Mythical Creature Catalog

Team #7

Team Members:

Shunyao Jin, Leyan Pan, Zachary Rudin, Ye Zhang

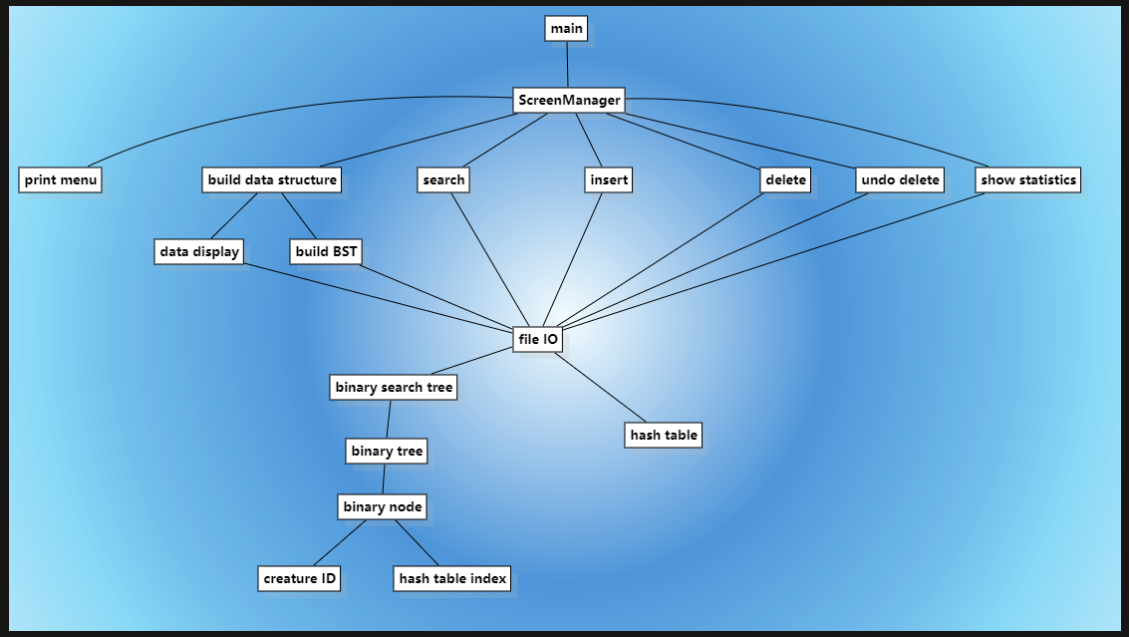
Application Data: Creature

- Creature ID (String) – Primary Key, abbreviated name – abbreviated creature type, e.g. FNKS-BD
- Name (String) – Full name of creature
- Category (String) – Type of creature, e.g. Bird
- History (String) – Famous Legend/Myth
- Habitat (String) – where the creature lives
- Description (String) – standout characteristics
- Relevant Year (Integer) – year of first mention/came to relevancy

Hash function:

```cpp
// Hash function implementation
int HashTable::hashFunction(const string& key) const {
    unsigned long hash = 5381;
    for (char c : key) {
        hash = ((hash << 5) + hash) + c; // hash * 33 + c
    }
    return hash % capacity;
}
```

Structure Chart



Zachary Rudin (Team Leader)

- Coordinated the project: defined milestones, assigned tasks, and maintained the overall schedule
- Authored weekly progress reports and designed the project poster
- Developed main.cpp, implementing the interactive menu and wiring up all menu-driven features
- Led system testing to ensure program stability and correctness

Leyan Pan (Hash Table)

- Designed and implemented the hash table module using the classic polynomial rolling hash (djb2)
- Optimized performance by using bit-shift operations (<<) instead of multiplication
- Handled collisions via chaining; each Creature object is stored (in full) in the hash table

Shunyao Jin (BST & File I/O)

- Implemented a binary search tree where each node holds a creature's key and its index in the hash table
- Developed the File I/O component to read from and write to the data file, maintaining both the BST and hash table structures
- Added automatic rehashing logic in File I/O, ensuring that after a resize the BST node indices remain in sync

