```
#include <graphics.h>
#include <iostream.h>
#include <conio.h>
#include <dos.h>
#include <stdlib.h>
#include <stdio.h>

//#pragma warn -wrch
#define MAX      50

#define UP_ARROW      72
#define DOWN_ARROW    80
#define LEFT_ARROW    75
#define RIGHT_ARROW   77

#define WinMinX       40
#define WinMaxX       600
#define WinMinY       40
#define WinMaxY       440

enum Direction {Forward,Backward,Upward,Downward};

struct Coord
{
int x , y;
};

class Snake;
class Point
{
int x , y , color ;
public:
Point ( )
{
set ();
}

void set();
void draw( );
int getx() {  return x; }
int gety() { return y;  }
friend int point_vanished ( Point &p , Snake &s );
};

class Snake
{
Coord *_Snake;
int _CurSize, _color,_MaxSize, _Points;
char _player;
Direction _Direction;
public:
Snake ( int size = 20, int color = RED , char player = 'M' )
{
_Snake = new Coord [ size ];
_CurSize = 3;
if ( player == 'C' )
{
_Snake [0].x = WinMaxX - 10;
_Direction = Backward;
}
else
{
_Snake [0].x = WinMinX + 10;
_Direction = Forward;
}
//_Snake [0].x = WinMinX + 10;
_Snake [0].y = WinMinY + 10;
_color = color;
_MaxSize = size;
```

```
  _player = player;
  _Points = 0;
  }
  void set( int size = 20, int color = RED , char player = 'M' )
  {
  delete _Snake;
  _Snake = new Coord [ size ];
  _CurSize = 3;
  if ( player == 'C' )
  {
  _Direction = Backward;
  _Snake [0].x = WinMaxX - 10;
  }
  else
  {
  _Snake [0].x = WinMinX + 10;
  _Direction = Forward;
  }
  _Snake [0].x = WinMinX + 10;
  //_Snake [0].y = WinMinY + 10;
  _color = color;
  _MaxSize = size;
  _player = player;
  _Points = 0;
  }

  void change_direction ( Direction d);
  void increment ();
  void inc_disp ();
  void shift_all ();
  void display ( int color = BLACK );
  void com_play ( Point p1 );
  friend int point_vanished ( Point &p , Snake &s );
  };
  //>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
  void Sound ( int s );
  void Message_Display ( char msg[30] , char color );
  void show_Header();
  void signature();
  int menu ();
  void drawMenu ( int selected , int defCol , int selCol );
  void show_About();
  void show_HowTOPlay();
  void show_New();
  void Play();
  //>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  int main()
  {

  int g = DETECT , d;
  initgraph ( &g , &d , "d:\tc\bgi" );

  int selected_option;

  Start:
  selected_option = menu();

  switch ( selected_option )
  {
  case 1:
  Play();
  goto Start;
  case 2:
  show_HowTOPlay();
  goto Start;
  case 3:
  show_New();
  goto Start;
```

```
case 4:
show_About();
goto Start;
case 5:
return 1;
}
return 1;
}


void Snake :: increment ( )
{
 //int i;

shift_all();
if ( _Direction == Forward )
{
if ( _Snake[0].x >= WinMaxX )
{
_Snake[0].x = WinMinX ;
}
else
_Snake[0].x = _Snake[0].x + 10;
}
else if ( _Direction == Backward  )
{
if ( _Snake[0].x <= WinMinX )
{
_Snake[0].x = WinMaxX ;
}
else
_Snake[0].x = _Snake[0].x - 10;
}
else if ( _Direction == Upward  )
{
if ( _Snake[0].y <= WinMinY )
{
_Snake[0].y = WinMaxY ;
}
else
_Snake[0].y = _Snake[0].y - 10;
}
else if ( _Direction == Downward  )
{
if ( _Snake[0].y >= WinMaxY )
{
_Snake[0].y = WinMinY ;
}
else
_Snake[0].y = _Snake[0].y + 10;
}
}


void Snake :: shift_all ()
{
int i;
for ( i = _CurSize -1 ; i > 0; i-- )
{
_Snake[i].x = _Snake[i-1].x;
_Snake[i].y = _Snake[i-1].y;
}
}

void Snake :: inc_disp ()
{
display ( BLACK );
increment();
display ( _color );
```

```
}


void Snake :: display ( int color)
{
setfillstyle ( 1, color );
if ( color == 0 )
{
setcolor ( 0 );
bar ( _Snake[_CurSize - 1].x - 5 , _Snake[_CurSize - 1].y - 5 , _Snake[_CurSize - 1].x + 5 , _Snake[_CurSize - 1].y + 5 );
