Spike: Task 09
Title: Game Data Structures

Author: Hayden Whiteford, 104001272

Goals / deliverables:

1. Research and evaluate four different data structures that could be used to create the player inventory for the Zorkish game. At a minimum, you must show your awareness of advantages and disadvantages for this application. Document your evaluation criteria and results in a **short report**.

2. Using your decision (as documented in your short report), create a working inventory system demonstration program. Your work must demonstrate (bug free) inventory **access** (view), **addition** and **removal**.

Technologies, Tools, and Resources used:
List of information needed by someone trying to reproduce this work
- Xcode
- http://en.cppreference.com/w/cpp/container - list of c++ data structures

Tasks undertaken:
- Created a new Xcode project
- Created a simple GameObject class containing a description
- Created 4 inventory variables using a Vector, List, Array and Deque
- Created a timed test for each inventory for Access, Addition and Removal
- Created a Short evaluation report summarising my findings and choosing my data structure for the Zorkish game

What we found out:
The conclusion I came to in the short report was, to summarise:

Vectors
Fast access and removal, slow addition

Lists
Fast addition, slow access and removal

Array
Fast access, slow addition and removal

Deque
Fast access and addition, slow removal

And finally I decided on using Deques over Vectors, as I assume my game will contain more inventory addition than removal.

Open issues/risks:

My array testing was not robust enough - because an Array structure is not dynamic in size, the addition and removal testing is not quite correct as it is simply creating a new array for each test. But a more precise use case would be to iterate through the original array and move/copy each element into a new array with the additional item, or without the removed item. Either way it would have resulted in a far slower performing structure, which was unfavourable anyway.