Rafaela Lomboy
Assignment 1 - Hashing

CLASS AND ALGORITHM DESIGNS
Includes class designs for **Slot, Bucket,** and **Hashtable**
and algorithm designs for **Search** and **Insertion**

INCLUDE iostream, string, vector libraries (Bucket included in MAIN program)

class Hashtable (template class of **Buckets**)
  Member Variables
  + const int CAP (primary buckets)
  + int nextOFBucket
  - vector<Bucket> buckets (initialize size to CAP in constructor)

  Default constructor
  + Hashtable()

  Class Methods
  + void insertToHT(string key, string data, int index)
  + void insertToHT(string key, string key)
  + void GenStatReport()
  + void hashRecords(string dataFileName)
  + void writeHTtoDisk()
  + void restoreHTtoMem()
  + void ReportHT(string reportFileName, string reportName)
  + void SearchHT(string searchFileName)
  + void collisionReport()

  Helper functions
  - int hashFunc(string key) const
  - void findRecord(string key, bool& found, int& slot, int& index)

endclass

Insertion Method Pseudocode
(Member of template class Hashtable<Bucket>)
Takes a key and data pair, hashes key to an index in hashtable, then inserts into table.

void **insertToHT** (string key, string data, int index) - do NOT hash index!

```
void insertToHT (string key, string data)
        Initialize slot index to 0
        Initialize bucket index to 0

        CALL hashFunc(key) and SET bucket index to result
```

```
    IF bucket[index] has a free slot
        SET key and data of buckets[index] at slot
        IF slot NOT 0, INCREMENT buckets[index] collisions counter
    ELSE IF bucket[index] points to overflow bucket
        INCREMENT buckets[index] collisions counter
        SET index to overflow bucket index
        CALL insertToHT(key, data, index)
    ELSE
        INCREMENT buckets[index] collisions counter

        Create new overflow bucket
        Allocate space in buckets and add new bucket

        SET buckets[index] overflow pointer to overflow bucket index

        SET index to overflow bucket index
        CALL insertToHT(key, data, index)
    ENDIF
END insertToHT method
```

Search Algorithm Pseudocode
(Member of template class Hashtable<Bucket>)
Takes an index from a hashed key and determines bucket index and slot of item if already present in table (looks for EXACT match)
Note: hash function is called before this method!

```
void findRecord(string key, bool& found, int& slot, int& index)
    Initialize slot index to 0
    SET found to false
    Initialize current key

    LOOP until all slots searched OR found
        SET current key to key of buckets[index] at slot index

        IF current key EQUALS key of buckets[index] at slot (EXACT match)
            SET found to true
            SET slot to slot index

        INCREMENT  slot index
    ENDLOOP

    IF NOT found AND overflow bucket is allocated AND index < buckets vector size
        SET index to overflow pointer of buckets[index]
        CALL findRecord(key, found, slot, index)
        ENDIF
END findRecord method


INCLUDE Slot header, iostream and string

class Bucket
   Member Variables
    +   const int numSlots
    +   int nextOpenSlot
    +   int count    (collision count)
    +   int OFindex
    +   Slot slots[numSlots]
  Constructors
    +   Bucket()    (default)
    +   Bucket(string key, string data)    (parametrized)
    +   Bucket(const Bucket& rhs)    (copy)
  Class Methods
    +   void set(string key, string data, int slotIndex)
    +   string getKey(int slotIndex) const
    +   string getData(int slotIndex) const
    +   void setKey(string key, int slotIndex)
    +   void setData(string data, int slotIndex)
    +   bool hasFreeSlot()
    +   bool isOpen(int sIndex)
    +   Bucket& operator=(const Bucket& rhs)     (copy assignment)
endclass


INCLUDE string library

class Slot
   Member Variables
    -   string key
    -   string data
   Constructors
    +   Slot()  (default)
    +   Slot(const Slot& rhs)    (copy)
   Class Methods
    +   void set(string key, string data)
    +   string getKey() const
    +   string getData() const
    +   void setKey(string key)
    +   void setData(string data)
```

```
endclass
```