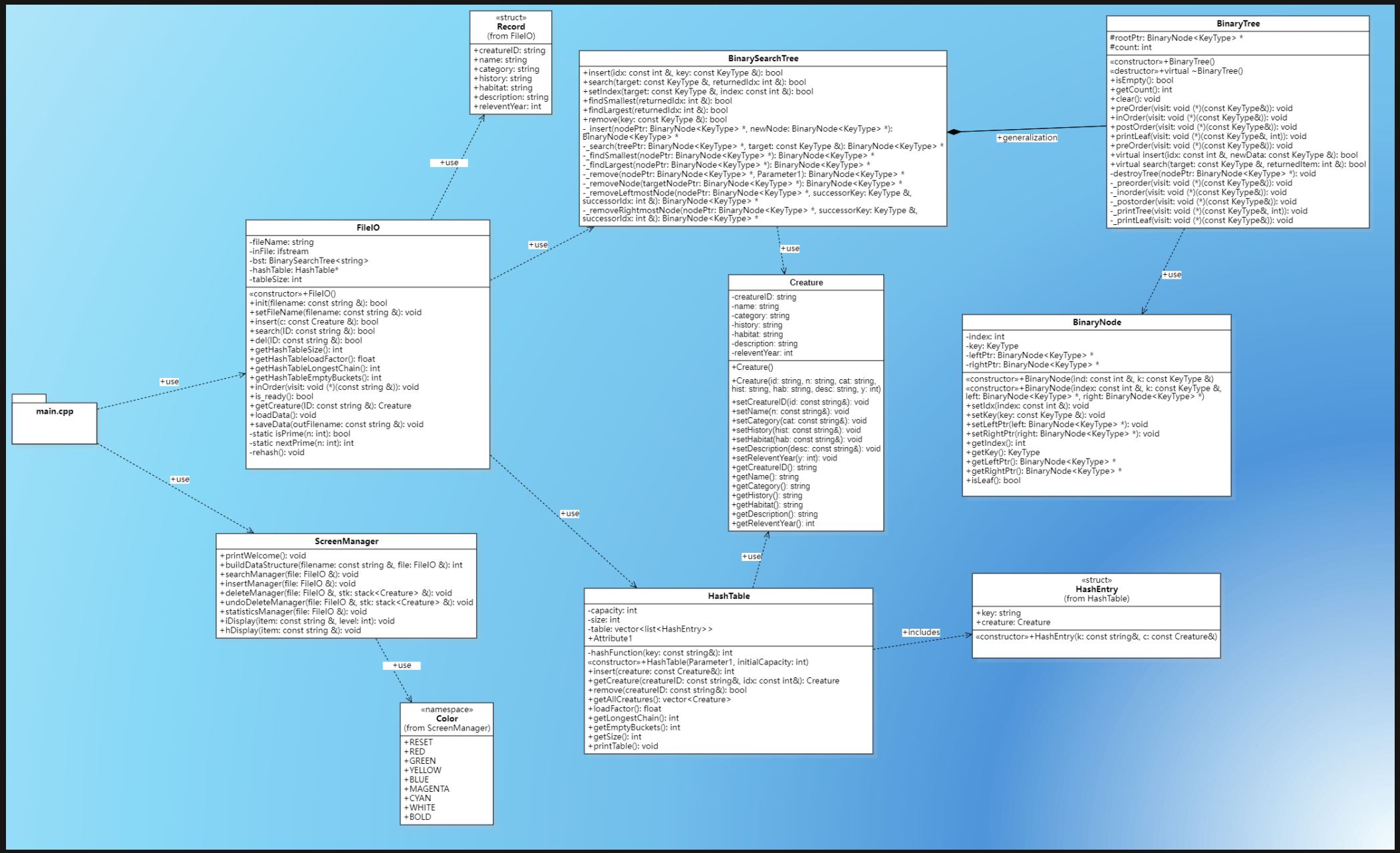
Ye Zhang (ScreenManager)

- Built ScreenManager.h, which drives all user interactions and menu features
- Acts as the bridge between the user interface and the underlying database, formatting and displaying query results and prompts

