11. Implement a C program to perform symbol table operations.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 100

typedef struct {

char name[50];

char type[20];

int address;

} Symbol;

typedef struct {

Symbol table[MAX];

int count;

} SymbolTable;

void initSymbolTable(SymbolTable \*symTab) {

symTab->count = 0;

}

void insertSymbol(SymbolTable \*symTab, char \*name, char \*type, int address) {

if (symTab->count < MAX) {

strcpy(symTab->table[symTab->count].name, name);

strcpy(symTab->table[symTab->count].type, type);

symTab->table[symTab->count].address = address;

symTab->count++;

printf("Symbol inserted: %s, %s, %d\n", name, type, address);

} else {

printf("Symbol table is full!\n");

}

}

int searchSymbol(SymbolTable \*symTab, char \*name) {

for (int i = 0; i < symTab->count; i++) {

if (strcmp(symTab->table[i].name, name) == 0) {

return i; // Found, return index

}

}

return -1; // Not found

}

// Display symbol table

void displaySymbolTable(SymbolTable \*symTab) {

printf("\nSymbol Table:\n");

printf("--------------------------------\n");

printf("Name\tType\tAddress\n");

printf("--------------------------------\n");

for (int i = 0; i < symTab->count; i++) {

printf("%s\t%s\t%d\n", symTab->table[i].name, symTab->table[i].type, symTab->table[i].address);

}

printf("--------------------------------\n");

}

int main() {

SymbolTable symTab;

initSymbolTable(&symTab);

int choice, address;

char name[50], type[20];

while (1) {

printf("\n1. Insert Symbol\n");

printf("2. Display Symbol Table\n");

printf("3. Search Symbol\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter name: ");

scanf("%s", name);

printf("Enter type: ");

scanf("%s", type);

printf("Enter address: ");

scanf("%d", &address);

insertSymbol(&symTab, name, type, address);

break;

case 2:

displaySymbolTable(&symTab);

break;

case 3:

printf("Enter symbol name to search: ");

scanf("%s", name);

int index = searchSymbol(&symTab, name);

if (index != -1) {

printf("Symbol found at index %d\n", index);

} else {

printf("Symbol not found!\n");

}

break;

case 4:

exit(0);

default:

printf("Invalid choice! Try again.\n");

}

}

return 0;

}

