1310 PROGRAM 3 – HASH TABLES

DESCRIPTION

A hash table is a data structure that is used to store keys/value pairs. It is perfect to use when you have a large amount of directory-type information and the operations you need to perform are to insert, delete, print, and search.

I am giving you all a lot more freedom in this program in that **the value held in your hash table can be a pointer to any object created from your own custom class**.

Follow the specifications below to get full credit on this assignment.

FILES TO BE INCLUDED IN YOUR SUBMISSION

- **HashTable.h** – header file containing your HashTable class template.
- **HashEntry.h** – header file containing your HashEntry class template.
- **Classname.h** – the specification header file containing your class declaration (name it the same name as your class)
- **Classname.cpp** – the implementation file for your class (name it the same name as your class)
- **dataFile.txt –** a text file containing the data specific to the class you create. It should have enough data to create 10 objects. **(you get 10 points subtracted if you forget to do this!)**
- **Main.cpp** – contains the main function and any additional functions you created to make the program run
- **Makefile** – you must create a Makefile to easily compile your program **(you get 15 points subtracted if you forget to do this!)**

MAIN PROGRAM (MAIN.CPP)

- Must contain the main function as well as four other functions.
- The main function should DYNAMICALLY create a new HashTable object of size 10, which should be created with the template type being a pointer to an object of your custom class.
  `HashTable<YourCustomClass*> *myTable = new HashTable<YourCustomClass*>(10);`
- You should have a menu of options to add data from a file, add one object manually, search for an object, delete an object, print the hash table, or end the program.
- Validate the user's choice.
- Call the appropriate function depending on their choice.
  - You should have one function that will read in data from the file, DYNAMICALLY create an object of your custom class type, and then add this object to the hash table.
  - You should have another function that will allow the user to enter in an object's data manually and then create an object from this data, and then add to the hash table.
  - The third function should allow the user to search for an object in the hash table.
  - The forth function should allow a user to delete an object from the hash table.

YOUR CUSTOM CLASS (CLASSNAME.H & CLASSNAME.CPP)

Whatever you choose for your class, your objects created from this class **MUST** have an integer **idNum**, (which will be used as the key in the hash table) and **at least two other attributes**. You should write appropriate functions for this class that you will need. You should also have a function that will overload the **<<** (insertion operator) so that the HashTable class can print out your object data in the way that you want.

HASH TABLE TEMPLATE CLASS (HASHTABLE.H)

Your **HashTable** class must be a template class. The attributes include an integer table size and a pointer to an array of pointers to HashEntry<T>.

You must implement the following functions for **HashTable** class:

- **Constructor** – dynamically allocate the hash table of the parameter size and initialize other variables as necessary.
- **Destructor** – release memory that was dynamically allocated in the HashTable class
- **T getValue(int key)** – send in a key, return the value in the hash table at that key
- **void putValue(int key, T value)** – inserts into the hash table using the hash function `key%10`.
- **void removeValue(int key)** – removes a value at the given key in the hash table
- **void printHashTable()** – prints out the hash table in a way that is easy to see which bucket objects were placed in and also indicates which buckets are still NULL

