ANSWERS TO QUESTIONS IN SKILLS LIST
(Please note - numbers refer to the question's number in the Skills Inventory list.)

19. A for loop iterates a set amount of times, while a while loop iterates for as long as a condition is met.

22. Parameters are variables used when declaring a function. Arguments are the bits of data you pass through the parameters.

25. A class is like a cookie cutter, and an object is like the cookie it cuts. An object can be made from a class, and will inherit its properties.
26. A constructor function is what tells the program how to make an object. It initializes all of the values for a new object when that object is created.
27. Each class should have its own tab for organization purposes. Otherwise, the program gets way too long!

31. An array has a predefined maximum size, whereas an ArrayList can be resized dynamically.
32. If you iterate through a list forwards, then deleting an item will effectively skip the next item in the list. By iterating *backwards* through the list, any deleted items only cause items you've already passed to fall back one space, and so none get skipped.

37. PVector "should" be used when working with movement in two or more dimensions, as it can make calculations substantially easier.
42. A normalized vector is a vector scaled so that the magnitude is 1. This is super useful for making sure your speed doesn't exceed a certain amount, or for making sure you get consistent movement regardless of whether the player is holding multiple inputs.

LOCATIONS OF SKILLS INVENTORY ITEMS
(Each entry will reference a line in a specific class, with a short description of what the item is used for.)

1. Class Crate, line 42. Draws an outline around the crate graphic.
2. Class Main, line 65. Removes fill on setup.
3. Class Main, line 70. Sets rectMode to CENTER.

4. Class Main, lines 62 and 78. Literally just setup() and draw().
5. Class Main, line 64. Adds a default background colour in case of no background image.
6. Class Main, lines 295, 296, 298, and 302. Prevents player movement values from going outside of reasonable bounds.
7. Class Main, lines 256 and 308. Checks for keyboard/joystick/button input.

8. Class Main, lines 258, 261, and many, *many* more. For this example specifically, increments or decrements the player's horizontal input value.
9. Class Bullet, line 65 + line 67. Declared as a way to get the difference between two angles, and for readability. Used to rotate the projectile towards the opponent.
10. Class Main, line 14 + line 81. Declared to manage game state, used to check whether the game should display the menu or play the game.

11. Class Player, lines 272 and 366. Used to catch weapons that haven't been fully added yet.

12. Class Main, line 81. Used to check if the game should display the menu or play the game.
13. Class Main, lines 105 and 117. Used to check if a player has collected a weapon crate, and if not, returns false.
14. Class Main, lines 104 and 116. Used to check if a player is touching a crate *and* if they're holding their "shoot" button *and* if the crate hasn't despawned yet.
15. Class Player, lines 213 through 274 and 286 through 368. Used to set variables and spawn projectiles differently based on weapon.

16. Class Main, lines 165 through 196. Used to handle bullets, their motion, their collisions, and how they deal damage.
17. Class Main, lines 220 through 226. Used to generate tile objects when the game is first run.
18. Class Player, line 219 and more. Used to prevent every case underneath the required one from running when the switch statement is run.

20. Class Main, line 207 + line 84. Declared as a way to reset necessary values and objects, used right before the game state updates from "menu" to "playing."
21. Class Player, line 162 + line 125. Declared as a collision detection script, returns either true or false depending on whether a collision is detected or not. Used to… detect collisions.

23. Class Main, line 229 + lines 184 and 187. Declared as a quick way to spawn particles at a target location, with a specific number, velocity, and "friction." Used to spawn particles on player death to show who died first.
24. Class Main, line 248 + line 240. Declared as a way to check if particles are moving too slowly, and used in a conditional to remove said sluggish projectiles.

28. Literally any class aside from main. If I had to choose just one, I'd pick Class Tile, line 6, but I definitely don't have any one favourite class, noooo, that'd be craaaazy.
29. Class Main, lines 67, 68, and *69*. Used to create both Player objects and the Crate object.
30. Class Player, line 30. Used to specify player spawn positions, number (1 or 2), and initial health.

33. Class Player, lines 26 and 27 + lines 44 through 50. Used to store graphics as PImage objects.
34. Class Main, lines 19, 20, and 21 + line 231 (for example). Used to store Bullet, Tile, and DeathParticle objects. They're populated throughout the program, but the example I gave is specifically for the deathParticles ArrayList.
35. See above. It's exactly the same.
36. Class Main, line 166. Used to store the current Bullet object that the loop is managing for easy reference.

38. Do I have to pick just one example? Alright, alright. Class Player, lines 3 and 4. Used in a lot more places than this, but these lines handle declaration of PVector variables for position and velocity.
39. Class Bullet, line 40 (velocity) + line 41 (acceleration due to gravity) + line 43 (position). Used to handle bullet physics.
40. Class Bullet, line 67 (direction) + line 68 (distance). Used to find the direction towards the opponent, then multiply the angle difference by the distance between the origin (0, 0) and the bullet's velocity (basically just .mag() for dummies, but I had to add it *somewhere*, right?).
41. Class Main, line 231. Used to give death particles a random heading.

43. Class Bullet, line 68. The .rotate() function was new, and I used it to make homing missiles.

OPTIONAL ITEMS

45. Class Player, line 104 (and more). The onGround boolean variable is used to check if the player is on the ground or in the air.

48. Class Player, lines 26 and 27, and more. Used to display graphics for the player and held weapons.
49. Class Player, lines 162 through 174. Used to check collisions with the tilemap.