|  |  |
| --- | --- |
| **Unit:** Methodology | **Turn In List:** **1. Terms** |
| *“I will vow to format code so that it is readable and easy to interpret. Good developers don’t try to hide things in source code.”* | |

**Conditions and Formatting Code: Using proper format while introducing conditions in code**

**Content Objectives:** Students will be able to identify and format code appropriately while using appropriate methods with return values.

|  |
| --- |
| **Starter Activity** |
|  |
| Students will modify etch\_a\_sketch with keyPressed and saveFrame.  A star in the background  Description automatically generated  //Global Variable //<>//  int x, y;  void setup() {  size(1000, 400);  background(#581845);  strokeWeight(5);  stroke(#900C3F);  x = 20;  y = 50;  }  void draw() {  fill(255);  if (keyPressed) {  if (key == 'w' || key == 'W') {  moveUp(5);  } else if (key == 'a' || key == 'A') {  moveLeft(5);  } else if (key == 's' || key == 'S') {  moveDown(5);  } else if (key == 'd' || key == 'D') {  moveRight(5);  }  }  //drawName();  //noLoop();  }  void mousePressed() {  saveFrame("line-######.png");  }  void keyPressed() {  if (key == CODED) {  if (keyCode == RIGHT) {  moveRight(10);  } else if (keyCode == LEFT) {  moveLeft(10);  } else if (keyCode == UP) {  moveUp(10);  } else if (keyCode == DOWN) {  moveDown(10);  }  }  }  // Algorithm for your first name  void drawName() {  moveRight(100);  moveDown(300);  moveLeft(100);  moveUp(100);  moveDown(100);  moveRight(120);  moveUp(150);  moveRight(100);  moveDown(150);  moveLeft(100);  moveRight(120);  moveUp(150);  moveRight(100);  moveDown(150);  moveRight(20);  moveUp(150);  moveRight(80);  moveDown(150);  moveLeft(80);  moveRight(80);  moveUp(20);  moveRightDown(20);  moveRight(20);  moveRight(50);  moveUp(300);  moveDown(120);  moveLeft(50);  moveRight(100);  moveLeft(50);  moveDown(180);  moveRight(70);  moveUp(300);  moveDown(150);  moveRight(100);  moveDown(150);  moveRight(20);  moveUp(150);  moveRight(80);  moveDown(150);  moveLeft(80);  moveRight(80);  moveUp(20);  moveRightDown(20);  moveRight(20);  moveUp(150);  moveRight(100);  moveDown(150);  moveRight(20);  }  //Method to draw left line  void moveLeft (int rep) {  for (int i = 0; i < rep; i++) {  point(x-i, y);  }  x=x-rep;  }  // Method to draw right line  void moveRight(int rep) {  for (int i = 0; i < rep; i++) {  point(x+i, y);  }  x=x+rep;  }  //Method to draw lines up  void moveUp (int rep) {  for (int i = 0; i < rep; i++) {  point(x, y-i);  }  y=y-rep;  }  //Method to draw lines down  void moveDown (int rep) {  for (int i = 0; i < rep; i++) {  point(x, y+i);  }  y=y+rep;  }  //Method to draw lines right and up  void moveRightUp (int rep) {  for (int i = 0; i < rep; i++) {  point(x+i, y-i);  }  x=x+rep;  y=y-rep;  }  //Method to draw lines right and down  void moveRightDown (int rep) {  for (int i = 0; i < rep; i++) {  point(x+i, y+i);  }  x=x+rep;  y=y+rep;  } |

|  |  |
| --- | --- |
| **Key Terms:** | |
| White Space | Makes code readable, stripped out by compiler when code is run. |
| Camel or Pascal Case | Camel case is lower first character, Pascal is capital first letter. |
| Condition | Any decision-making strategy usually involving an if statement or a switch statement. |
| If | Most common conditional statement allowing you to compare two or more values. |
| Else if | Asking for a more specific Boolean expression. |
| Boolean Expression | Any expression that evaluates to true or false. |
| Boolean Variable | Any variable that stores true or false. |