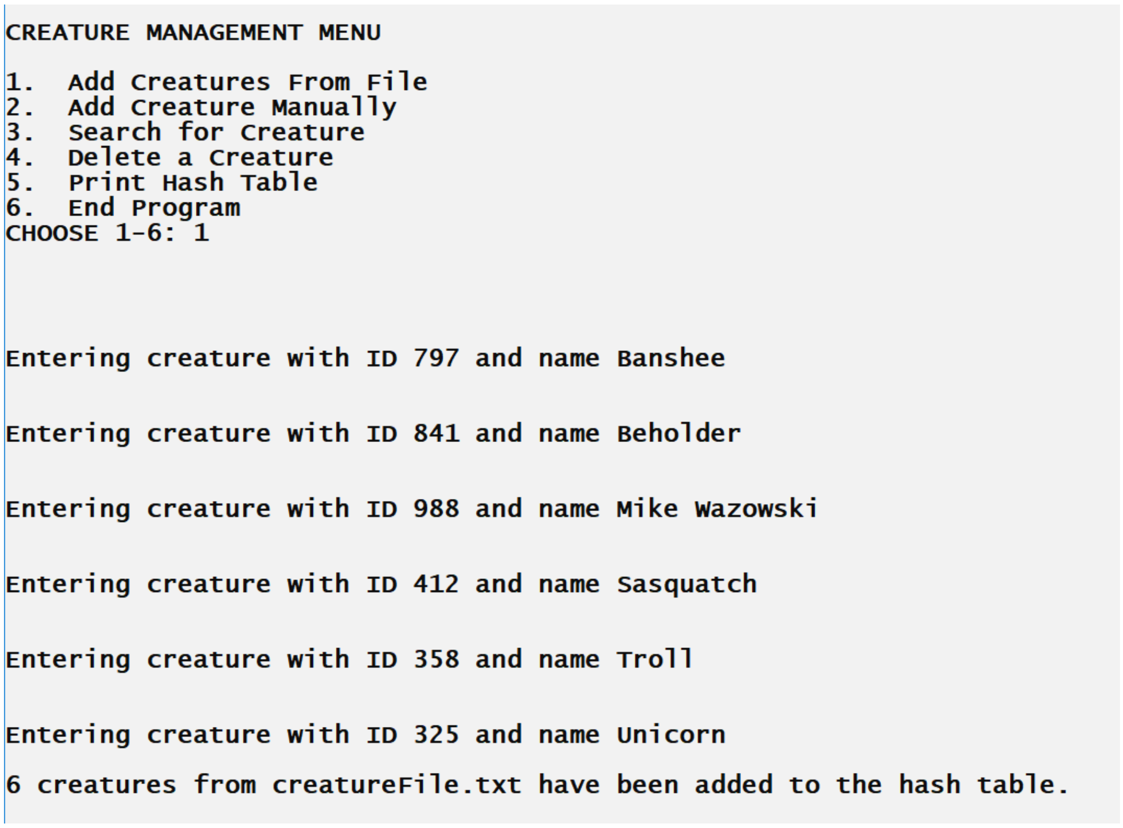
HASH ENTRY TEMPLATE CLASS (HASHENTRY.H)

Your **HashEntry** class must be a template class, which describes an entry into the hash table. The attributes include an integer key, and the value is of the template type. Also, since the HashTable class will be using chaining, then you will also have a pointer to the next HashEntry as an attribute.

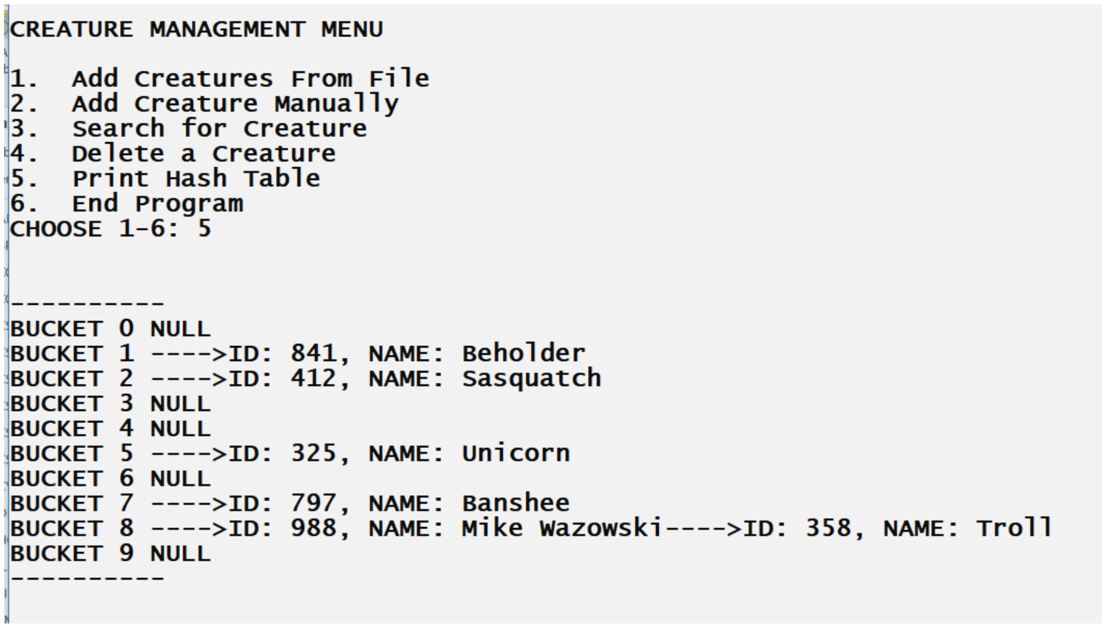
You must implement the following functions for HashEntry class:

