
120
          switch (choice)
121
          {
122
              case 1:
123
                   clrscr();
                               frame();
                   x = (MAX_X - MIN_X + 1)/2, y = (MAX_Y - MIN_Y + 1)/2;
124
125
                   addfood();
126
                   keystroke();
127
                   break;
128
              case 2:
129
                   clrscr();
                               frame();
                   gotoxy((MAX_X - MIN_X + 1)/2 - 3, (MAX_Y - MIN_Y + 1)/2 - 3);
130
                   textcolor(YELLOW);
131
132
                   cprintf("CONTROLS");
                   textcolor(WHITE);
133
                   gotoxy((MAX_X - MIN_X + 1)/2 - 15, (MAX_Y - MIN_Y + 1)/2 - 1);
134
135
                   cout<<"1. Esc - Pauses the game";</pre>
                   gotoxy((MAX_X - MIN_X + 1)/2 - 15, (MAX_Y - MIN_Y + 1)/2 + 1);
136
137
                   cout<<"2. Arrow keys to move the snake"
                   gotoxy((MAX_X - MIN_X + 1)/2 - 18, (MAX_Y - MIN_Y + 1)/2 + 3);
138
139
                   cout<<"Press any key to return to main menu...";</pre>
140
                   getch();
141
                   goto flag3;
142
              case 3:
143
                   flag4:
144
                   clrscr();
                               frame();
                   //////OPTIONS MENU//////
145
146
                   gotoxy((MAX_X - MIN_X + 1)/2 - 3, (MAX_Y - MIN_Y + 1)/2 - 6);
147
                   textcolor(YELLOW);
                   cprintf("OPTIONS");
148
149
                   textcolor(WHITE);
                   gotoxy((MAX_X - MIN_X + 1)/2 - 8, (MAX_Y - MIN_Y + 1)/2 - 4); cout<<"1. Set difficulty level: ";
150
151
152
                   gotoxy((MAX_X - MIN_X + 1)/2 - 7, (MAX_Y - MIN_Y + 1)/2 - 3);
153
                   cout<<"1.1 Easy
                   gotoxy((MAX_X - MIN_X + 1)/2 - 7, (MAX_Y - MIN_Y + 1)/2 - 2);
154
                   cout<<"1.2 Medium"
155
156
                   gotoxy((MAX_X - MIN_X + 1)/2 - 7, (MAX_Y - MIN_Y + 1)/2 - 1);
157
                   cout<<"1.3 Hard"
                   gotoxy((MAX_X - MIN_X + 1)/2 - 8, (MAX_Y - MIN_Y + 1)/2 + 1);
158
159
                   cout<<"2. Set screen size"
                   gotoxy((MAX_X - MIN_X + 1)/2 - 7, (MAX_Y - MIN_Y + 1)/2 + 2);
160
161
                   cout<<"2.1 Small";</pre>
```

```
163
                  cout<<"2.2 Medium"
                  gotoxy((MAX X - MIN X + 1)/2 - 7, (MAX Y - MIN Y + 1)/2 + 4);
164
                  cout<<"2.3 Large";
165
166
                  gotoxy((MAX X - MIN X + 1)/2 - 8, (MAX Y - MIN Y + 1)/2 + 6);
                  cout<<"3. Back to main menu";
167
168
                  int pos_{gap2[12]} = \{1, 1, 3, 3, 3, 1, 1, 2, 2, 1, -1\};
                  choice = select((MAX_X - MIN_X + 1)/2 - 9, (MAX_Y - MIN_Y + 1)/2 - 3, pos_gap2, 10);
169
170
                  switch (choice)
171
172
                       case 1:
173
                           frame width = 150;
                           gotoxy((MAX X - MIN X + 1)/2 - 16, (MAX Y - MIN Y + 1)/2 + 9);
174
175
                           textcolor(GREEN);
176
                           cprintf("The difficulty level is now set to 'Easy'");
177
                           textcolor(WHITE);
178
                           getch();
                           goto flag4;
179
180
                       case 2:
181
                           frame width = 100;
                           gotoxy((MAX X - MIN X + 1)/2 - 16, (MAX Y - MIN Y + 1)/2 + 9);
182
183
                           textcolor(BLUE);
                           cprintf("The difficulty level is now set to 'Medium'");
184
185
                           textcolor(WHITE);
186
                           getch();
187
                           goto flag4;
188
                       case 3:
                           frame_width = 50;
189
190
                           gotoxy((MAX X - MIN X + 1)/2 - 16, (MAX Y - MIN Y + 1)/2 + 9);
191
                           textcolor(RED);
192
                           cprintf("The difficulty level is now set to 'Hard'");
                           textcolor(WHITE);
193
194
                           getch();
195
                           goto flag4;
196
197
                       case 6:
                           MAX X = max_xs;
198
199
                           MAX_Y = max_ys;
200
                                       frame();
                           clrscr():
                           gotoxy((MAX_X - MIN_X + 1)/2 - 16, (MAX_Y - MIN_Y + 1)/2 + 9);
201
                           textcolor(CYAN);
202
203