rectangle ( _Snake[_CurSize - 1].x - 5 , _Snake[_CurSize - 1].y - 5 ,_Snake[_CurSize - 1].x + 5 , _Snake[_CurSize - 1].y + 5 );
 //return ;
}
else
{
setcolor ( WHITE );
for ( int i = 0; i< _CurSize; i++ )
{
bar ( _Snake[i].x - 5 , _Snake[i].y - 5 , _Snake[i].x + 5 , _Snake[i].y + 5 );
rectangle ( _Snake[i].x - 5 , _Snake[i].y - 5 , _Snake[i].x + 5 , _Snake[i].y + 5 );
}

/*
//int i = 0;
bar ( _Snake[i].x - 5 , _Snake[i].y - 5 , _Snake[i].x + 5 , _Snake[i].y+ 5 );
rectangle ( _Snake[i].x - 5 , _Snake[i].y - 5 , _Snake[i].x + 5 ,_Snake[i].y );
*/
setfillstyle ( 1 , 0 );
fillellipse ( _Snake[0].x  , _Snake[0].y  , 2  , 2);

char msg[50];
setcolor ( WHITE );

if ( _player == 'C' )
{
bar ( 250 , 12 , 630 , WinMinY - 10 );
sprintf ( msg , "Com Snake at :- ( %d , %d ) Score:- %d", _Snake[0].x, _Snake[0].y , _Points );
outtextxy ( 250 , 12 , msg );
}
else
{
bar ( 250 , 1 , 630 , WinMinY - 10 );
sprintf ( msg , "Ur Snake at :-  ( %d , %d ) Score:- %d", _Snake[0].x, _Snake[0].y , _Points );
outtextxy ( 250 , 1 , msg );
}
}
}


void Snake :: change_direction ( Direction d)
{
if ( ( _Direction == Forward ) && ( d == Backward ) )
{
Sound ( -1 );
}
else if ( ( _Direction == Backward ) && ( d == Forward ) )
{
Sound ( -1 );
}
else if ( ( _Direction == Upward ) && ( d == Downward ) )
{
Sound ( -1 );
}
else if ( ( _Direction == Downward ) && ( d == Upward ) )
{
```

```
Sound ( -1 );
}
else
{
_Direction = d;
Sound ( 1 );
}
}


void Point :: draw ( )
{
char msg[30];
setfillstyle ( 1 , color );
setcolor ( YELLOW );
bar ( x - 4 , y - 4 , x + 4 , y + 4 );
rectangle ( x - 4 , y - 4 , x + 4 , y + 4 );


setfillstyle ( 1 , 0 );
fillellipse ( x  , y  , 2 , 2 );

bar ( 1 , 1 , 300 , WinMinY - 10 );
sprintf ( msg , "Point at :- ( %d , %d )", x , y );
outtextxy ( 40 , 1 , msg );
}

void Point :: set ( )
{
color = random ( 15 ) + 1;
x = random ( ( ( WinMaxX - WinMinX  ) / 10 )  )   ;
y = random ( ( ( WinMaxY - WinMinY  ) / 10 )  )   ;
x = ( x * 10 ) + WinMinX;
y = ( y * 10 ) + WinMinY;
draw ( );
}

int point_vanished ( Point &p , Snake &s )
{
if ( ( s._Snake[0].x == p.x ) && ( s._Snake[0].y == p.y ) )
{
s._CurSize++;
if ( s._CurSize == s._MaxSize )
{
return 2;
}
s.increment ();
s.display ( RED );
Sound ( 2 );
delay ( 100 );

s._Points = s._Points + 20 ;
p.set();
return 1;
}
else
{
return -1;
}
}

void Sound ( int s )
{
if ( s == -1 )
{
sound ( 150 );
delay ( 30 );

sound ( 250 );
```

```
342   delay ( 30 );
343   nosound ();
344   }
345   else if ( s == 1 )
346   {
347   sound ( 450 );
348   delay ( 20 );
349   nosound ();
350   }
351   else if ( s == 2 )
352   {
353   sound ( 650 );
354   delay ( 20 );
355   nosound ();
356   }
357
358   }
359
360   void Snake :: com_play ( Point p1 )
361   {
362   if ( p1.getx() < _Snake[0].x )
363   {
364   if ( _Direction == Forward )
365   _Direction = p1.gety() < _Snake[0].y ? Upward : Downward;
366   else
367   _Direction = Backward;
368   }
369   else if ( p1.getx() > _Snake[0].x )
370   {
371   if ( _Direction == Backward )
372   _Direction = p1.gety() < _Snake[0].y ? Upward : Downward;
373   else
374   _Direction = Forward;
375   }
376   else
377   {
378   if ( p1.gety() < _Snake[0].y )
379   {
380   _Direction = Upward;
381   }
382   else if ( p1.gety() > _Snake[0].y )
383   {
384   _Direction = Downward;
385   }
386   }
387   }
388
389   void Message_Display ( char msg[30] , char color )
390   {
391   settextstyle ( 1 , 0 , 5 );
392   setcolor ( 8 );
393   outtextxy ( 195 , 205 , msg);
394
395   settextstyle ( 1 , 0 , 5 );
396   setcolor ( color );