- **Constructor** – initializes the key, value, and next pointer. The key & value is sent to the constructor
- **Getter functions** (to get the key, to get the value, to get next pointer value)
- **Setter functions** (setNext & setValue)

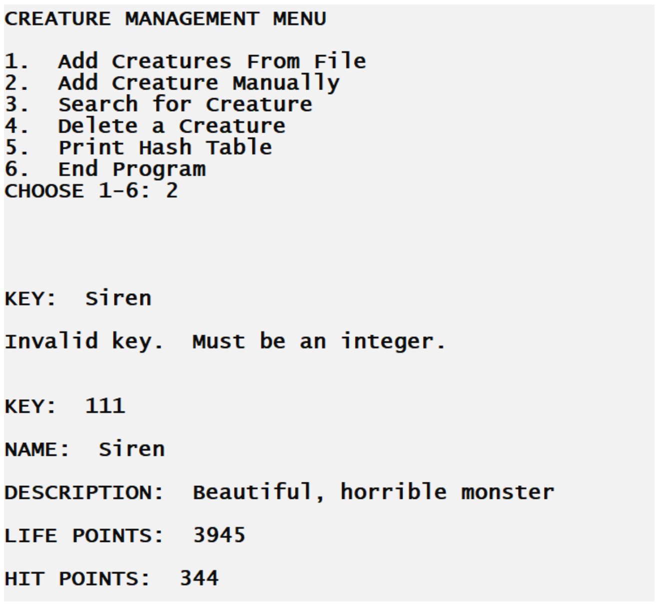
SAMPLE OUTPUT (I USED THE CREATURE CLASS FOR MY CLASS)



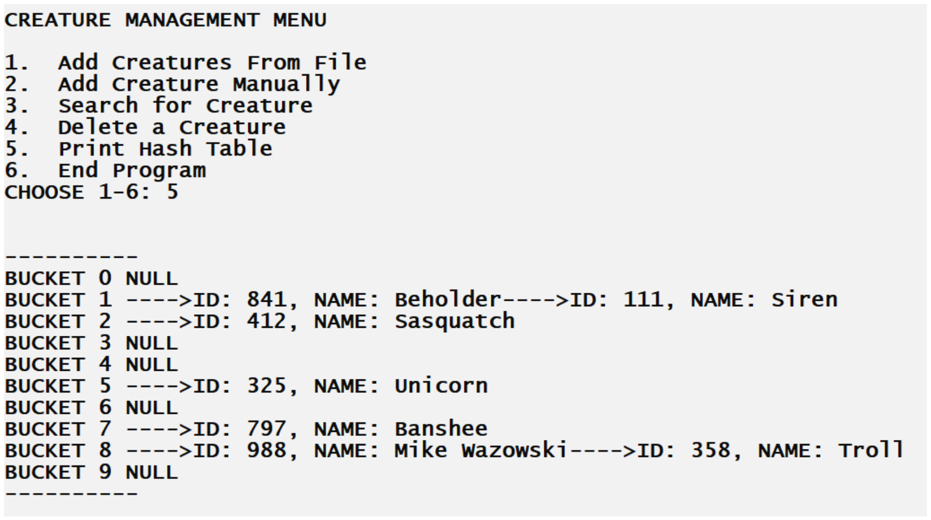
```
CREATURE MANAGEMENT MENU

1.   Add Creatures From File
2.   Add Creature Manually
3.   Search for Creature
4.   Delete a Creature
5.   Print Hash Table
6.   End Program
CHOOSE 1-6: 1



Entering creature with ID 797 and name Banshee


Entering creature with ID 841 and name Beholder


Entering creature with ID 988 and name Mike Wazowski


Entering creature with ID 412 and name Sasquatch


Entering creature with ID 358 and name Troll


Entering creature with ID 325 and name Unicorn

6 creatures from creatureFile.txt have been added to the hash table.
```