                           cprintf("The screen size is now set to 'Small'");
204
                           textcolor(WHITE);
205
                           getch():
206
                           goto flag4;
207
                       case 7:
208
                           MAX_X = max_xm;
                           MAX_Y = max_ym;
209
210
                           clrscr():
                                       frame();
211
                           gotoxy((MAX_X - MIN_X + 1)/2 - 16, (MAX_Y - MIN_Y + 1)/2 + 9);
                           textcolor(CYAN);
212
213
                           cprintf("The screen size is now set to 'Medium'");
214
                           textcolor(WHITE);
215
                           getch();
                           goto flag4;
216
217
                       case 8:
                           MAX_X = max_xl;
218
                           MAX_Y = max_yl;
219
220
                           clrscr();
                                       frame();
                           gotoxy((MAX_X - MIN_X + 1)/2 - 16, (MAX_Y - MIN_Y + 1)/2 + 9);
221
222
                           textcolor(CYAN):
223
                           cprintf("The screen size is now set to 'Large'");
224
                           textcolor(WHITE);
225
                           getch();
                           goto flag4;
226
227
                       case 10:
228
                           goto flag3;
229
230
                  break:
231
                  //////END OF OPTIONS MENU///////
232
              case 4:
233
                  clrscr();
                              frame();
                  gotoxy((MAX_X - MIN_X + 1)/2 - 15, (MAX_Y - MIN_Y + 1)/2 - 1);
234
235
                  textcolor(LIGHTRED);
236
                  cprintf("Are you sure you want to exit?");
237
                  gotoxy((MAX_X - MIN_X + 1)/2 - 15, (MAX_Y - MIN_Y + 1)/2 + 1);
238
                  cprintf("(hit key y or n...)");
239
                  textcolor(WHITE);
240
                  choice2 = getch();
                  if(choice2=='y' || choice2=='Y')
241
242
                  {
```

 $gotoxy((MAX_X - MIN_X + 1)/2 - 7, (MAX_Y - MIN_Y + 1)/2 + 3);$

162

```
243
                        exit(0);
                   }
244
245
                   else
246
                   {
247
                        goto flag3;
248
249
                   break;
250
251
          goto flag3;
252
         ////END OF WELCOME SCREEN////
        ////END OF MAIN////
253
254
255
      void keystroke()
256
257
      {
258
          flag:
          movehead(x, y);
259
                                    //to show initial direction of the snake
          char c0, c = getdir();
//TO SET INITIAL TAIL COORDINATES//
260
261
262
          switch(c)
263
264
               case 'w':
265
                   x1 = x;
266
                   y1 = y+1;
                   break;
267
268
               case
                   x1 = x;
269
                   y1 = y-1;
270
271
                   break;
272
               case
                     'a':
273
                   x1 = x+1;
                   y1 = y;
274
275
                   break;
276
               case
                    'd':
                   x1 = x-1;
277
278
                   y1 = y;
279
                   break;
280
               case 27 :
281
                   if(pause())
282
                   {
283
                        return; //gives the option of exiting the game before starting to play
284
                   }
285
                   else
286
                   {
287
                        goto flag;
288
289
               default :
                   goto flag;
290
291
292
          pos_{key}[x1][y1] = c;
293
          //LOOPS TO SET HEAD COORDINATES, MAP THE DIRECTION VALUE OF THE HEAD//
294
           //CORRESPONDING TO ITS POSITION AND CALL MOVEHEAD() AND MOVETAIL() //
295
           flag2:
296
               switch(c)
297
               {
                   case 'w':
298
299
                       do
300
                        {
301
                            adjustxy(x, y);
302
303
                            movehead(x, y);
304
                            y==MAX Y ? pos key[x][MIN Y] : pos key[x][y+1] = 'w';
                            movetail(pos_key);
305
306
                            printgameover();
307
                            delay(frame width);
308
                        }while(!kbhit());
309
                        break;
310
                   case 's':
311
                        do
312
                        {
313
                            adjustxy(x, y);
314
315