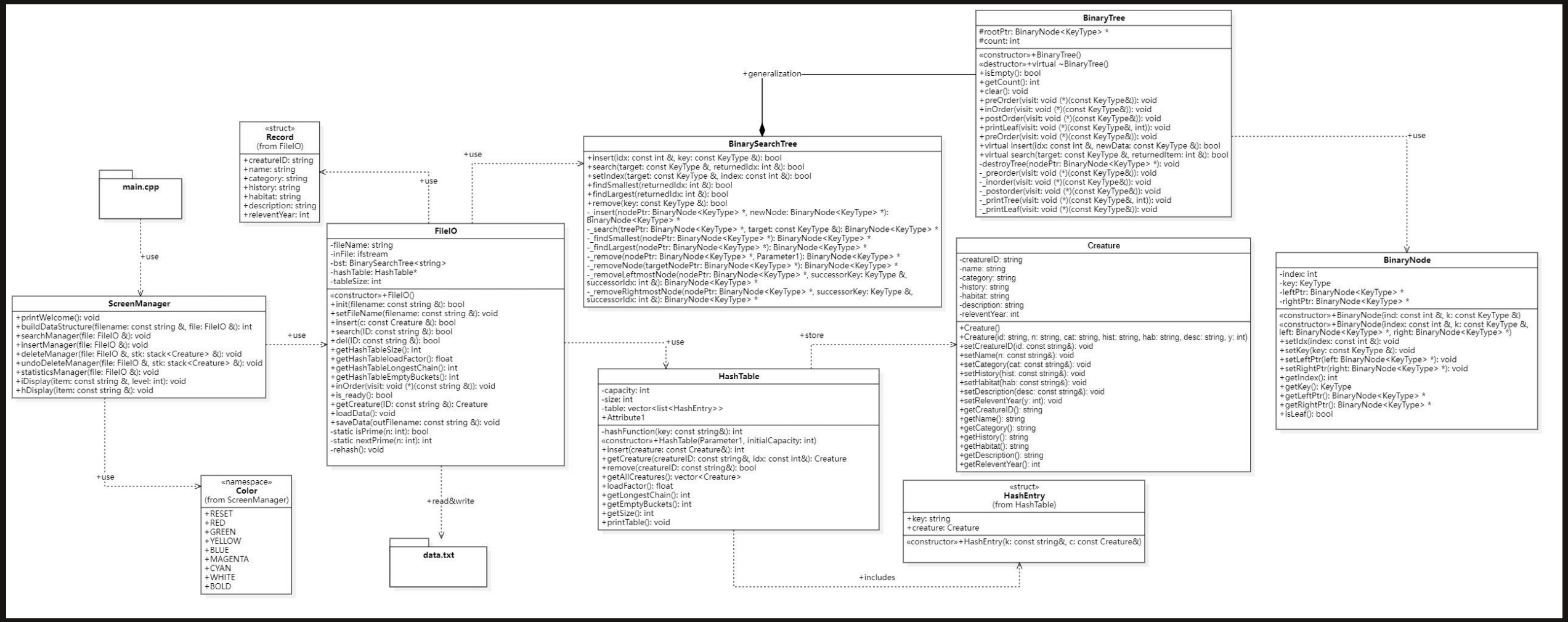
Sample Run:

```
Warning: Data file not loaded. All commands except for loading file are currently unavailable.

Main Menu:
  [L] - Load data file
  [A] - Add data
  [S] - Search data (Primary key: creature_id)
  [D] - Delete data
  [U] - Undo delete
  [P] - Print data
  [T] - Show statistics
  [W] - Write to file
  [H] - Help
  [Q] - Quit
User command: L

Warning: Data file not loaded. All commands except for loading file are currently unavailable.

What is the input file's name? Press Enter for Default "Creatures.txt": test
Reading data from "test..."
Error opening the input file: "test"
Reenter your data file or leave blank for default...
no
Reading data from "no..."
Error opening the input file: "no"
Reenter your data file or leave blank for default...

Reading data from "Creatures.txt..."
File Creatures.txt was successfully loaded!

User command: H
Main Menu:
  [L] - Load data file
  [A] - Add data
  [S] - Search data (Primary key: creature_id)
  [D] - Delete data
  [U] - Undo delete
  [P] - Print data
  [T] - Show statistics
  [W] - Write to file
  [H] - Help
  [Q] - Quit

User command: P
========== All stored data in order ==========
(BSLSK-RE) (FNKS-BD) (KRKN-OC) (XMR-CE) (YTI-HM)

User command: T
========== Statistics in HashTable ==========
Number of data: 5
Load factor: 0.454545
Longest chain in hash table: 2
Empty buckets: 7


User command: S
========== Searching ==========
Please enter a creature ID: XMR-CE

Found the creature "XMR-CE":

Creature ID: XMR-CE
Name:  Chimera
Category:  Hybrid
History:  Greek myth describes it as a fire-breathing beast with the head of a lion, body of a goat, and tail of a serpent.
Habitat:  Mountains of Lycia (modern-day Turkey)
Description:  Three-headed monster symbolic of chaos and unnatural combination.
Year: -600

User command: D
========== Deleting a creature ==========
Please enter a creature ID to delete: XMR-CE
Creature XMR-CE was successfully deleted!

User command: P
========== All stored data in order ==========
(BSLSK-RE) (FNKS-BD) (KRKN-OC) (YTI-HM)

User command: U
========== Undo deleting ==========
Undo delete succeeded!

User command: P
========== All stored data in order ==========
(BSLSK-RE) (FNKS-BD) (KRKN-OC) (XMR-CE) (YTI-HM)

User command: Q
Thank you for visiting!
```

