Hearthstone Database Demonstration Test Plan

Santosh Lakshman, Barr Avrahamov, Gokul Nair, Noah Gutierrez

1. Open exe
2. Demonstrate file path choosing
    a. Use default path
3. Demonstrate Sorted Output
    a. Show output
4. Demonstrate Add function
    a. Show input validation
        i. Type invalid values first and then type properly formatted values
5. Demonstrate Remove function
    a. Remove an element that was in the file already

    b. Remove the same element again

        i. Show that you can’t remove an item not in the list

6. Demonstrate hash table Efficiency

    a. Show output

7. Demonstrate Indented BST Feature

    a. Show output

8. Demonstrate hash table sequence

    a. Show output

    b. Show that the elements added and removed are consistent

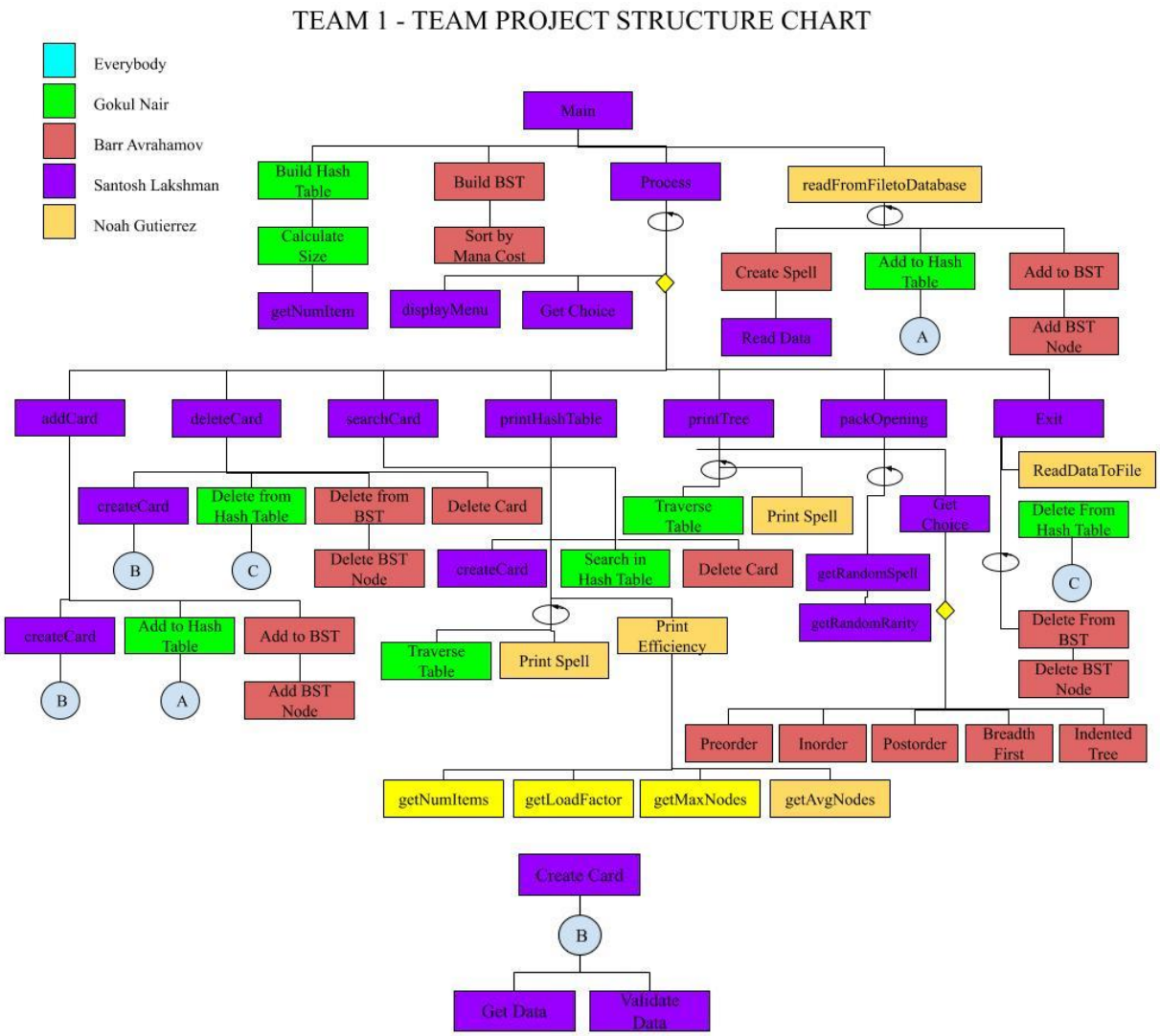
9. Demonstrate Sorted output

    a. Show output

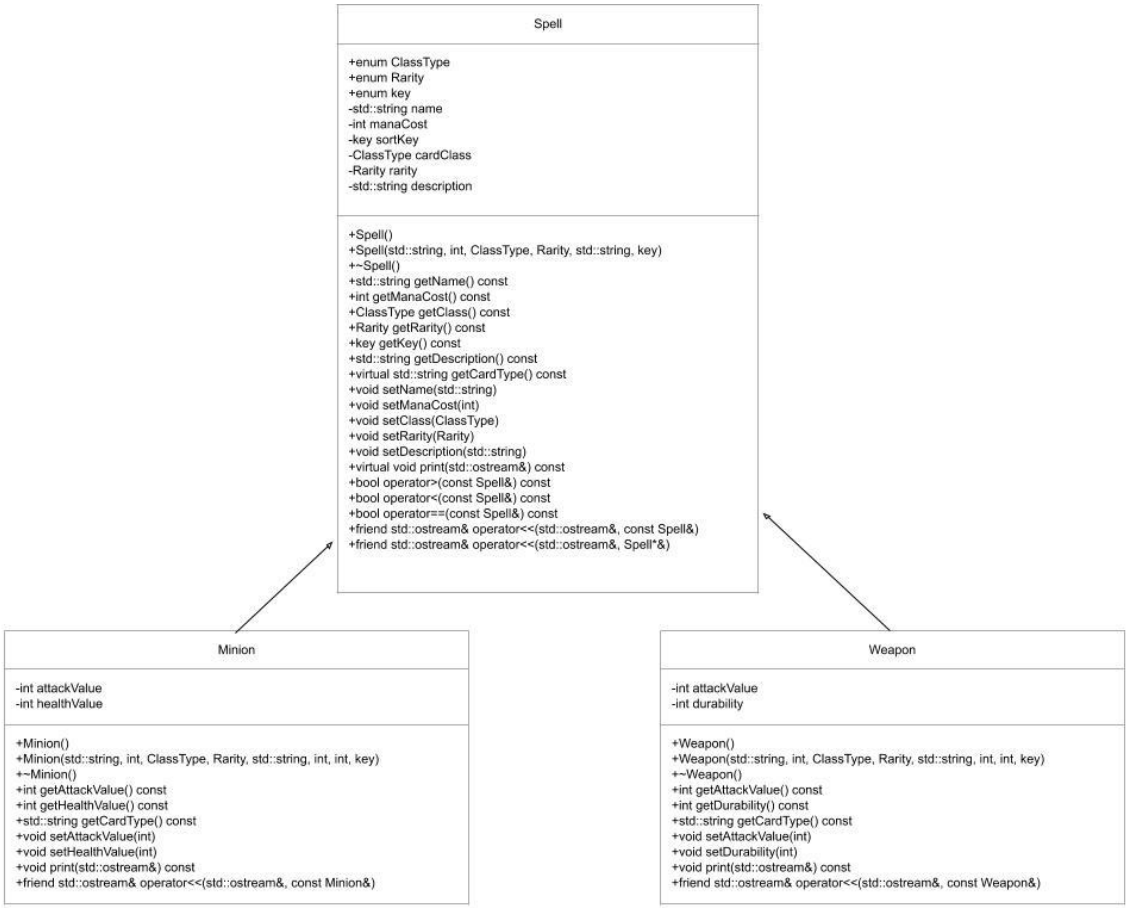
    b. Show that elements are consistent with hash table output

10. Demonstrate Card pack opening
    a. Show output
    b. Describe the rarities of each card
    c. Run twice to show random output
11. Exit Program and open again using the same file
    a. Demonstrate that all previous changes were saved to the file
        i. Print sorted output
12. Open CSV file in Excel to show easy editing