                            movehead(x, y);
316
                            y==MIN_Y ? pos_key[x][MAX_Y] : pos_key[x][y-1] = 's';
317
                            movetail(pos_key);
318
                            printgameover();
319
                            delay(frame_width);
                        }while(!kbhit());
320
321
                        break;
322
                   case
                          a':
323
                        do
```

```
324
                        {
325
                            X--:
                            adjustxy(x, y);
326
                            movehead(x, y);
327
                            x==MAX X ? pos key[MIN X][y] : pos key[x+1][y] = 'a';
328
                            movetail(pos_key);
329
330
                            printgameover();
331
                            delay(frame_width);
                        }while(!kbhit());
332
333
                        break;
334
                   case
                         'd':
335
                        do
336
                        {
337
                            X++;
338
                            adjustxy(x, y);
339
                            movehead(x, y);
340
                            x==MIN_X ? pos_key[MAX_X][y] : pos_key[x-1][y] = 'd';
341
                            movetail(pos_key);
342
                            printgameover();
343
                            delay(frame_width);
344
                        }while(!kbhit());
345
                        break;
346
               }
347
               c0 = getdir();
348
               //TO IGNORE OPPOSITE DIRECTION KEYSTROKE AND ANY OTHER KEYSTROKE//
               if(c0==27)
349
350
                   if(pause())
351
352
                    {
353
                        return;
354
355
                   else
356
357
                        goto flag2;
                   }
358
359
               }
360
               else
361
362
                   switch (c)
363
                    {
364
                        case 'w':
                            if (c0=='a' || c0=='d')
365
366
367
                                 c = c0;
368
                                 goto flag2;
369
                            }
370
                            else
371
                            {
                                 goto flag2;
372
373
                            }
374
                        case
                            if (c0=='a' || c0=='d')
375
376
377
                                 c = c0;
378
                                 goto flag2;
379
                            }
380
                            else
381
                            {
                                 goto flag2;
382
383
384
                        case
                            if (c0=='w' || c0=='s')
385
386
387
                                 c = c0;
388
                                 goto flag2;
389
390
                            else
391
392
                                 goto flag2;
393
                            }
                        case 'd':
394
395
                            if (c\theta == 'w' \mid | c\theta == 's')
396
397
                                 c = c0;
398
                                 goto flag2;
399
                            }
400
                            else
401
402
                                 goto flag2;
403
404
                   }
```

```
405
               }
406
407
408
      void adjustxy(int &x, int &y)
409
      {
          if (y == MAX_Y + 1)
410
411
              y = MIN_{\overline{Y}};
          else if (y == MIN_Y - 1)
412
               y = MAX_Y;
413
414
           if (x == MAX X + 1)
               x = MIN_X;
415
416
          else if (x == MIN_X - 1)
417
              x = MAX X;
          gotoxy(x, y);
cout<<"@";//<<x<<", "<<y;
418
419
                                        //for testing purposes
420
          return;
421
      }
422
      void movehead (int x, int y)
423
      {
          gotoxy(x, y);
424
425
          textcolor(WHITE);
426
          cprintf("@");
427
                                       //for testing purposes
          delay(500);
428
      }
429
430
      void movetail (char pos_key[ARR_X][ARR_Y])
431
      {
432
          if(checkfood()!=0)
433
           {
434
               ++snaklen;
435
               addfood();
436
               return;
437
          else if (checkfood()==0)
438
439
440
           //TO SET THE NEXT TAIL COORDINATES ACCORDING TO THE DIRECTION //
          //VALUE STORED IN POS KEY AT THE EXISTING POSITION OF THE TAIL//
441
442
               switch(pos_key[x1][y1])