397   outtextxy ( 200 , 200 , msg);
398   delay ( 1000 );
399   }
400
401   int menu ()
402   {
403   int ch;
404   int selected = 1;
405   int TotalOptions = 5;
406
407   cleardevice();
408   setbkcolor ( BLUE );
409   show_Header();
410   signature();
```

```
411
412   drawMenu ( selected , RED , GREEN );
413   do
414   {
415   ch = getch();
416   if ( ch == DOWN_ARROW )
417   {
418   selected = selected >= TotalOptions ? 1 : selected + 1;
419   drawMenu ( selected , RED , GREEN );
420   }
421   else if ( ch == UP_ARROW )
422   {
423   selected = selected < 2 ? TotalOptions : selected - 1;
424   drawMenu ( selected , RED , GREEN );
425   }
426
427   }while ( ch != '
428   ' );
429
430   return selected;
431   }
432
433   void drawMenu ( int selected , int defCol , int selCol )
434   {
435   int x = 250;
436   int y = 100;
437   int width = 150;
438   int height = 30;
439   int i;
440   int TotalOptions = 5;
441   char menu_option[5][14]= {
442   "    PLAY     ",
443   " HOW TO PLAY ",
444   " WHAT'S NEW  ",
445   "   ABOUT ME  ",
446   "     EXIT    "
447   };
448   setcolor ( WHITE );
449
450   for ( i = 1; i <= TotalOptions; i++ )
451   {
452   if ( i == selected )
453   setfillstyle ( 1 , selCol );
454   else
455   setfillstyle ( 1 , defCol );
456   bar ( x , y , x + width , y + height );
457   rectangle ( x , y , x + width , y + height );
458   outtextxy ( x + 20 , y + 10 , menu_option[i - 1] );
459   y = y + height + 30;
460   }
461   }
462
463   void show_About()
464   {
465   cleardevice();
466   setbkcolor ( BLACK );
467   show_Header();
468   setcolor ( WHITE );
469   settextstyle ( 0 , 0 , 0 );
470
471   signature();
472
473   getch();
474   }
475
476   void show_HowTOPlay()
477   {
478   cleardevice();
479   setbkcolor ( BLACK );
```

```
480   show_Header();
481   settextstyle ( 0 , 0 , 0 );
482   setcolor ( WHITE );
483   outtextxy ( 20 , 100 , "Objective:" );
484   outtextxy ( 20 , 150 , "Playing:" );
485   outtextxy ( 20 , 220 , "Tip:" );
486
487   setcolor ( LIGHTGREEN );
488   outtextxy ( 120 , 120 , "To collect 50 boxes before the computer Snake." );
489   outtextxy ( 120 , 170 , "1. Use arrow keys to control your Snake." );
490   outtextxy ( 120 , 180 , "2. To collect the box just come near to the BOX." );
491   outtextxy ( 120 , 190 , "3. Press <ESC> to QUIT any time." );
492   outtextxy ( 120 , 240 , "1. Use shortcuts to collect the BOX. [ Computer Snake never " );
493   outtextxy ( 120 , 250 , "   uses shortcut]" );
494   outtextxy ( 120 , 260 , "2. Computer Snake can't Hurt you, so enjoy moving around." );
495
496   signature();
497
498   getch();
499   }
500
501   void signature()
502   {
503   outtextxy ( 350 , 400 , "WWW " );
504   }
505
506   void show_Header()
507   {
508   setcolor ( RED );
509   settextstyle ( 1 , 0 , 4 );
510   outtextxy ( 193 , 27 , " SNAKE WAR - I " );
511   setcolor ( YELLOW );
512   outtextxy ( 195 , 25 , " SNAKE WAR - I " );
513   }
514
515   void show_New()
516   {
517   cleardevice();
518   setbkcolor ( BLACK );
519   show_Header();
520   settextstyle ( 0 , 0 , 0 );
521   setcolor ( WHITE );
522
523   outtextxy ( 20 , 100 , "What's new" );
524   outtextxy ( 20 , 150 , "What's next" );
525   outtextxy ( 20 , 260 , "When to expect next version" );
526   outtextxy ( 20 , 320 , "Comments, Bugs and Suggestions" );
527
528   setcolor ( LIGHTGREEN );
529   outtextxy ( 70 , 120 , "Nothing, cos it's the first version. :-)" );
530
531   outtextxy ( 70 , 170 , "In next version of this Game:- " );
532   outtextxy ( 90 , 180 , " > One or more player will be able to play."
