```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef union {
    int intValue;
    float floatValue;
    char charValue;
} Value;

typedef struct {
    char name[30];
    char dataType[30];
    char scope[20];
    Value value;
} Symbol;

void add(Symbol *symbols, int n);
void display(Symbol *symbols, int n);
void search(Symbol *symbols, int n);
void countIdentifiers(Symbol *symbols, int n);
void deleteSymbol(Symbol *symbols, int *n);
void sortSymbolsByName(Symbol *symbols, int n);
void sortSymbolsByScope(Symbol *symbols, int n);

int main() {
    int n, choice;

    printf("Enter the number of symbols: ");
    scanf("%d", &n);

    Symbol *symbols = (Symbol *)malloc(n * sizeof(Symbol));

    if (symbols == NULL) {
        printf("Memory allocation failed.\n");
        return 1;
    }

    do {
        printf("\nMenu Options:\n");
        printf("1. Add Symbols\n");
        printf("2. Display All Symbols\n");
        printf("3. Search for a Symbol by Name\n");
        printf("4. Count Identifiers by Data Type or Scope\n");
        printf("5. Remove a Symbol by Name\n");
        printf("6. Sort Symbols by Name\n");
        printf("7. Sort Symbols by Scope\n");
        printf("8. Exit\n");
```

```
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                add(symbols, n);
                break;
            case 2:
                display(symbols, n);
                break;
            case 3:
                search(symbols, n);
                break;
            case 4:
                countIdentifiers(symbols, n);
                break;
            case 5:
                deleteSymbol(symbols, &n);
                break;
            case 6:
                sortSymbolsByName(symbols, n);
                break;
            case 7:
                sortSymbolsByScope(symbols, n);
                break;
            case 8:
                printf("Exiting the program.\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 8);

    free(symbols);
    return 0;
}

void add(Symbol *symbols, int n) {
    for (int i = 0; i < n; i++) {
        printf("Enter name for symbol %d: ", i + 1);
        scanf("%s", symbols[i].name);
        printf("Enter data type (int/float/char): ");
        scanf("%s", symbols[i].dataType);
        printf("Enter scope (global/local): ");
        scanf("%s", symbols[i].scope);

        if (strcmp(symbols[i].dataType, "int") == 0) {
            printf("Enter integer value: ");
```

```
            scanf("%d", &symbols[i].value.intValue);
        } else if (strcmp(symbols[i].dataType, "float") == 0) {
            printf("Enter float value: ");
            scanf("%f", &symbols[i].value.floatValue);
        } else if (strcmp(symbols[i].dataType, "char") == 0) {
            printf("Enter character value: ");
            getchar(); // Consume newline left in buffer
            scanf("%c", &symbols[i].value.charValue);
        } else {
            printf("Invalid data type!\n");
        }
    }
}

void display(Symbol *symbols, int n) {
    printf("\nSymbol Table:\n");
    for (int i = 0; i < n; i++) {
        printf("Name: %s, DataType: %s, Scope: %s, Value: ",
               symbols[i].name, symbols[i].dataType, symbols[i].scope);
        if (strcmp(symbols[i].dataType, "int") == 0) {
            printf("%d\n", symbols[i].value.intValue);
        } else if (strcmp(symbols[i].dataType, "float") == 0) {
            printf("%f\n", symbols[i].value.floatValue);
        } else if (strcmp(symbols[i].dataType, "char") == 0) {
            printf("%c\n", symbols[i].value.charValue);
        }
    }
}

void search(Symbol *symbols, int n) {
    char name[30];
    printf("Enter the name to search: ");
    scanf("%s", name);

    for (int i = 0; i < n; i++) {
        if (strcmp(symbols[i].name, name) == 0) {
            printf("Symbol Found:\n");
            printf("Name: %s, DataType: %s, Scope: %s, Value: ",
                   symbols[i].name, symbols[i].dataType, symbols[i].scope);
            if (strcmp(symbols[i].dataType, "int") == 0) {
                printf("%d\n", symbols[i].value.intValue);
            } else if (strcmp(symbols[i].dataType, "float") == 0) {
                printf("%f\n", symbols[i].value.floatValue);
            } else if (strcmp(symbols[i].dataType, "char") == 0) {
                printf("%c\n", symbols[i].value.charValue);
            }
            return;
        }
```

```
    }
    printf("Symbol not found.\n");
}

void countIdentifiers(Symbol *symbols, int n) {
    int intCount = 0, floatCount = 0, charCount = 0;
    int globalCount = 0, localCount = 0;

    for (int i = 0; i < n; i++) {
        if (strcmp(symbols[i].dataType, "int") == 0) intCount++;
        else if (strcmp(symbols[i].dataType, "float") == 0) floatCount++;
        else if (strcmp(symbols[i].dataType, "char") == 0) charCount++;

        if (strcmp(symbols[i].scope, "global") == 0) globalCount++;
        else if (strcmp(symbols[i].scope, "local") == 0) localCount++;
    }

    printf("Data Type Counts: int = %d, float = %d, char = %d\n", intCount, floatCount,
charCount);
    printf("Scope Counts: global = %d, local = %d\n", globalCount, localCount);
}

void deleteSymbol(Symbol *symbols, int *n) {
    char name[30];
    printf("Enter the name of the symbol to delete: ");
    scanf("%s", name);

    for (int i = 0; i < *n; i++) {
        if (strcmp(symbols[i].name, name) == 0) {
            for (int j = i; j < *n - 1; j++) {
                symbols[j] = symbols[j + 1];
            }
            (*n)--;
            printf("Symbol deleted successfully.\n");
            return;
        }
    }
    printf("Symbol not found.\n");
}

void sortSymbolsByName(Symbol *symbols, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (strcmp(symbols[i].name, symbols[j].name) > 0) {
                Symbol temp = symbols[i];
                symbols[i] = symbols[j];
                symbols[j] = temp;
            }
```

```
        }
    }
    printf("Symbols sorted by name.\n");
}

void sortSymbolsByScope(Symbol *symbols, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (strcmp(symbols[i].scope, symbols[j].scope) > 0) {
                Symbol temp = symbols[i];
                symbols[i] = symbols[j];
                symbols[j] = temp;
            }
        }
    }
    printf("Symbols sorted by scope.\n");
}
```