```
#include<graphics.h>
#include<windows.h>
#include<iostream>
//declaring the empty position
int emptyPosition=6;
//hit board array and its index
int hitBoard[4];
int hitBoardIndex;
//the hidden board which holds the main logic
int board[16]={1,13,5,3,14,9,0,15,6,11,12,10,8,2,4,7};
//coordinates of the boxes
int left[16]={240,400,560,720,240,400,560,720,240,400,560,720,240,400,560,720};
int top[16]={40,40,40,40,200,200,200,200,360,360,360,360,520,520,520,520};
int right[16]={340,500,660,820,340,500,660,820,340,500,660,820,340,500,660,820};
int bottom[16]={140,140,140,140,300,300,300,300,460,460,460,460,620,620,620,620};
class player
{
	public:
		char name[20];
		void getinput()
		{
			std::cout<<"Enter your first name\n";
			std::cin>>name;
		}
		void writeFile()
		{

		}
};
class numPuzzle
{
	private:
		 //height of the screen
		DWORD width,height;
	public:
		numPuzzle()
		{
			width=GetSystemMetrics(SM_CXSCREEN);
		height=GetSystemMetrics(SM_CYSCREEN);
		initwindow(width,height,"Number Puzzle Program");
		 }
```

```
        void drawBackground();
        void drawBoard();
        void drawTextBoard();
        void draw(player);
        void end();
        int checkWin();
        int mouseHitBox(int,int,int,int,int,int,int);
        void initialiseHitBoard();
        void start();
};
```
**//checking for the mouse to hit the boxes**
```
int numPuzzle:: mouseHitBox(int px,int py,int rx,int ry,int rw,int rh,int
userMouseHitPosition)
{
  if(px>=rx&&px<=rx+rw&&py>=ry&&py<=ry+rh)
  {
    return userMouseHitPosition;
  }
  return -1;
}
```
**//initialising the boxes that the user can hit**
```
//checking for boxes besides the empty box
void numPuzzle:: initialiseHitBoard()
{
   hitBoardIndex=0;
```
**   //Enabling the hitBoard**
**   //telling the function where to enable the hit option on the button**
**   //check for right position**

```
if(((emptyPosition+1)==4)||((emptyPosition+1)==8)||((emptyPosition+1)==12)||((emptyP
osition+1)==16))
   {}
   else
   {
     hitBoard[hitBoardIndex]=emptyPosition+1;
     hitBoardIndex++;
   }
   //check for left position

if(((emptyPosition-1)==-1)||((emptyPosition-1)==3)||((emptyPosition-1)==7)||((emptyPo
sition-1)==11))
```

```
  {}
  else
  {
    hitBoard[hitBoardIndex]=emptyPosition-1;
    hitBoardIndex++;
  }
  //check for top position
  if((emptyPosition==2)||(emptyPosition==0)||(emptyPosition==1)||(emptyPosition==3))
  {}
  else
  {
    hitBoard[hitBoardIndex]=emptyPosition-4;
    hitBoardIndex++;
  }
  //check for bottom position

if((emptyPosition==12)||(emptyPosition==13)||(emptyPosition==14)||(emptyPosition==1
5))
  {}
  else
  {
     hitBoard[hitBoardIndex]=emptyPosition+4;
  }
}
//check win logic
//check whenever the user clicks onto the boxes
int numPuzzle:: checkWin()
{
  int i;
  for(i=0;i<15;i++)
  {
     if(board[i]!=i+1)
       return 0;
  }
  return 1;
}
//if the player solves the game
//end the game
void numPuzzle:: end()
{
   int i;
```

```
   char t[6];
   cleardevice();
   setbkcolor(WHITE);
   setcolor(GREEN);
   settextstyle(SANS_SERIF_FONT,HORIZ_DIR,5);
   outtextxy((getmaxx()/2)-210,(getmaxy()/2)-100,"YOU WON THE GAME");
   outtextxy((getmaxx()/2)-240,(getmaxy()/2)-0,"PRESS ANY KEY TO EXIT");
   getch();
   for(i=0;i<(getmaxx()/2)-150;i++)
   {
     cleardevice();
     setcolor(BLUE);
     settextstyle(SANS_SERIF_FONT,HORIZ_DIR,5);
     outtextxy(i,(getmaxy()/2)-100,"THANK YOU");
     delay(1);
   }
   for(i=5;i>=1;i--)
   {
     cleardevice();
     sprintf(t,"%d",i);
     outtextxy((getmaxx()/2)-150,(getmaxy()/2)-100,"EXITING IN");
     outtextxy((getmaxx()/2)-70,(getmaxy()/2)-50,t);
     delay(1000);
   }
   exit(0);
}
//drawing the background
void numPuzzle::drawBackground()
{
        setcolor(BLUE);
    setfillstyle(SOLID_FILL,GREEN);
    rectangle(0,0,width,height);
    floodfill(2,2,GREEN);
}
//drawing the rectangle boxes
void numPuzzle::drawBoard()
{
  for(int i=0;i<16;i++)
  {