UML diagram


«struct»
Record
(from FileIO)
+creatureID: string
+name: string
+category: string
+history: string
+habitat: string
+description: string
+relevantYear: int
BinarySearchTree
+insert(idx: const int &, key: const KeyType &): bool
+search(target: const KeyType &, returnedIdx: int &): bool
+setIndex(target: const KeyType &, index: const int &): bool
+findSmallest(returnedIdx: int &): bool
+findLargest(returnedIdx: int &): bool
+remove(key: const KeyType &): bool
-_insert(nodePtr: BinaryNode<KeyType> *, newNode: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_search(treePtr: BinaryNode<KeyType> *, target: const KeyType &): BinaryNode<KeyType> *
-_findSmallest(nodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_findLargest(nodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_remove(nodePtr: BinaryNode<KeyType> *, Parameter1): BinaryNode<KeyType> *
-_removeNode(targetNodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_removeLeftmostNode(nodePtr: BinaryNode<KeyType> *, successorKey: KeyType &, successorIdx: int &): BinaryNode<KeyType> *
-_removeRightmostNode(nodePtr: BinaryNode<KeyType> *, successorKey: KeyType &, successorIdx: int &): BinaryNode<KeyType> *
BinaryTree
#rootPtr: BinaryNode<KeyType> *
#count: int
«constructor»+BinaryTree()
«destructor»+virtual ~BinaryTree()
+isEmpty(): bool
+getCount(): int
+clear(): void
+preOrder(visit: void (*)(const KeyType&)): void
+inOrder(visit: void (*)(const KeyType&)): void
+postOrder(visit: void (*)(const KeyType&)): void
+printLeaf(visit: void (*)(const KeyType&, int)): void
+preOrder(visit: void (*)(const KeyType&)): void
+virtual insert(idx: const int &, newData: const KeyType &): bool
+virtual search(target: const KeyType &, returnedItem: int &): bool
-destroyTree(nodePtr: BinaryNode<KeyType> *): void
-_preorder(visit: void (*)(const KeyType&)): void
-_inorder(visit: void (*)(const KeyType&)): void
-_postorder(visit: void (*)(const KeyType&)): void
-_printTree(visit: void (*)(const KeyType&, int)): void
-_printLeaf(visit: void (*)(const KeyType&)): void
+generalization
+use
FileIO
-fileName: string
-inFile: ifstream
-bst: BinarySearchTree<string>
-hashTable: HashTable*
-tableSize: int
«constructor»+FileIO()
+init(filename: const string &): bool
+setFileName(filename: const string &): void
+insert(c: const Creature &): bool
+search(ID: const string &): bool
+del(ID: const string &): bool
+getHashTableSize(): int
+getHashTableloadFactor(): float
+getHashTableLongestChain(): int
+getHashTableEmptyBuckets(): int
+inOrder(visit: void (*)(const string &)): void
+is_ready(): bool
+getCreature(ID: const string &): Creature
+loadData(): void
+saveData(outFilename: const string &): void
-static isPrime(n: int): bool
-static nextPrime(n: int): int
-rehash(): void
Creature
-creatureID: string
-name: string
-category: string
-history: string
-habitat: string
-description: string
-relevantYear: int
+Creature()
+Creature(id: string, n: string, cat: string, hist: string, hab: string, desc: string, y: int)
+setCreatureID(id: const string&): void
+setName(n: const string&): void
+setCategory(cat: const string&): void
+setHistory(hist: const string&): void
+setHabitat(hab: const string&): void
+setDescription(desc: const string&): void
+setRelevantYear(y: int): void
+getCreatureID(): string
+getName(): string
+getCategory(): string
+getHistory(): string
+getHabitat(): string
+getDescription(): string
+getRelevantYear(): int
BinaryNode
-index: int
-key: KeyType
-leftPtr: BinaryNode<KeyType> *
-rightPtr: BinaryNode<KeyType> *
«constructor»+BinaryNode(ind: const int &, k: const KeyType &)
«constructor»+BinaryNode(index: const int &, k: const KeyType &, left: BinaryNode<KeyType> *, right: BinaryNode<KeyType> *)
+setIdx(index: const int &): void
+setKey(key: const KeyType &): void
+setLeftPtr(left: BinaryNode<KeyType> *): void
+setRightPtr(right: BinaryNode<KeyType> *): void
+getIndex(): int
+getKey(): KeyType
+getLeftPtr(): BinaryNode<KeyType> *
+getRightPtr(): BinaryNode<KeyType> *
+isLeaf(): bool
main.cpp
ScreenManager
+printWelcome(): void
+buildDataStructure(filename: const string &, file: FileIO &): int
+searchManager(file: FileIO &): void
+insertManager(file: FileIO &): void
+deleteManager(file: FileIO &, stk: stack<Creature> &): void
+undoDeleteManager(file: FileIO &, stk: stack<Creature> &): void
+statisticsManager(file: FileIO &): void
+iDisplay(item: const string &, level: int): void
+hDisplay(item: const string &): void
«namespace»
Color
(from ScreenManager)
+RESET
+RED
+GREEN
+YELLOW
+BLUE
+MAGENTA
+CYAN
+WHITE
+BOLD
HashTable
-capacity: int
-size: int
-table: vector<list<HashEntry>>
+Attribute1
-hashFunction(key: const string&): int
«constructor»+HashTable(Parameter1, initialCapacity: int)
+insert(creature: const Creature&): int
+getCreature(creatureID: const string&, idx: const int&): Creature
+remove(creatureID: const string&): bool
+getAllCreatures(): vector<Creature>
+loadFactor(): float
+getLongestChain(): int
+getEmptyBuckets(): int
+getSize(): int
+printTable(): void
+includes
«struct»
HashEntry
(from HashTable)
+key: string
+creature: Creature
«constructor»+HashEntry(k: const string&, c: const Creature&)