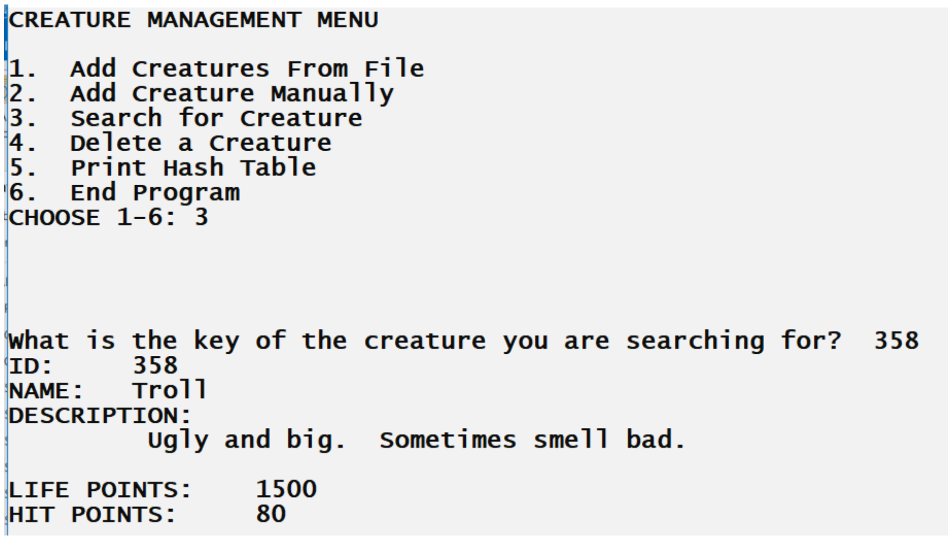
```
CREATURE MANAGEMENT MENU

1.   Add Creatures From File
2.   Add Creature Manually
3.   Search for Creature
4.   Delete a Creature
5.   Print Hash Table
6.   End Program
CHOOSE 1-6: 5


----------
BUCKET 0 NULL
BUCKET 1 ---->ID: 841, NAME: Beholder
BUCKET 2 ---->ID: 412, NAME: Sasquatch
BUCKET 3 NULL
BUCKET 4 NULL
BUCKET 5 ---->ID: 325, NAME: Unicorn
BUCKET 6 NULL
BUCKET 7 ---->ID: 797, NAME: Banshee
BUCKET 8 ---->ID: 988, NAME: Mike Wazowski---->ID: 358, NAME: Troll
BUCKET 9 NULL
----------
```