     setcolor(WHITE);
     rectangle(left[i],top[i],right[i],bottom[i]);
```

```
  }
}
//setting up the text in the board simultaneously as the board
void numPuzzle::drawTextBoard()
{
        int k=0,f;
   char a[10];
   settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
   for(int i=0;i<16;i++)
   {
        if(board[i]==0)
        {
         k++;
         continue;
        }
        sprintf(a,"%d",board[i]);
        setcolor(WHITE);
        outtextxy(left[k]+40,top[k]+40,a);
        k++;
   }
}
//main draw function
void numPuzzle:: draw(player p)
{
   //x=mouseXPosition  y=mouseYPosition
   //button=mouseClick
   //temp=swapping
   //win=checkWin()
   //hit=if there is a hit
   int i,x,y,button,hit,temp,win;
   char t[5];
        //drawing the initial things
   drawBackground();
   drawBoard();
   drawTextBoard();
   int page=0;
   //Run until the user presses any key
   while(!kbhit())
   {
        //double buffering - not working properly
       //setactivepage(page);
```

```
    //setvisualpage(1-page);
            outtextxy(1000,20,p.name);
             sprintf(t,"Mouse x position - %d",mousex());
            outtextxy(1000,50,t);
            sprintf(t,"Mouse y position - %d",mousey());
            outtextxy(1000,80,t);
            sprintf(t,"1.%d",hitBoard[0]);
            outtextxy(1000,110,t);
            sprintf(t,"2.%d",hitBoard[1]);
            outtextxy(1000,140,t);
            sprintf(t,"3.%d",hitBoard[2]);
            outtextxy(1000,170,t);
            sprintf(t,"4.%d",hitBoard[3]);
            outtextxy(1000,200,t);
         //if there is a mouse click from the user
        if(ismouseclick(WM_LBUTTONDOWN))
        {
             //initialise the hitBoard for the mouse
          initialiseHitBoard();
          //getting the mouseclick - although not using currently
          //a useless line
             getmouseclick(WM_LBUTTONDOWN,x,y);

             //looping through the hit board array, if there is a hit in one of the
possible allowed boxes
          for(i=0;i<=hitBoardIndex;i++)
          {
            //store the box reference if there is a hit in one of the boxes
            hit =
mouseHitBox(mousex(),mousey(),left[hitBoard[i]],top[hitBoard[i]],100,100,hitBoard[i]);
            //if there is a hit, swap it with the empty position
            if(hit!=-1)
            {
                  //swap the buttons
                  temp=board[emptyPosition];
                  board[emptyPosition]=board[hit];
                  board[hit]=temp;
                  emptyPosition=hit;
                  //checking if the player has won
                  win=checkWin();
                  if(win==1)
```

```
                    {
                       end();
                         }
                    //page is use in double buffering but not in use currently
                    //page=1-page;
                    //re-setup the things
                    cleardevice();
                    drawBackground();
                    drawTextBoard();
                    drawBoard();
                    setcolor(WHITE);
                 }
                }
            }//end of if there is a mouse click
     }//outer while loop
}
void numPuzzle:: start()
{
  int i,j;
  setcolor(WHITE);
  for(i=0;i<(width/2)-350;i++)
  {
     cleardevice();
     setcolor(i);
     rectangle(0,0,639,479);
     setcolor(WHITE);
     settextstyle(SANS_SERIF_FONT,HORIZ_DIR,8);
     outtextxy(i,(height/2)-140,"NUMBER PUZZLE");
   }
  setcolor(RED);
  settextstyle(SANS_SERIF_FONT,HORIZ_DIR,3);
  outtextxy((width/2)-200,height/2,"USE THE LEFT MOUSE TO CLICK");
  delay(2000);
  outtextxy((width/2)-350,(height/2)+40,"CIICK ON THE WINDOW AND PRESS
ENTER KEY TO START");
  getch();
}
int main()
{
        player p;
        p.getinput();
```

```
        numPuzzle num;
        num.start();
        num.draw(p);
        getch();
        cleardevice();
        closegraph();
        return 1;
}
```