Spike: 06
Title: Basic Game Data Structures

Author: Ford Killeen, 9731822

Goals / deliverables:
Goals this spike aims to achieve:

- Research and evaluate four different data structures
- Write a short report detailing the advantages and disadvantages of using each data structure for a game inventory system in the context of our Zorkish game
- Using the chosen data structure, implement an inventory system into your Zorkish game, capable of adding, accessing and removing objects

Deliverables required:

- Code for the inventory system
- Short report on four different data structures
- Spike report

Technologies, Tools, and Resources used:
The following is required to complete this spike:

- Visual Studio 2015
- Zorkish game specification
- Online C++ data structure references
  (eg. http://en.cppreference.com/w/cpp/container)

Tasks undertaken:
The list below details the steps taken to complete this spike.

- Using online references, four C++ data structures were chosen and investigated
- The pros and cons of each data structure were found out and put into a short report, also detailing the chosen data structure for the inventory
- Using the chosen data structure, I started on the `Inventory` class which uses the `vector` class to contain all inventory items

```
class Inventory {
public:
    Inventory();
    ~Inventory();
    bool addItem(InventoryItem* item);
    InventoryItem* getItem(int id);
    void update();
    void renderItems();
private:
    size_t maxItemCount;
    vector<InventoryItem*> items;
};
```

- The `InventoryItem` was then set up as the base class for the rest of the inventory items

```
class InventoryItem {
public:
    virtual int getID() = 0;
    virtual string getName() = 0;
    virtual string getDescription() = 0;
    virtual int getQuantity() = 0;
    virtual bool increaseQuantity(int amount) = 0;
    virtual void use() = 0;
    //Subclass IDs
    static const int SWORD = 1;
    static const int HEALTHPOTION = 2;
    static const int KEY = 3;
};
```

- With the base class ready I set up a `Sword`, `Key` and `HealthPotion` classes which inherit from the base `InventoryItem` class
- To show off the inventory I used the existing `GameplayState`. You can see in the image below addition, removal and access of inventory items within the inventory

```
Enter your next move > show inventory
Your inventory is empty!
Enter your next move > pickup sword
Enter your next move > pickup key
Enter your next move > show inventory
Your inventory contains:
  1 x Sword
  1 x Key
Enter your next move > pickup key
Enter your next move > pickup health potion
Enter your next move > show inventory
Your inventory contains:
  1 x Sword
  2 x Key
  1 x Health Potion
Enter your next move > use health potion
You use your health potion and regain 5 health!
Enter your next move > show inventory
Your inventory contains:
  1 x Sword
  2 x Key
Enter your next move >
```

What we found out:

By completing this spike we found out the advantages and disadvantages of four main data structures and how they can be used in games. We applied the knowledge we gained about these data structures to make an informed decision on which data structure to use for our Zorkish inventory system and then implemented it. By implementing it I was able to re-enforce my understanding and usage of the `vector` data structure.