2)

a) create your GDI drawing tool (CPen, CBrush, etc.)

b) declare a pointer to a GDI drawing tool (CPen, CBrush, etc.) named *pOld*

c) use *pDC->SelectObject* to use your created drawing tool. Set the pointer to the returned value.

1) When done drawing, use *pDC->SelectObject* to choose *pOld* as your tool.
- Keep a priority queue of nodes to search from, starting with only Monster's position.
- Grab node from queue, if player output the path
  - if already visited & pull off another node (or visited nodes)
  - for each neighbor node (of visited nodes) add
    - the neighbor to the priority queue using the
      cost to get to this neighbor plus the estimated
      cost to go from this neighbor to the player.

repeat until reach player.

4) PIXELFORMAT_DESCRIPTOR_pixelFormats;
int pixelFormat_index;
HGLRC openglRenderingContext;
pixelFormatIndex = ChoosePixelFormat(hdc, &pixelFormat);
SelectPixelFormat(hdc, pixelFormatIndex, &pixelFormat);
oglCreateContext(hdc_view);
openglRenderingContext = wglCreateContext(hdc_view);
oglMakeCurrent(hdc_view, openglRenderingContext);
oglBegin(GL_POINTS);
oglVertex2F(0.0, 0.0);
oglVertex2F(0.0, 1.0);
oglVertex2F(1.0, 1.0);
oglVertex2F(1.0, 0.0);
oglEnd();
glFinish();
SwapBuffers(hdc_view);
Override world shape

5. virtual int world_shape() {
   return cGame::SHAPE;
   return cGame::SHAPE_X_SCROLLER; 3
}
Create a CritterShow one child as buttons, which one child is for turned off state, one child is for turned on state.

Change critters to turned on state randomly, store the indexes of the critter in the order that they were turned on.

Use clistenerCursor to know which critter is clicked, compare with the array of indexes to check whether the order is correct.
Final Practice

- **Donkey Kong**
  
  Forces Needed:
  - Force Gravity - Applied to Mario to keep him on floor & for jumping
  - Applied to barrels too.
  - Force Object - For fire guys to chase down Mario

Listener needed:
- Listener Hopper - Allows you to run & jump.

Critters needed:
- Critter Walls
- Critter Prop Hammer
- Critter Prop Donkey Kong
- Critter Prop Girlfriend
- Critter Mario & The player!
- Critter Ladder
- Critter Barrel
- Critter Flaming Barrel
Color sniffing is a way to determine the color off certain sprites. The pop framework has the method:

```cpp
COLORREF CCritter::sniff(const Vector &sniffLocation, CPopView *pactiveview)
```

This tells you the color of the onscreen view corresponding to the given location.

In a race car game, we make the track one color and the wall and other off-road objects a different color.

Then, we can make the car stay on the track by doing the following:

In the critter's update code, you can have it look ahead and sniff to see if it is about to move off the track. If it is about to move off the track, we can either:

1. Look for a better direction to move it.
2. Make the car bounce off the wall.
A move box limits the movement of a critter. A critter is unable to move beyond the bounds of their move box.

A play box is used to limit where a critter can be dropped. A critter can be dropped beyond their move box if the play box bands are different.

Our group filmed the extreme pregnancy paradigm.

We only altered our project during our meetings.