443
               {
                   case 'w':
444
445
                       pos_key[x1][y1] = ' ';
446
                       y1--;
                       adjustxy(x1, y1);
447
448
                       gotoxy(x1, y1);
449
                        cout<<"
450
                        break;
451
                   case
                       pos_key[x1][y1] = ' ';
452
453
                       y1++;
454
                       adjustxy(x1, y1);
455
                       gotoxy(x1, y1);
456
                        cout<<"
457
                       break;
458
                   case
459
                       pos_key[x1][y1] = ' ';
460
                       x1--:
461
                       adjustxy(x1, y1);
462
                       gotoxy(x1, y1);
463
                        cout<<"
464
                       break;
465
                   case
466
                       pos key[x1][y1] = ' ';
467
                       x1++;
468
                       adjustxy(x1, y1);
469
                       gotoxy(x1, y1);
470
                        cout<<" ";
471
                       break;
472
473
               }
474
475
          }
476
      //
               delay(500);
                              //for testing purposes
477
               return;
478
479
480
481
      void addfood()
482
483
           //LOOP WILL RUN UNTIL X2, Y2 ARE SET TO VALUES WHICH ARE NOT ON THE SNAKE//
          do
484
485
           {
```

```
x2 = random(MAX_X - MIN_X + 1) + MIN X;
486
               y2 = random(MAX_Y - MIN_Y + 1) + MIN_Y;
487
488
           }while(x2==x && y2==y || (pos_key[x2][y2]=='w' || pos_key[x2][y2]=='s' ||
489
                          pos_key[x2][y2]=='a' || pos_key[x2][y2]=='d'
490
           gotoxy(x2, y2);
           textcolor(LIGHTMAGENTA);
491
492
           cprintf("@");
493
           textcolor(WHITE);
494
           return;
495
496
497
      int checkfood()
498
      {
           if(x==x2 && y==y2)
499
500
501
               return 1;
502
           }
503
           else
504
           {
505
               return 0;
506
           }
507
      int checkdie()
508
509
      ////IF POS KEY[X][Y] IS ALREADY MAPPED WITH A////
510
      ////DIRECTION VALUE WHEN THE HEAD REACHES (X, Y) COORDINATES////
511
           if(pos_key[x][y]=='w' || pos_key[x][y]=='s' ||
   pos_key[x][y]=='a' || pos_key[x][y]=='d'
512
513
514
515
               return 1;
516
           }
517
           else
518
           {
519
               return 0;
520
           }
521
522
      void printgameover()
523
524
           if(checkdie()!=0)
525
526
                    int k = 1;
                    delay(frame_width);
527
528
                    //LOOP TO FLASH "GAME OVER" 4 TIMES//
529
                    do
530
                    {
                        textcolor(RED);
531
532
                        gotoxy((MAX_X - MIN_X + 1)/2 - 5, (MAX_Y - MIN_Y + 1)/2);
533
                        cprintf("GAME OVER!!\a");
                        textcolor(WHITE);
534
                        delay(400);
535
                        clrscr();
536
                                     frame();
537
                        delay(600);
538
                        ++k;
539
                    }while(k<=4);</pre>
540
                    while(kbhit())
                                          //to ignore keystrokes pressed while displaying game over//
541
                    {
                                          //so that final score can be visible//
542
                        getch();
543
544
                    textcolor(RED);
545
                    gotoxy((MAX_X - MIN_X + 1)/2 - 5, (MAX_Y - MIN_Y + 1)/2);
                    cprintf("GAME OVER!!");
546
547
                    textcolor(WHITE);
                    gotoxy((MAX_X - MIN_X + 1)/2 - 8 , (MAX_Y - MIN_Y + 1)/2 + 1); cout<<"Final Score: "<<snaklen*10;
548
549
550
                    gotoxy((MAX_X - MIN_X + 1)/2 - 12, (MAX_Y - MIN_Y + 1)/2 + 2);
551
                    cout<<"Press any key to exit...";</pre>
552
                    getch();
553
                    ++counter:
554
                    main();
555
               }
556
557
      char getdir()
558
559
           char ch = getch();
           if(ch==0)
560
561
               ch = getch();
562
563
               switch(ch)
564
                    case 'H':
565
566
                        return 'w';
```

```
case 'P':
567
568
                        return 's';
569
                    case
                         return 'a';
570
571
                    case
572
                         return 'd';
573
