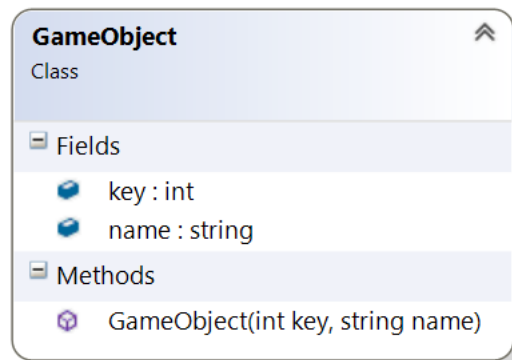


C++ Week 7 Hash Tables

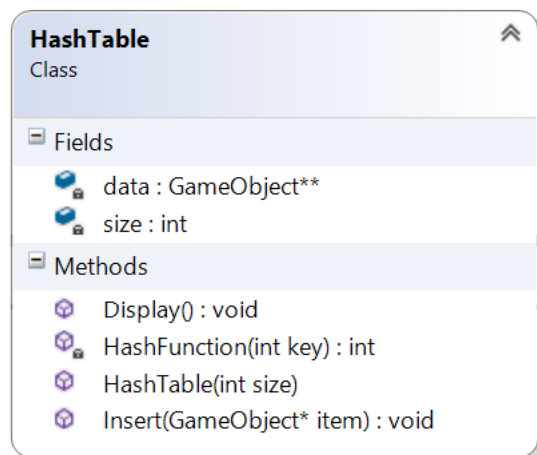
Hash Tables

This week we shall develop a Hash table that stores pointers to GameObjects.

1. Create a new VS C++ Project and name it Hash.
2. Define the class below



3. The Constructor parameters should be assigned to the corresponding data members.
4. Now define the HashTable class



C++ Week 7 Hash Tables

5. The constructor should
 - a. Assign the size parameter to the corresponding data member.
 - b. Instantiate the array size elements of pointers
 - c. Iterate through the array assigning each element a value of NULL
6. The Display member function iterates over each element of the array displaying the index, key and name. But only if the element is not NULL.
7. Add the Insert member function below

```
void HashTable::Insert(GameObject * item){  
    int key = item->key;  
    int hash = HashFunction(key);  
    while (data[hash] != NULL){  
        ++hash;  
        hash %= size;  
    }  
    data[hash] = item;  
}
```

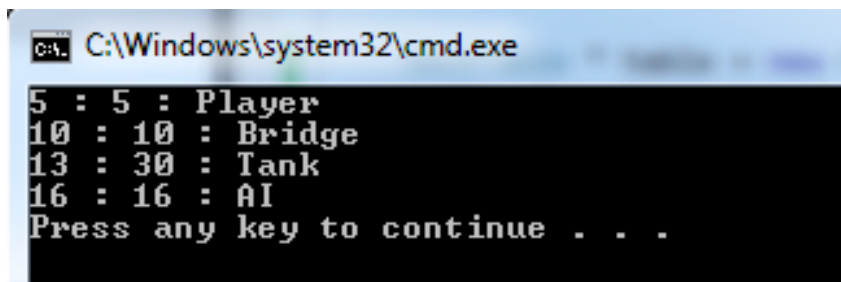
8. Now add the Hash function

```
int HashTable::HashFunction(int key){  
    return key % size;  
}
```

9. Within the main function instantiate some GameObjects and insert them into the HashTable. Finally display the contents within the Console.

```
HashTable * table = new HashTable(17);  
table->Insert(new GameObject(10,"Bridge"));  
table->Insert(new GameObject(5, "Player"));  
table->Insert(new GameObject(30, "Tank"));  
table->Insert(new GameObject(16, "AI"));  
table->Display();
```

10. Run the project. The output should resemble this



```
C:\Windows\system32\cmd.exe  
5 : 5 : Player  
10 : 10 : Bridge  
13 : 30 : Tank  
16 : 16 : AI  
Press any key to continue . . .
```

C++ Week 7 Hash Tables

Note that the key for the Tank was 30 and the array has only 17 elements! However the Hash Function has applied mod to the key to return 13 and this is the index in which the Tank has been stored.

Exercise 1

Insert another GameObject that causes a collision with the AI. Why was it placed at element 0?

11. Add the skeleton of the Find member function to the cpp and the declaration to the h file.

```
GameObject * HashTable::Find(int key){  
  
}
```

Exercise 2

Implement the pseudo code for Find

```
Define hash as type int and assign it the hash for the key  
Loop while data[hash] not null  
    If data[hash]'s key == key  
        Return data[hash]  
    End if  
    Increment hash  
    Set hash to hash mod size  
End loop  
Return null
```

Now try and locate and display the Tank

12. Add the skeleton of the Delete member function to the cpp and h file and the declaration to the h file.

```
GameObject * HashTable::Delete(int key){  
  
}
```

C++ Week 7 Hash Tables

Exercise 3

Implement the pseudo code for Delete

```
Define hash as type int and assign it the hash for the key
Loop while data[hash] not null
    If data[hash]'s key == key
        Define temp as pointer to GameObject and assign it data[hash]
        Set data[hash] to null
        Return temp
    End if
    Increment hash
    Set hash to hash mod size
End loop
Return null
```

13. Delete the tank and redisplay the contents of the HashTable to confirm it has been removed.

Exercise 4

The integer keys we've been using are meaningless. It would be far better to use a string that described the type of GameObject the key referred to.

Create a new class called HashTableStr that uses strings as key values. The key being the same as the name of the GameObject.

Create a new class called GameObjectStr which is similar to the previous version except the key is a string type.

The HashFunction should add the ASCII values of the key characters.

Within main create the same objects and perform the same operations on them.
Hint : the number of characters in a string is returned by the string's length() member function.