533   outtextxy ( 90 , 190 , " > You'll be able to select Zero or more computer players." );
534   outtextxy ( 90 , 200 , " > You'll be able to PAUSE the Game any time." );
535   outtextxy ( 90 , 210 , " > You'll be able to select the color of each snake." );
536   outtextxy ( 90 , 220 , " > Keys will be customizable." );
537   outtextxy ( 90 , 230 , " > Snakes will be able to Hurt each other." );
538
539   outtextxy ( 70 , 280 , "Don't worry, i'll mail the code of next version too. [ Very
      Soon ]" );
540
541   outtextxy ( 70 , 340 , "For any suggestion or comment or Bug report feel free to mail
      me." );
542   outtextxy ( 70 , 350 , "There may be Bugs too in this game, so please let me know
      them." );
543
544   signature();
545   getch();
```

```
546   }
547
548   void Play()
549   {
550   Snake s1 ( MAX , GREEN , 'M' );
551   Snake s2 ( MAX , MAGENTA , 'C' );
552
553   char ch , KeyPressed = 0;
554
555   cleardevice();
556   randomize ();
557
558   rectangle ( WinMinX - 7, WinMinY - 7, WinMaxX + 7 , WinMaxY + 7 );
559   Point p1;
560
561   setbkcolor ( BLUE );
562   s1.inc_disp();
563   s2.inc_disp();
564
565   setcolor ( YELLOW );
566   outtextxy ( 10 , 450 , "> Collect 50 Boxes to WIN.   > Use shortcuts to WIN.");
567   setcolor ( CYAN );
568   outtextxy ( 10 , 460 , "> Use <ESC> to QUIT anytime. > LEFT , RIGHT , UP , DOWN Arrow
      Keys to Play. ");
569   getch();
570
571   KeyPressed = 1;
572   ch = 'R';
573   while ( 1 )
574   {
575   while ( !kbhit() )
576   {
577   s1.inc_disp();
578   if ( point_vanished ( p1 , s1 ) == 2 )
579   {
580   Message_Display ( "YOU WIN " , GREEN );
581   ch=0x1b;
582   getch();
583   break;
584   }
585
586   s2.com_play ( p1 );
587   s2.inc_disp();
588   if ( point_vanished ( p1 , s2 ) == 2 )
589   {
590   Message_Display ( "YOU LOSE " , GREEN );
591   ch=0x1b;
592   getch();
593   break;
594   }
595
596   delay ( 100 );
597   if ( KeyPressed == 1 )KeyPressed = 0;
598   }
599   if ( ch == 0x1b )
600   break;
601
602   ch = getch();
603   if ( KeyPressed == 1 )
604   {
605   KeyPressed = 0;
606   continue;
607   }
608   if ( ch == 0x1b )
609   break;
610   else if ( ch == 0 )
611   {
612   ch = getch ();
613   if ( ch == UP_ARROW )
```

```
{
s1.change_direction ( Upward );
KeyPressed = 1;
}
else if ( ch == DOWN_ARROW )
{
s1.change_direction ( Downward );
KeyPressed = 1;
}
else if ( ch == LEFT_ARROW )
{
s1.change_direction ( Backward );
KeyPressed = 1;
}
else if ( ch == RIGHT_ARROW )
{
s1.change_direction ( Forward );
KeyPressed = 1;
}
}
}
}

```