# CPSC 427
# Project 5 MSVS

## YOU MAY WORK 2 TO A PROJECT

**Name: Balloons**

I have given you most of an implementation of a balloon popping game. You will finish the implementation and add a few extra bits. I have also given you a working executable to see how it should run. This executable has only been tested on Win7 64 bit.

**In Class Go Over**

The class diagram
The drawing canvas is (see controller constructor)
The drawing cycle is (see main)
It will run at different speeds on different systems, not framerate independent gaming loop
The console app is MS specific
SCARY_INNARDS – thread getchar with mutex protection, stay out of there for now
Explain how balloons are added to a vector and then erased if they are popped or fall off the screen.

**Constants:**

Map the keyboard inputs to whatever you want, mine may not be the same as yours

**Submission:**

When finished clean your project, delete the 'SQL Server Compact edition Database file' files and then zip up the entire thing and submit it.

**Requirements:**

**10 points**

Currently you have to close the application by killing the window. Make it so that when the user types an X or x it exits. Update the instructions object to reflect this.

**30 Points**

You are given an abstract base class Moveable
You are given 3 class headers Balloon. Person and Instructions
Instructions.cpp is implemented.
Please update Moveable as needed
Please implement Balloon.cpp and Person.cpp (graphics are given).
Note that both balloon and person move whereas instructions does not.

Please ensure that balloon is removed from the polymorphic vector holding it when it drops off the screen.

**15 points**

Derive a new class called Anvil that inherits from Moveable. It will behave much like the balloon class except it drops faster, does not hover before dropping, and earns either 5 points for Cosmo or the balloon depending on the outcome of `Controller::hasCollidedWithCosmo`. Please ensure that a balloon will appear 5 times more often than an anvil. Please ensure that the anvil is removed from the polymorphic vector holding it when it drops off the screen.

**15 points**

Derive a new class called TerribleBalloon that inherits from Balloon
If its popped for any reason it turns into a Bee and flys all over the screen. Its direction will be determined randomly (think rand()). Keep in mind that even though it flys in a random direction it should have gentle changes. Think some sort of rolling average for the movement.
This bee will have a limited time to live but should fly around for a bit before it disappears off screen and is removed from the polymorphic vector holding it. Make it so that when the user types T or t, the only kind of Balloon is a TerribleBalloon.

**30 points**

Use polymorphism to update all the moving objects other than Cosmo. That is one vector should hold Terribleballoons, balloons, and anvils. This means you must make the vector in the controller class polymorphic as well as all accesses to its members.

**Stuff to remember:**

Remember \ is a control char so '\\' is equal to \  or just 1 char.
So  "   \\\\\\\\|//// " is still only 13 chars wide.

**Questions:**

**I'm not sure what I need to do to get Cosmo to move. Is it in the Controller.cpp that I need to change where it says**
**"//render cosmo to screenbuffer**
**cosmo.draw(myScreenVector);"?**
**Or somewhere else?**

You call cosmo.draw and pass him the screen buffer to draw to.  Look in the base class (Moveable), there is something called myLoc which gives his current location.  You will do something like this for EVERY draw call (this is going LEFT incidently)  myLoc.x = myLoc.x - spd; where speed is how fast he moves, which is also defined in the base class.  There is also a myLoc.y for when you make him move up and down.  Also do not use absolute line numbers, make them relative to myloc like the following (also for cosmo going left)

```
            myScreenVector[myLoc.y  ].replace(myLoc.x,PERSON_ WIDTH,"   \\\\\\\\|//// ");
myScreenVector[myLoc.y+1]. replace(myLoc.x,PERSON_WIDTH,"    |~ ////  ");
myScreenVector[myLoc.y+2]. replace(myLoc.x,PERSON_WIDTH,"    |O  //   ");
```

**My bee looks terrible can I have yours?**

LEFT:

//the top 2 are if the wing is flapping or not, seems to need a bool to keep track of which wouldn't you say?

```
" ()) "     //flapping

" ()  "     //not flapping
"%000-"
"     "
```

RIGHT:

```
"  (()"

"  (()"
"-000%"
"     "
```

UP:

```
"  %  "
" =0= "
"  |  "
```

DOWN:

```
"  |  "
" =0= "
"  %  "
```

**Does the game pause or is Cosmo defenseless if I hit I (instructions) during the game?**
Its a single threaded game.  When the instructions are displayed the game is paused.

**Do anvils and bees have custom animations?**
Derived classes, anvil from movable, Terribleballloon from Balloon, the draw function for each is the only customization.

**Can the bee potentially move back towards Cosmo  immediately after spawning?**
Yep, mine is set to move up though and then moves in random directions.

**Where does the bee spawn in relation to the balloon that exploded?**
It uses the same location as the balloon(upper left corner) its just not as wide or tall.  You can look in constants for the size of its bounding box.

**How long does the bee live?**
The bee lives until any part of its bounding box touches edge of screen then it disapears (removed from vector).  No points awarded at all, my bee has collisions turned off.  Whatever you want to do.

**Is the bee class separate from the terrible balloon class?**
Terribleballoon derives from balloon, you use balloons draw method until its popped, then you use the terribleballoons draw method which draws a bee.

**What does the comment "must be defined by derived class" mean?**
Pure virtual function, derived classes must implement it

**Do you see the balloon pop/explode or does it immediately transform into a bee?**
However you would like to do it

**What can we change in controller?**
Whatever you want, you will have to mod the vector at least.  In fact you can change any part of the project.

**Do I have to make Cosmo’s eyes blink on the instruction page?**
No