UML diagram


BinaryTree
#rootPtr: BinaryNode<KeyType> *
#count: int
«constructor»+BinaryTree()
«destructor»+virtual ~BinaryTree()
+isEmpty(): bool
+getCount(): int
+clear(): void
+preOrder(visit: void (*)(const KeyType&)): void
+inOrder(visit: void (*)(const KeyType&)): void
+postOrder(visit: void (*)(const KeyType&)): void
+printLeaf(visit: void (*)(const KeyType&, int)): void
+preOrder(visit: void (*)(const KeyType&)): void
+virtual insert(idx: const int &, newData: const KeyType &): bool
+virtual search(target: const KeyType &, returnedItem: int &): bool
-destroyTree(nodePtr: BinaryNode<KeyType> *): void
-_preorder(visit: void (*)(const KeyType&)): void
-_inorder(visit: void (*)(const KeyType&)): void
-_postorder(visit: void (*)(const KeyType&)): void
-_printTree(visit: void (*)(const KeyType&, int)): void
-_printLeaf(visit: void (*)(const KeyType&)): void
+generalization
+use
«struct»
Record
(from FileIO)
+creatureID: string
+name: string
+category: string
+history: string
+habitat: string
+description: string
+releventYear: int
BinarySearchTree
+insert(idx: const int &, key: const KeyType &): bool
+search(target: const KeyType &, returnedIdx: int &): bool
+setIndex(target: const KeyType &, index: const int &): bool
+findSmallest(returnedIdx: int &): bool
+findLargest(returnedIdx: int &): bool
+remove(key: const KeyType &): bool
-_insert(nodePtr: BinaryNode<KeyType> *, newNode: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_search(treePtr: BinaryNode<KeyType> *, target: const KeyType &): BinaryNode<KeyType> *
-_findSmallest(nodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_findLargest(nodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_remove(nodePtr: BinaryNode<KeyType> *, Parameter1): BinaryNode<KeyType> *
-_removeNode(targetNodePtr: BinaryNode<KeyType> *): BinaryNode<KeyType> *
-_removeLeftmostNode(nodePtr: BinaryNode<KeyType> *, successorKey: KeyType &, successorIdx: int &): BinaryNode<KeyType> *
-_removeRightmostNode(nodePtr: BinaryNode<KeyType> *, successorKey: KeyType &, successorIdx: int &): BinaryNode<KeyType> *
main.cpp
FileIO
-fileName: string
-inFile: ifstream
-bst: BinarySearchTree<string>
-hashTable: HashTable*
-tableSize: int
«constructor»+FileIO()
+init(filename: const string &): bool
+setFileName(filename: const string &): void
+insert(c: const Creature &): bool
+search(ID: const string &): bool
+del(ID: const string &): bool
+getHashTableSize(): int
+getHashTableloadFactor(): float
+getHashTableLongestChain(): int
+getHashTableEmptyBuckets(): int
+inOrder(visit: void (*)(const string &)): void
+is_ready(): bool
