Snake Game part 8

Step 8: New Game Screen

Great! Our game works. Now we just need to add a new game screen, that only starts the game when you press SPACE.

We need a new gameOver variable to keep track of whether the game is in play or not.

```
boolean gameOver = true;
```

We need a new method to display the new game screen.

```
void newGameScreen() {
  textSize(24);
  textAlign(CENTER);
  text("Press SPACE To Play", width/2, height/2 - gridSize);
}
```

Let's refactor all of the methods that run our game, into a `runGame()` method.

```
void runGame() {
  showFood();
  eatFood(food);
  moveTail();
  moveSnake();
  showSnake();
  showTail();
  if (collide()) {
    newGame();
    gameOver = true;
  }
}
```

Then, we can use a simple if statement to either display the new game screen, or run the game.

```
void draw() {
  background(0);

  if (gameOver) {
    newGameScreen();
  } else {
    runGame();
  }
}
```

We will also add code to the `keyPressed()` method so that pressing space (when the game is over) will start the game.

```
  if (gameOver) {
    if (key == ' ') {
      gameOver = false;
    }
  }
```

The final code looks like this:

```
int gridSize = 20;

PVector snake;
int snakeSize;
int xSpeed;
int ySpeed;
int tailLength;
ArrayList<PVector> tail = new ArrayList();

PVector food;

boolean gameOver = true;

void setup() {
  size(600, 600);
  frameRate(10);
  newGame();
}

void draw() {
  background(0);

  if (gameOver) {
    newGameScreen();
  } else {
    runGame();
  }
}

void runGame() {
  showFood();
  eatFood(food);
  moveTail();
  moveSnake();
  showSnake();
  showTail();
  if (collide()) {
    newGame();
    gameOver = true;
```

```
  }
}

void newGame() {
  snake = new PVector(width/2, height/2);
  snakeSize = gridSize;
  direction(0, 0);
  newFood();
  tail.clear();
  tailLength = 0;
}

void newGameScreen() {
  textSize(24);
  textAlign(CENTER);
  text("Press SPACE To Play", width/2, height/2 - gridSize);
}

void showSnake() {
  fill(100, 215, 0);
  square(snake.x, snake.y, snakeSize);
}

void moveSnake() {
  snake.x += xSpeed;
  snake.y += ySpeed;

  // This keeps the snake on the board
  snake.x = constrain(snake.x, 0, width-gridSize);
  snake.y = constrain(snake.y, 0, height-gridSize);
}

void moveTail() {
  if (tailLength > 0) {
    if (tailLength == tail.size() && !tail.isEmpty()) {
      tail.remove(0);
    }
    tail.add(new PVector(snake.x, snake.y));
  }
}

void showTail() {
  for (PVector section : tail) {
    square(section.x, section.y, gridSize);
  }
}

void direction(int x, int y) {
  xSpeed = x * gridSize;
  ySpeed = y * gridSize;
}
```

```
void keyPressed() {
  if (keyCode == UP) {
    direction(0, -1);
  } else if (keyCode == DOWN) {
    direction(0, 1);
  } else if (keyCode == RIGHT) {
    direction(1, 0);
  } else if (keyCode == LEFT) {
    direction(-1, 0);
  }

  if (gameOver) {
    if (key == ' ') {
      gameOver = false;
    }
  }
}

void newFood() {
  int columns = width/gridSize;
  int rows = height/gridSize;
  int randomColumn = (int)random(columns) * gridSize;
  int randomRow = (int)random(rows) * gridSize;
  food = new PVector(randomColumn, randomRow);
}

void showFood() {
  fill(255, 0, 100);
  square(food.x, food.y, gridSize);
}

void eatFood(PVector food) {
  if (snake.x == food.x && snake.y == food.y) {
    tailLength++;
    newFood();
  }
}

boolean collide() {
  for (PVector section : tail) {
    if (snake.x == section.x && snake.y == section.y) {
      return true;
    }
  }
  return false;
}
```

Challenges

1. Make a variation of the game, where your snake can wrap around the screen. For example, when its head goes off to the left, it will come around on the right.
2. Add a high score.