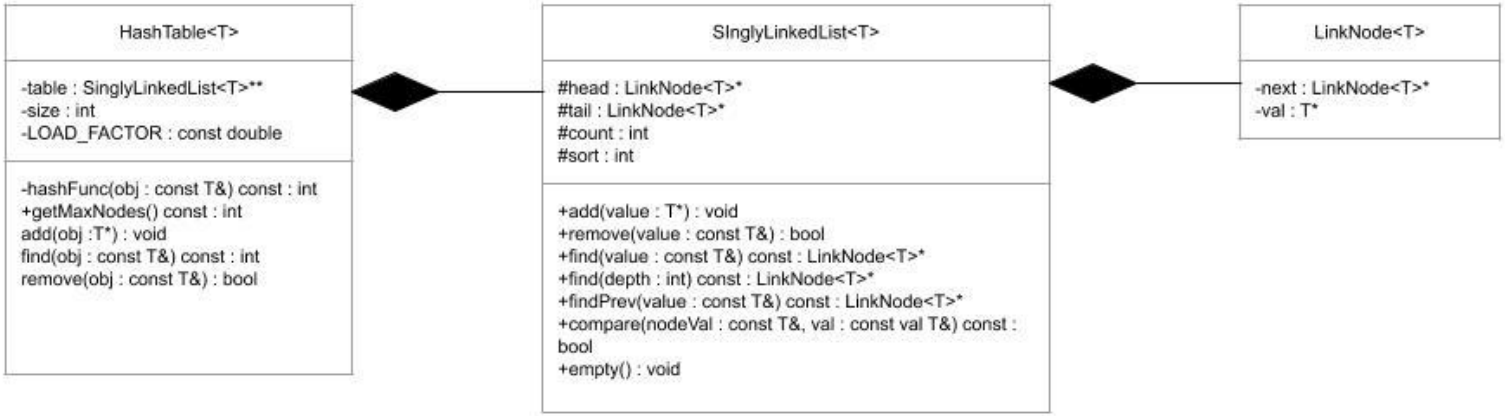
Structure Chart



UML


Spell
+enum ClassType
+enum Rarity
+enum key
-std::string name
-int manaCost
-key sortKey
-ClassType cardClass
-Rarity rarity
-std::string description
+Spell()
+Spell(std::string, int, ClassType, Rarity, std::string, key)
+~Spell()
+std::string getName() const
+int getManaCost() const
+ClassType getClass() const
+Rarity getRarity() const
+key getKey() const
+std::string getDescription() const
+virtual std::string getCardType() const
+void setName(std::string)
+void setManaCost(int)
+void setClass(ClassType)
+void setRarity(Rarity)
+void setDescription(std::string)
+virtual void print(std::ostream&) const
+bool operator>(const Spell&) const
+bool operator<(const Spell&) const
+bool operator==(const Spell&) const
+friend std::ostream& operator<<(std::ostream&, const Spell&)
+friend std::ostream& operator<<(std::ostream&, Spell*&)
Minion
-int attackValue
-int healthValue
+Minion()
+Minion(std::string, int, ClassType, Rarity, std::string, int, int, key)
+~Minion()
+int getAttackValue() const
+int getHealthValue() const
+std::string getCardType() const
+void setAttackValue(int)
+void setHealthValue(int)
+void print(std::ostream&) const
+friend std::ostream& operator<<(std::ostream&, const Minion&)
Weapon
-int attackValue
-int durability
+Weapon()
+Weapon(std::string, int, ClassType, Rarity, std::string, int, int, key)
+~Weapon()
+int getAttackValue() const
+int getDurability() const
+std::string getCardType() const
+void setAttackValue(int)
+void setDurability(int)
+void print(std::ostream&) const
+friend std::ostream& operator<<(std::ostream&, const Weapon&)


BSTNode
-T data
-BSTNode<T>* leftChild
-BSTNode<T>* rightChild
+BSTNode()
+BSTNode(T)
+~BSTNode()
+BSTNode(T, BSTNode<T>*, BSTNode<T>*)
+void setData(T)
+void setRightChild(BSTNode<T>*)
+void setLeftChild(BSTNode<T>*)
+T getData() const
+BSTNode<T>*& getRightChild()
+BSTNode<T>*& getLeftChild()
BST
-BSTNode<T>* rootPtr
-void recursiveAdd(BSTNode<T>*&, BSTNode<T>*)
-bool recursiveDelete(T, BSTNode<T>*)
-bool recursiveSearch(BSTNode<T>*, T) const
-BSTNode<T> findLargest(BSTNode<T>*)
-void recurPreOrder(BSTNode<T>*, std::ostream&) const
-void recurPostOrder(BSTNode<T>*, std::ostream&) const
-void recurInOrder(BSTNode<T>*, std::ostream&) const
-void recurBreadthFirst(BSTNode<T>*, std::ostream&) const
-void recurPrintCardsIndent(BSTNode<T>*, std::ostream&) const
-void recurEmpty(BSTNode<T>*&);
+BST()
+~BST()
+T getRootData() const
+void setRootData(T)
+bool deleteNode(T)
+void addNode(T)
+bool search(T) const
+void preOrderTraversalPrint(std::ostream& out) const
+void postOrderTraversalPrint(std::ostream& out) const
+void inOrderTraversalPrint(std::ostream& out) const
+void breadthFirstTraversalPrint(std::ostream& out) const
+void printCardsIndent(std::ostream& out) const
+void empty()
+bool isEmpty() const


HashTable<T>
-table : SinglyLinkedList<T>**
-size : int
-LOAD_FACTOR : const double
-hashFunc(obj : const T&) const : int
+getMaxNodes() const : int
add(obj :T*) : void
find(obj : const T&) const : int
remove(obj : const T&) : bool
SInglyLinkedList<T>
#head : LinkNode<T>*
#tail : LinkNode<T>*
#count : int
#sort : int
+add(value : T*) : void
+remove(value : const T&) : bool
+find(value : const T&) const : LinkNode<T>*
+find(depth : int) const : LinkNode<T>*
+findPrev(value : const T&) const : LinkNode<T>*
+compare(nodeVal : const T&, val : const val T&) const : bool
+empty() : void
LinkNode<T>
-next : LinkNode<T>*
-val : T*

Data Structure Diagram