Contact List Hash Table

This program uses a hash table structure to manage a list of names and phone numbers, which are the keys and values, by utilizing specific functions that add the contacts to the table as well as pull up each contact’s information. The language used was C++ to write the program including the built-in hash function object class in C++ to calculate where in the table each contact is stored. Also, the node and hash table classes are created using generic templates which allows them to store various types of data making it more convenient for the user.

|  |  |  |  |
| --- | --- | --- | --- |
| Class/Function Name | Parameters | Return Type | Explanation |
| **HNode( )** | None | None | This creates a new Node constructor that contains no data |
| **HNode(a)** | Typename K | None | This creates a new hash Node that contains data specified by what is entered in as an argument as an initialized key |
| **HNode(a,b)** | Typename K & V | None | This creates a new hash Node that contains data specified by what is entered in as an argument as an initialized key & value |
| **GetKey** | None | Typename K | Returns the key of node that calls it. |
| **GetValue** | None | Typename V | Returns the value of node that calls it. |

Classes

**Node**

**Htable**

|  |  |  |  |
| --- | --- | --- | --- |
| Class/Function Name | Parameters | Return Type | Explanation |
| **Htable( )** | None | None | This creates a new Htable constructor that contains an empty hash table array of type K & V. |
| **Htable( )** | None | None | This creates an empty constructor.. |
| **HInsert(a,b)** | Key & Value | Void | Creates a new node with the key-value pair inserted & uses the hash algorithm & key to calculate what index the new pair belongs in. Then it adds the new node to that index |
| **Int HashKey(a)** | Key | Int | Takes in key as argument and applies the C++ built in hash function to get a number. Mods this number by the size of the table and returns it. |
| **Retrieve(a)** | Key | Key | First uses HashKey( ) to figure out where desired key is located. Then checks if key at the index matches desired key. If so, prints out the matching key & its value |

**Main/Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Main( )** | None | Void | Main function that begins execution of the entire program. It also starts the menu loop. |
| **Menu** | N/A | N/A | Uses a do While loop to offer the user options for executing the testing suite. When user chooses an option, certain functions are automatically ran to test for a full range of conditions including creating new key value pairs, inserting pairs into the table and retrieving pairs. |