Spike: Task 24
Title: Measuring Performance & Optimisations

Author: Hayden Whiteford, 104001272

Goals / deliverables:

1. A description of each collision test (method) approach (and the differences between them).
   - The method you used to collect your data and your reasons for choosing it
   - The raw data you collect to support your results.
   - A summary table of the results

Using the results of your measurements, modify the provided code to use the most collision detection method found to be the most optimal. When this is done, add your own modifications to further improve performance. Using your performance measurement method, demonstrate that your modification result in performance improvements (however minor they might be) over the most optimal path found.

Technologies, Tools, and Resources used:

- Xcode
- SDL2

Tasks undertaken:

- Recorded Data into Google Sheets and Made a report analysing each collision method
- Created a new collision function that performs even faster

What we found out:

Often, simpler is better. I initially looked into some of the more complicated collision logic, such as spatial partitioning, and rudimentary radius-based bounding circle collision, but every implementation of this was rather lacklustre, as computing these new ways to be more efficient often cost just as much as what I was trying to cut down on. The best solution I found was simply do add the break statement in the for loop. Doing this eliminated quite a bit of unnecessary collision checks.