Assignment 1

• Symbol Table
• Implement the functions define in SymTab.h in a file called SymTab.c
• Implement a driver program to test your implementation
• “Due” September 16
SymTab.h

/* The symbol table structure proper. Implemented as a hash table that uses separate chaining to resolve collisions Contents is dynamically allocated according to size */

struct SymTab { int Size;
    struct SymEntry **Contents;
};

/* The Name/Attributes association structure used in the symbol tables linked lists. */

struct SymEntry { char *Name;
    void *Attributes;
    struct SymEntry *Next;
};
SymTab.h

/*  CreateSymTab:    create and return a reference to a symbol table of
    approximately Size many entries.

    DestroySymTab:   destroy all storage associated with a Symbol Table
    Which is under the table's control. This does not include the
    attributes
*/

struct SymTab *   CreateSymTab(int Size);

void   DestroySymTab(struct SymTab *ATable);
/* EnterName: enter a Name into a symbol table. Passes back an argument containing an entry reference for the name. Return 1 if the name was not already in the symbol table, otherwise return 0. EnterName must allocate space for the Name and copy the contents the parameter Name.

FindName: find a Name in a symbol table. Return an entry reference or NULL depending on whether the Name was found.
*/

int EnterName(struct SymTab *ATable,
              const char *Name,
              struct SymEntry * *AnEntry);

struct SymEntry * FindName(struct SymTab *ATable, const char * Name);
SymTab.h

/*    SetAttr:    set the attribute pointer associated with an entry.
    GetAttr:    get the attribute pointer associated with an entry.
    GetName:    get the name string associated with an entry.
*/

void  SetAttr(struct SymEntry *AnEntry, void *Attributes);

void *  GetAttr(struct SymEntry *AnEntry);

const char *  GetName(struct SymEntry *AnEntry);
/* These two functions can be used to enumerate the contents of a table. The enumeration order is arbitrary.

FirstEntry: return the "first" entry in a symbol table or NULL if the table is empty. "First" does not imply a particular order (e.g. alphabetical) it is simply the order found in the table.

NextEntry: return the next entry after the supplied entry or NULL if no more entries.
*/

struct SymEntry * FirstEntry(struct SymTab *ATable);
struct SymEntry * NextEntry(struct SymTab *ATable, struct SymEntry *AnEntry);