Lab 9 – Spawnables

Task 1

We will begin by building the example from the lecture. Frist, update the Sprite class to have a Boolean variable to track if the sprite is alive, along with getters/setters.
Build the ball class shows in the lecture. A simple class that has some movement code within it.
Next, create the BallManager class discussed in the lecture. For ease the code is presented below:

BeachBallManager.h

```
#pragma once
#include "Ball.h"
#include <math.h>
#include <vector>

class BeachBallManager
{
public:
	BeachBallManager();
	~BeachBallManager();

	void spawn();
	void update(float dt);
	void deathCheck();
	void render(sf::RenderWindow* window);

private:
	std::vector<Ball> balls;
	sf::Vector2f spawnPoint;
	sf::Texture texture;
};
```

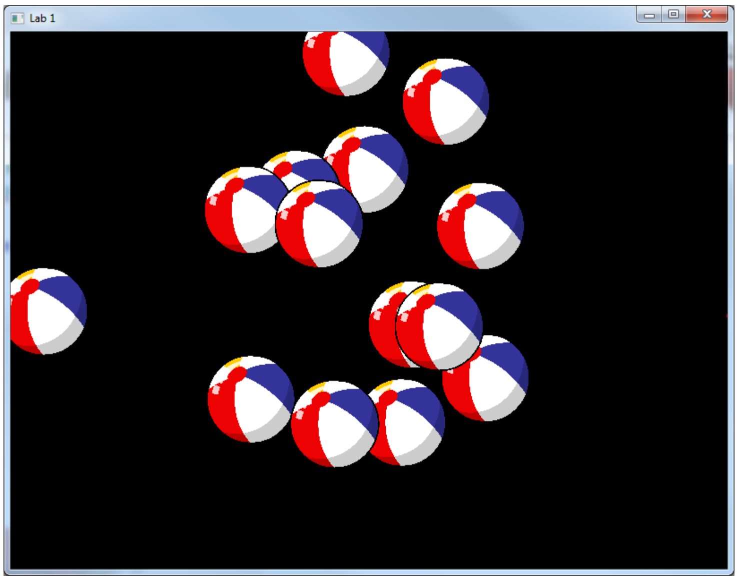
BeachBallManager.cpp

```cpp
#include "BeachBallManager.h"

BeachBallManager::BeachBallManager()
{
	// some default values
	spawnPoint = sf::Vector2f(350, 250);

	texture.loadFromFile("gfx/Beach_Ball.png");

	for (int i = 0; i < 20; i++)
	{
		balls.push_back(Ball());
		balls[i].setAlive(false);
		balls[i].setTexture(&texture);
		balls[i].setSize(sf::Vector2f(100, 100));
	}
}

BeachBallManager::~BeachBallManager()
{
}

void BeachBallManager::update(float dt)
{
	// call update on all ALIVE balls
	for (int i = 0; i < balls.size(); i++)
	{
		if (balls[i].isAlive())
		{
			balls[i].update(dt);
		}
	}
	deathCheck();
}

// Spawn new ball
// Find a dead ball, make alive, move to spawn point, give random velocity
void BeachBallManager::spawn()
{
	for (int i = 0; i < balls.size(); i++)
	{
		if (!balls[i].isAlive())
		{
			balls[i].setAlive(true);
			balls[i].setVelocity(rand() % 200 - 100, rand() % 200 - 100);
			balls[i].setPosition(spawnPoint);
			return;
		}
	}
}
```

```
// Check all ALIVE balls to see if outside screen/range, if so make dead
void BeachBallManager::deathCheck()
{
	for (int i = 0; i < balls.size(); i++)
	{
		if (balls[i].isAlive())
		{
			if (balls[i].getPosition().x < -100)
			{
				balls[i].setAlive(false);
			}
			if (balls[i].getPosition().x > 800)
			{
				balls[i].setAlive(false);
			}
			if (balls[i].getPosition().y < -100)
			{
				balls[i].setAlive(false);
			}
			if (balls[i].getPosition().y > 600)
			{
				balls[i].setAlive(false);
			}
		}
	}
}

// Render all alive balls
void BeachBallManager::render(sf::RenderWindow* window)
{
	for (int i = 0; i < balls.size(); i++)
	{
		if (balls[i].isAlive())
		{
			window->draw(balls[i]);
		}
	}
}
```

Now create the application shown in the lecture. This will require adding the BeachBallManager to the game class. The game class will need to call manager.update(), on keypress manager.spawn() and to render all the sprites manager.render().

The screen will be blank until key presses, then should look like this (after a few key presses):



Task 2

Create another Manager class. This class should handle 40 sprites. These sprites should spawn just outside the top of the window with a random x-axis position (between the left and right of the window). The sprites should move downwards and be killed when the pass the bottom of the window. The sprites should spawn on key press.

Task 3

Make use of any remaining time to make a start on your coursework.