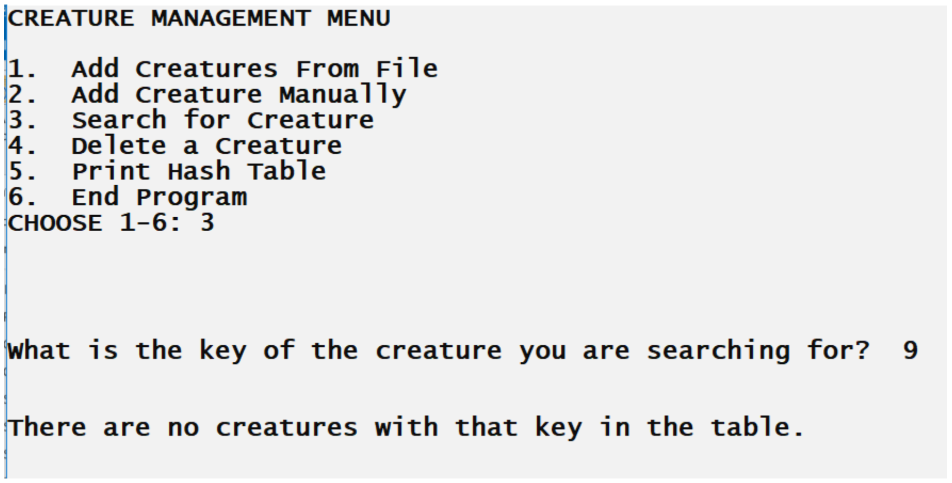
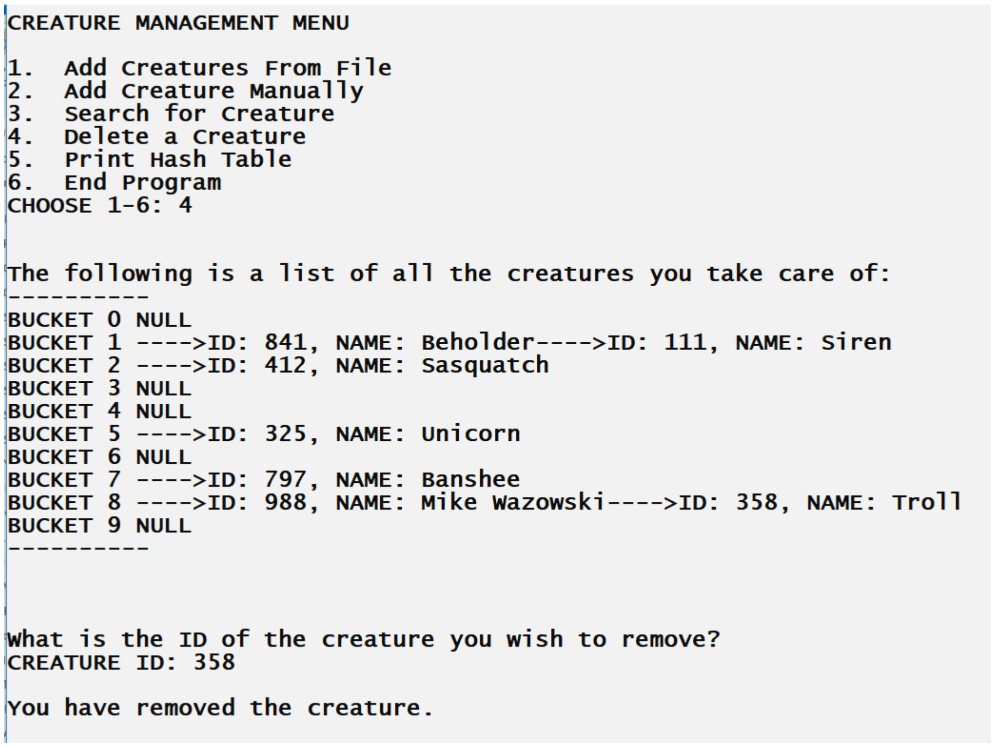
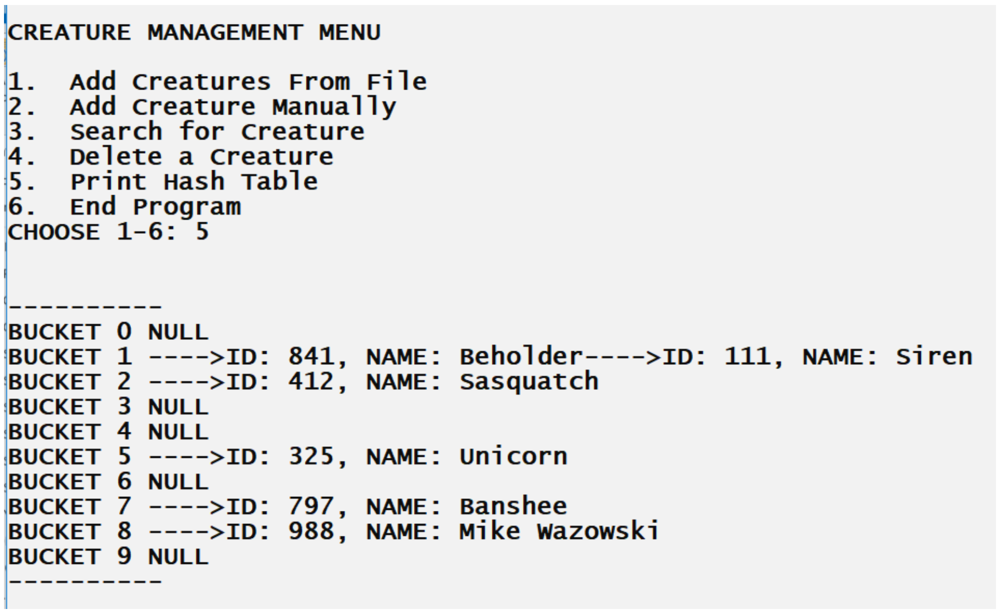
```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 2




KEY:  Siren

Invalid key.  Must be an integer.


KEY:  111

NAME:  Siren

DESCRIPTION:  Beautiful, horrible monster

LIFE POINTS:  3945

HIT POINTS:  344
```

