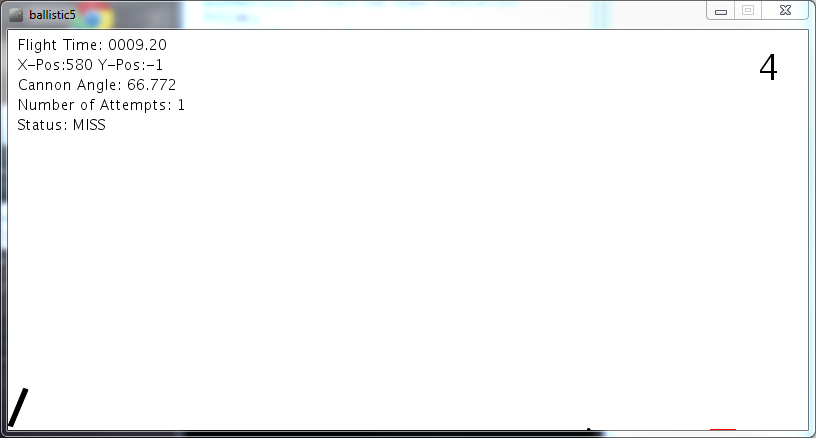
# Processing Assignment 5: Keeping Score & Game Restart

Assignment Due 11/18

## Description

This deliverable will include several new features over the Processing 4 assignment:

1. The game presents the number of the remaining attempts.
2. The game presents a ‘winner’ indicator when the user hits the target within number of attempts.
3. The game presents a ‘loser’ indicator when the user misses the target ‘attempts’ times.
4. The game halts when the user wins or loses i.e. the user can no long fire the cannon.
5. The game can be restarted after winning or losing (see 4). Restart resets the game screen (including the remaining attempts) and moves the target. I suggest using the ‘r’ key.



## Psudocode

### Changes to setup()

The function reset() was added to support the user resetting the game. Portions of the setup() were removed and moved into reset(); We also reset currT (current time) and maxT (flight time) to zero.

void setup() {

size(800, 400);

//frameRate(1);

**reset**();

}

void **reset**() {

targetStart = random(100, width-targetWidth);

numAttempts = 0;

**currT = 0;**

**maxT = 0;**

status = "halt";

}

### Changes to draw()

The only change to draw() was the addition of the drawScore() function which draws the user’s attempts in the game.

void draw() {

background(255);

if (status == "run") {

if (currT > maxT) {

checkTarget(); // Stop animation

}

else {

currT = currT + deltaT;

}

}

**drawScore**();

drawTarget();

drawCannon();

drawShot();

drawStatus();

}

### Adding the drawScore() function

The drawScore() function supports presenting the user’s attempts and the win/lose graphics. The function also stops Processing’s execution when the game is won or lost. Notice that a loss occurs when the user misses and there are no more attempts.

void **drawScore**() {

int score = maxAttempts - numAttempts;

textSize(35);

text(score, width-50, 50);

if(status == "HIT") {

String txt = "Winner";

text(txt, (width/2), (height/2));

**noLoop**(); // Halt the game

}

// if miss and no more attempts

else if (**status == "MISS" && score < 1**) {

String txt = "Loser";

text(txt, (width/2), (height/2));

**noLoop**(); // Halt the game

}

}

### Changes to the keyPress() function

The keyPressed() function is called when the user presses a key. The game reset ‘r’ command was added. This causes the reset() function to be called as described above.

void keyPressed() {

//println("Key Pressed: " + key);

if (key == CODED && keyCode == LEFT) {

cannonAngle = cannonAngle + .01;

if (cannonAngle > HALF\_PI) {

cannonAngle = HALF\_PI;

}

}

else if (key == CODED && keyCode == RIGHT) {

cannonAngle = cannonAngle - .01;

if (cannonAngle < 0) {

cannonAngle = 0;

}

}

else if (key == ' ') {

fireCannon();

}

else if (**key == 'r'**) {

**reset**();

**loop**(); // Restart Game Processing

}

}

## Deliverables

Each team will submit their sketchbook directory containing their submission on a USB thumb drive. Just copy the processing directory containing your work onto a USB thumb drive.

Also include a README.txt file on the USB drive containing your section and team numbers.   
**No File..No Grade.**

THE THUMB DRIVE MUST CONTAIN THE FOLDER WHICH IS THE PROCESSING SKETCHBOOK TO BE GRADED AND THE README FILE.

Graders will evaluate the submissions by executing the submitted program file and ensuring that it executes correctly.