|  |
| --- |
| **Assignment:** |
| Students will explore methods with a return type. Consider the following:  C = (F – 32) \* (5 / 9)  \_\_\_\_\_\_ tempConverter(float \_\_\_\_\_\_\_\_) {  \_\_\_\_\_ \_\_\_\_\_\_\_ = \_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  }  Answer:  float convertToCel(float val) {  val = (val - 32)\*(5.0/9.0);  return val;  }  float convertToFah(float val) {  val = val\*(9.0/5.0)+32;  return val;  } |
| For this assignment students will create a conversion app that utilizes a method with a return value and the position of the mouse or a line on the screen controlled by the keyboard (or both). Make sure to include the following:   * Title and developer info (your name) * Onscreen instructions * Reference line or shape * Numbered increments and tic marks on screen (hint: use loop) * Updated total as the mouse moves or the arrow keys are pressed   Appropriate conversions may include any of the following:   * Any distance measurement i.e. miles to km etc. * Any volume measurement * Any currency conversion * Math functions i.e. squares or squareroots * Etc. |

Notes (Points of interest, mistakes, lessons learned, web resources, and thoughts):

|  |
| --- |
| A picture containing icon  Description automatically generated  PFont font1, font2;  boolean hover = false;  void setup() {  size(1201, 501);  font1 = loadFont("AndaleMono-15.vlw");  font2 = loadFont("Trattatello-20.vlw");  }  void draw() {  background(255);  fill(0);  strokeWeight(1);  println("mouseX: " + mouseX, "mouseY: " + mouseY);  textFont(font1);  //Reference items  ellipse(width/4, height/2, 300, 300);  line(1100, 0, 1100, height);  int i = height;  while (i>0) {  i-=20;  line(1095, i, 1105, i);  fill(0);  ellipse(1100, mouseY, 5, 5);  textSize(8);  text(mouseY\*0.1, 1075, mouseY+3);  textSize(15);  textAlign(CENTER);  fill(#29F72D);  text("Knot: " + mouseY\*0.1, width/4, 190);  text("KPH: " + knotToKph(mouseY\*0.1), width/4, 220);  text("MPH: " + knotToMph(mouseY\*0.1), width/4, 250);  text("M/S: " + knotToMps(mouseY\*0.1), width/4, 280);  text("F/S: " + knotToFps(mouseY\*0.1), width/4, 310);  }  steeringWheel();  info();  }  void steeringWheel() {  stroke(0);  fill(255);  strokeWeight(15);  ellipse(700, height/2, 300, 300);  fill(0);  ellipse(700, height/2, 50, 50);  line(700, height/2, 700, 100);  line(700, height/2, 560, 300);  line(700, height/2, 840, 300);  }  void info() {  //Developer information box  fill(255);  strokeWeight(2);  stroke(#B7CBF5);  rectMode(CENTER);  rect(width/2, 50, 365, 40);  //Developer information  textFont(font2);  textSize(20);  fill(#B7CBF5);  textAlign(CENTER);  text("Conversion App | Jonathan Su | 10/09/2020", width/2, 60);  //Onscreen instructions  textFont(font1);  hover = (mouseX > 30 && mouseX < 130 && mouseY > 350 && mouseY < 450);  if (hover == true) {  fill(0);  text("Move mouse up and down to see the conversion of knots to KPH, MPH, M/S, and F/S", width/2, 450);  }  fill(255, 0, 0);  stroke(255, 0, 0);  rect(80, 400, 100, 100);  fill(0);  text("SOS", 80, 410);  }  //Convert knot to kilometers per hour  float knotToKph (float val) {  val = val\*1.852;  return val;  }  //Convert knot to miles per hour  float knotToMph (float val) {  val = val\*1.151;  return val;  }  //Convert knot to meters per second  float knotToMps(float val) {  val = val/1.944;  return val;  }  //Convert knot to feet per second  float knotToFps (float val) {  val = val\*1.688;  return val;  } |