CUPERTINO PET STORE

Team #3:

Anita Huang, Sungjun An , Katie Leong, Juhwan Park, Gezheng Kang

Application Data:

A Pet class with the following member variables:

- Owner name primary key (string, e.g. Tom)
- Pet name secondary key (string, e.g. Jerry)
 Pet age (double, e.g. 2.5)
- Pet type (string, e.g. Cat)
- Pet weight(double, e.g. 4.3)

Hash Function

```
int key_tc_index {const Pet &key, int size {
    string k = key.getName();
    int sum = 0;
    for (int i = 0; k[i]; i++)
        sum += k[i];
    return sum % heapSize;
}
```

Comment

Our system stores data about a pet store and provides various applications, including inserting a set of data of a new pet, deleting a set of data of a pet, and searching a particular pet by its owner's name. In technical, in our project, both the binary search tree and hash table are ADT (abstract data types), which enhances the reusability of our data system.



UML Diagram HashNode Pet RinarySearchTree : RinaryTree HachTable -owner: string RootPtr: BinaryNode<ItemType>* -name: string - un (allisons (int) - hashsize (in+) Insert(&item): bool · Countrint - item (objet Pet ches) -type: string Search(&target.&returnedItem):bool - count (int) -age: double insert(*nodePtr.newNode): BinaryNode<ItemType> - hash Ary (Hash Mode doi) BinaryTree() -weight: double + Hash Midel search(*treePtr.target): RinaryNode<ItemType> BinaryTree(const BinaryTree<ItemType>&) + HainNode (item: Per day, occurs · "RinaryTree!) +Houtabb() tint, notollisens that +Pet(owner: string, name: string, type: string, age: double, weight: double) that table counting, harbiteint IsEmpty(): bool +setOwner(o: string) + set Item (anItem: pet al.) hashAru: Hashmacle asinter) GetCount():int +setName(n: string) BinaryNode Clear() + AHAMTAble () +setType(t: string) InOrder(void/*)(const &)) const + act (amt () : int +setWeight(w: double) PrintTree(void(*)(const & int)) const Item: ItemTyne + netItem(1: pet ob. +setAge(a: double) + met 1120 (1: int · Insert(const &): bool LeftPtr: RinaryNodecItemType>* +getOwner(): string + actorcupied () : int Search(const &. &): bool RightPtr: RinaryNodecItemType>* +getName(): string destrovTree(BinaryNode<ItemType>*): void + actibionisons(): int + is Empty (1: boo) +getType(): string RinaryNode(const |temTyne&) inorder(void(*)(const &, int),BinaryNode<(temType>*, int): + is Full ! bool + highest (ollidon(): inc BinaryNode/cont temTupe& EnaryNodestemTupes* SinaryNodestemTupes* +getAge(): double +getWeight(): double SetItem(const ItemType&) + invert Hash (Pet): bool printTree(void)*(const &, int).BinaryNode<(temType>*, int) SetLeftPtrl BinaryNodecItemType>* +onerators (n. Pet): hool + remove (Pet, string Key): bool +operatを持つ: Pet): bool SetRightPtrlBinaryNode<ItemType>* +operator== (p: Pet): bool GetItem(): ItemType + search (Pet, Hing key): int GetLeftPtr(): BinarvNode<ItemType>* + Builditash (HashTable dos) wid operator << (p: Pet) : ostream GetRightPtr(): RinaryNode<ItemType>*