+getCreature(ID: const string &): Creature
+loadData(): void
+saveData(outFilename: const string &): void
-static isPrime(n: int): bool
-static nextPrime(n: int): int
-rehash(): void
Creature
-creatureID: string
-name: string
-category: string
-history: string
-habitat: string
-description: string
-releventYear: int
+Creature()
+Creature(id: string, n: string, cat: string, hist: string, hab: string, desc: string, y: int)
+setCreatureID(id: const string&): void
+setName(n: const string&): void
+setCategory(cat: const string&): void
+setHistory(hist: const string&): void
+setHabitat(hab: const string&): void
+setDescription(desc: const string&): void
+setReleventYear(y: int): void
+getCreatureID(): string
+getName(): string
+getCategory(): string
+getHistory(): string
+getHabitat(): string
+getDescription(): string
+getReleventYear(): int
BinaryNode
-index: int
-key: KeyType
-leftPtr: BinaryNode<KeyType> *
-rightPtr: BinaryNode<KeyType> *
«constructor»+BinaryNode(ind: const int &, k: const KeyType &)
«constructor»+BinaryNode(index: const int &, k: const KeyType &, left: BinaryNode<KeyType> *, right: BinaryNode<KeyType> *)
+setIdx(index: const int &): void
+setKey(key: const KeyType &): void
+setLeftPtr(left: BinaryNode<KeyType> *): void
+setRightPtr(right: BinaryNode<KeyType> *): void
+getIndex(): int
+getKey(): KeyType
+getLeftPtr(): BinaryNode<KeyType> *
+getRightPtr(): BinaryNode<KeyType> *
+isLeaf(): bool
ScreenManager
+printWelcome(): void
+buildDataStructure(filename: const string &, file: FileIO &): int
+searchManager(file: FileIO &): void
+insertManager(file: FileIO &): void
+deleteManager(file: FileIO &, stk: stack<Creature> &): void
+undoDeleteManager(file: FileIO &, stk: stack<Creature> &): void
+statisticsManager(file: FileIO &): void
+iDisplay(item: const string &, level: int): void
+hDisplay(item: const string &): void
HashTable
-capacity: int
-size: int
-table: vector<list<HashEntry>>
+Attribute1
-hashFunction(key: const string&): int
«constructor»+HashTable(Parameter1, initialCapacity: int)
+insert(creature: const Creature&): int
+getCreature(creatureID: const string&, idx: const int&): Creature
+remove(creatureID: const string&): bool
+getAllCreatures(): vector<Creature>
+loadFactor(): float
+getLongestChain(): int
+getEmptyBuckets(): int
+getSize(): int
+printTable(): void
«struct»
HashEntry
(from HashTable)
+key: string
+creature: Creature
«constructor»+HashEntry(k: const string&, c: const Creature&)
«namespace»
Color
(from ScreenManager)
+RESET
+RED
+GREEN
+YELLOW
+BLUE
+MAGENTA
+CYAN
+WHITE
+BOLD
data.txt
+read&write
+store
+includes