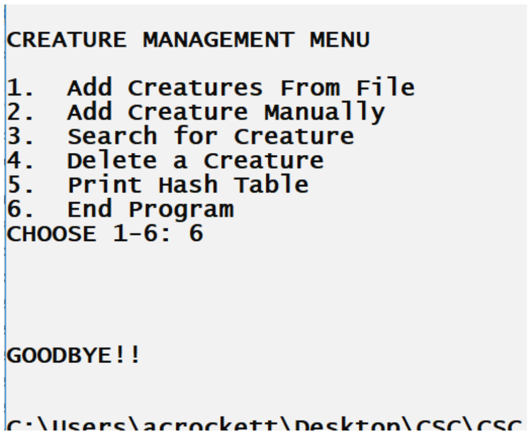


```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 5


----------
BUCKET 0 NULL
BUCKET 1 ---->ID: 841, NAME: Beholder---->ID: 111, NAME: Siren
BUCKET 2 ---->ID: 412, NAME: Sasquatch
BUCKET 3 NULL
BUCKET 4 NULL
BUCKET 5 ---->ID: 325, NAME: Unicorn
BUCKET 6 NULL
BUCKET 7 ---->ID: 797, NAME: Banshee
BUCKET 8 ---->ID: 988, NAME: Mike Wazowski---->ID: 358, NAME: Troll
BUCKET 9 NULL
----------
```



```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 3




What is the key of the creature you are searching for?  358
ID:     358
NAME:   Troll
DESCRIPTION:
         Ugly and big.  Sometimes smell bad.

LIFE POINTS:    1500
HIT POINTS:     80
```


```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 3




What is the key of the creature you are searching for?  9


There are no creatures with that key in the table.
```

```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 4


The following is a list of all the creatures you take care of:
----------
BUCKET 0 NULL
BUCKET 1 ---->ID: 841, NAME: Beholder---->ID: 111, NAME: Siren
BUCKET 2 ---->ID: 412, NAME: Sasquatch
BUCKET 3 NULL
BUCKET 4 NULL
BUCKET 5 ---->ID: 325, NAME: Unicorn
BUCKET 6 NULL
BUCKET 7 ---->ID: 797, NAME: Banshee
BUCKET 8 ---->ID: 988, NAME: Mike Wazowski---->ID: 358, NAME: Troll
BUCKET 9 NULL
----------



What is the ID of the creature you wish to remove?
CREATURE ID: 358

You have removed the creature.
```

```
CREATURE MANAGEMENT MENU

1.  Add Creatures From File
2.  Add Creature Manually
3.  Search for Creature
4.  Delete a Creature
5.  Print Hash Table
6.  End Program
CHOOSE 1-6: 5


----------
BUCKET 0 NULL
BUCKET 1 ---->ID: 841, NAME: Beholder---->ID: 111, NAME: Siren
BUCKET 2 ---->ID: 412, NAME: Sasquatch
BUCKET 3 NULL
BUCKET 4 NULL
BUCKET 5 ---->ID: 325, NAME: Unicorn
BUCKET 6 NULL
BUCKET 7 ---->ID: 797, NAME: Banshee
BUCKET 8 ---->ID: 988, NAME: Mike Wazowski
BUCKET 9 NULL
----------
```



```
CREATURE MANAGEMENT MENU

1.   Add Creatures From File
2.   Add Creature Manually
3.   Search for Creature
4.   Delete a Creature
5.   Print Hash Table
6.   End Program
CHOOSE 1-6: 6




GOODBYE!!


C:\Users\acrockett\Desktop\CSC\CSC
```