                    default :
574
                        return 'x';
575
               }
576
577
           else if(ch==27)
578
579
                return ch;
580
           }
581
           else
582
           {
583
                return 'x';
                                 //any random character, so that it can be ignored by keystroke()
584
           }
585
586
      void screen()
587
588
           struct text_info info;
           gettextinfo(&info);
589
590
591
           MAX_X = (int) info.winright - 1;
           MIN_X = (int) info.winleft + 1;
592
           MAX^{T}Y = (int) info.winbottom - 2;
593
           MIN_Y = (int) info.wintop + 1;
594
595
           max_x = (int) MAX_x * 0.6;
           max\_xm = (int) MAX\_X * 0.8;
596
597
           max_xl = (int) MAX_X;
           max_ys = (int) MAX_Y * 0.6;
max_ym = (int) MAX_Y * 0.8;
598
599
           max_yl = (int) MAX_Y;
600
601
           frame();
602
           return;
603
      }
604
      void frame()
605
      {
606
607
           int width = MAX_X - MIN_X + 3;
           int height = MA\overline{X}_Y - MI\overline{N}_Y + 4;
608
609
           textcolor(YELLOW);
           gotoxy(1,1);
610
611
           for(int i = 0; i < width; i++)
612
                cprintf("%c", '*');
613
614
           }
615
616
           for(i = 2; i <= height - 2; i++)</pre>
617
                gotoxy(1, i); cprintf ("%c", '*');
618
                gotoxy(width, i); cprintf ("%c", '*');
619
           }
620
621
           gotoxy(1, height - 1);
622
623
           for(i = 0; i < width; i++)
624
                cprintf ("%c", '*');
625
626
           textcolor(WHITE);
627
628
629
      int pause()
630
      {
631
           textcolor(GREEN);
632
           gotoxy((MAX_X-MIN_X+1)/2 - 15, MIN_Y - 1);
           cprintf("PAUSED*Press esc again to exit");
gotoxy((MAX_X-MIN_X+1)/2 - 15, MAX_Y + 1);
633
634
635
           cprintf("Press any other key to resume");
636
           textcolor(WHITE);
637
           char ch = getch();
638
           frame();
           if(ch==27)
639
640
           {
641
                return 1;
642
           }
643
           else
644
           {
645
                return 0;
           }
646
647
      }
```

```
int select(int x_opt, int y_opt, int pos_gap[], int totalgap)
//X_OPT - X COORDINATE OF THE BULLET IN ALL POSITIONS//
648
649
650
      //Y OPT - Y COORDINATE OF BULLET FOR FIRST OPTION//
       //P\overline{O}S\_GAP[Y] - THE GAP B/W THE OPTION AT (Y+1) COORDINATE AND THE NEXT OPTION TO IT//
651
       //TOTALGAP - THE TOTAL LINES OCCUPIED BY ALL OPTIONS//
652
653
       {
654
           char ch = 26;
655
           int y_init = y_opt;
           int x_init = x_opt;
656
           gotoxy(x_init, y_opt);
657
           cprintf("%c", ch);
658
659
           do
660
           {
                //gotoxy(2, 2);cout<<x_init<<" "<<y_init<<" "
661
662
                //cout<<x_opt<<" "<<y_opt<<" "<<pos_gap[y_opt]-y_init]; //for testing purposes
663
                char c = getch();
664
                if(c==0)
665
                {
666
                    c = getch();
667
                    gotoxy(x_init, y_opt);
                                           //to delete initial bullet
                    cout<<'
668
                    cout<<' ';  //to delete initial bullet
//TO SET Y COORDINATE OF NEW BULLET ACCORDING TO ARROW KEY PRESSED//</pre>
669
                    switch(c)
670
671
                         case 'H' :
672
673
                             if(y_opt==y_init)
674
                                  y_opt = y_init + totalgap - 1;
675
676
                             }
677
                             else
678
                             {
679
                                  y_opt = y_opt - pos_gap[y_opt-y_init - 1];
680
681
                             break;
682
                         case
683
                             if(y_opt==(y_init + totalgap - 1))
684
685
                                  y_opt = y_init;
686
                             }
687
                             else
688
                             {
689
                                  y_opt = y_opt + pos_gap[y_opt-y_init];
690
691
                             break:
692
                    gotoxy(x_init, y_opt);
cprintf("%c", ch);
693
                                                    //to print new bullet
694
695
                else if(c==13)
696
697
698
                    return y_opt - y_init + 1; //to return option value according
699
                                       //to y coordinate of bullet
700
           }while(1);
701
702
      }
```