Project 2 – Skeet Shooting

Program Description:

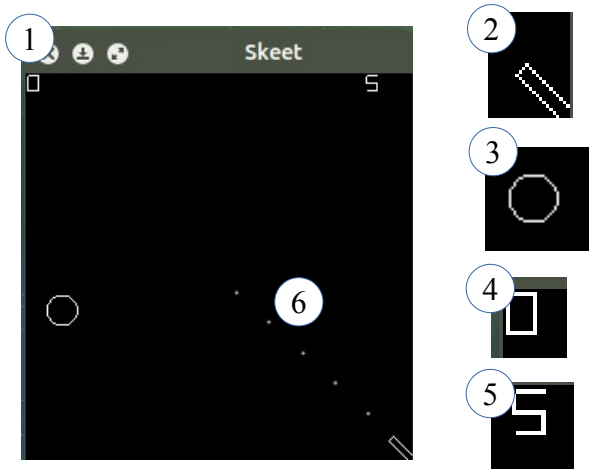
A simple game in which one player aims a gun at a piece of skeet which flies across the screen left to right at random intervals, and try to hit them. A score is kept throughout the game keeping track of how many are hit, and how many are missed. Each player may only have 5 bullets displayed on the screen at any given time.

Design Overview:

This game utilizes the OpenGL library written for C++ to display every output of the game. Classes will be utilized to encapsulate every aspect of the code, insuring that mistakes cannot be made. The user will be restricted to using the arrow keys and the space bar on their keyboard, all other input will be ignored. Vector based formulas will be used to determine if the pigeon is hit, or if it is not hit. The score at the top of the page will be kept count with the score class.

Interface Design:

Output:



1. The overall display of the game.
2. The rifle, displayed at the bottom right of the screen, can be 90 degrees or 180 degrees.
3. The pigeon / puck. It flies across the screen at random intervals and velocities.
4. The number of hits the user obtains
5. The number of pigeons missed by the user.
6. The bullets from the rifle. Five allowed on the screen at any given time.
7. Board refreshed 30 frames every 1 second.

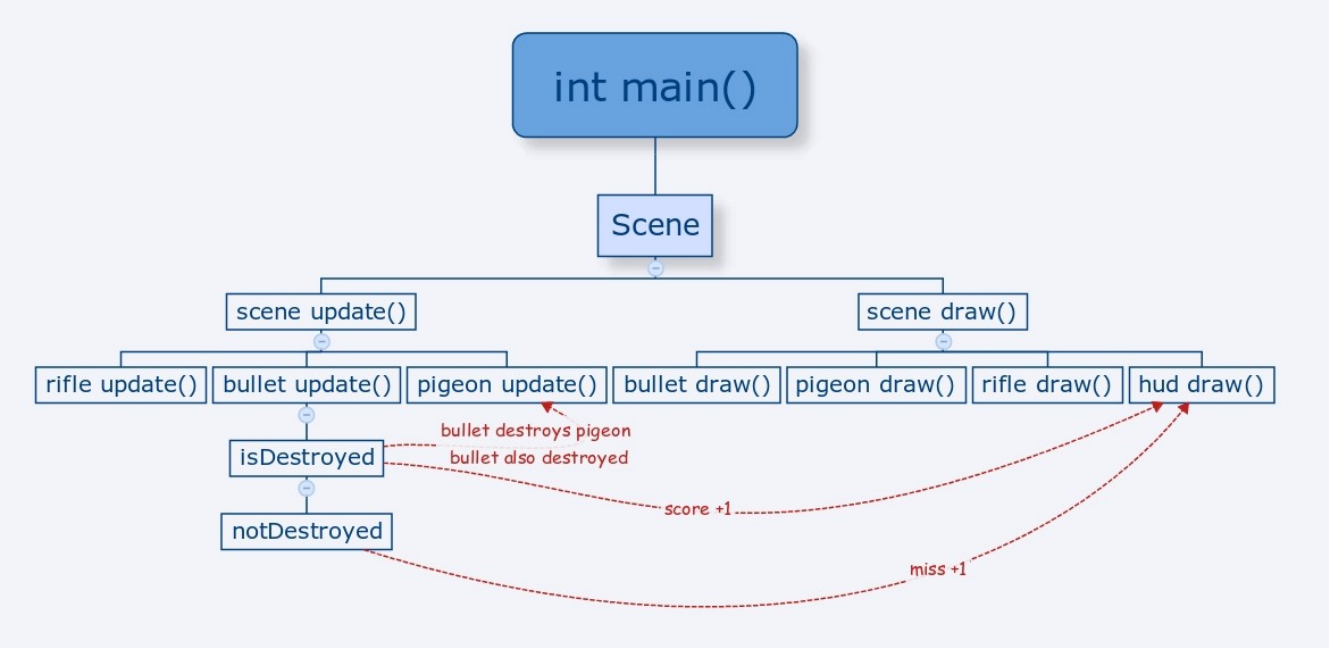
Input:

<spacebar>	Fires the rifle, allowing up to 5 bullets on the screen at one time.
<up arrow> <right arrow>	Move the rifle upward 3 deg per 5 frames arrow depressed.
<down arrow> <left arrow>	Move the rifle downward 3 deg per 5 frames arrow depressed.

Errors:

Do nothing!	When any key other than space bar or arrow keys are used.

Structure Chart:



Data Structures:

Scene
-bullets -birds -cannon -hud -interface
+update +draw

Vector
+x +y
+add +subtract +multiply +divide +rotate +toUnit +toString

Instance
-position -velocity -radius -isdestroyed
+getPosition +getVelocity +getSize +setPosition +setVelocity +setSize +isOutside +update +draw +destroy +toString

Bird (is an instant)
-size
+update +draw +reactTo -hitTest -reSpawn

Bullet (is an instant)
+update

Gun (is an instant)
-angle -width -height -ammo -clipSize
+update +draw +fire +turn -validate

Hud (is an instant)
-score
+draw +getScore

Algorithms:

```
Instance :: isOutside
        IF position x – size x / 2 < 0
                THEN true
        ENDIF
        IF position y – size y / 2 < 0
                THEN true
        ENDIF
        IF position x + size x / 2 >  USER DEFINED WIDTH
                THEN true
        ENDIF
        IF position y + size y / 2 > USER DEFINED HEIGHT
                THEN true
        ENDIF

RETURN false

Bullet :: fire(rifle)

        SET position x TO Xmax – 1
        SET position Y TO Ymin + 1

        SET position dx TO end of the rifle
        SET position dy TO end of the rifle
        SET dead TO true

RETURN NOTHING
END Bullet :: fire(rifle)

Bird :: isOutside()
        padding = random number between 50 and 100

        IF position x > xMax + padding || y > yMax + padding || y < yMin – padding-bottom
                THEN RETURN true
        ENDIF
END Bird :: isOutside()
```