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

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define SIZE 100

typedef struct Symbol {

char name[50];

int value;

struct Symbol\* next;

} Symbol;

Symbol\* hashTable[SIZE];

unsigned int hash(char\* name) {

unsigned int hashValue = 0;

while (\*name) {

hashValue = (hashValue << 5) + \*name++;

}

return hashValue % SIZE;

}

void insert(char\* name, int value) {

unsigned int index = hash(name);

Symbol\* newSymbol = (Symbol\*)malloc(sizeof(Symbol));

strcpy(newSymbol->name, name);

newSymbol->value = value;

newSymbol->next = hashTable[index];

hashTable[index] = newSymbol;

printf("Inserted: %s -> %d\n", name, value);

}

Symbol\* search(char\* name) {

unsigned int index = hash(name);

Symbol\* current = hashTable[index];

while (current) {

if (strcmp(current->name, name) == 0) {

return current;

}

current = current->next;

}

return NULL;

}

void delete(char\* name) {

unsigned int index = hash(name);

Symbol\* current = hashTable[index];

Symbol\* prev = NULL;

while (current) {

if (strcmp(current->name, name) == 0) {

if (prev) {

prev->next = current->next;

} else {

hashTable[index] = current->next;

}

free(current);

printf("Deleted: %s\n", name);

return;

}

prev = current;

current = current->next;

}

printf("Symbol not found: %s\n", name);

}

int main() {

insert("x", 10);

insert("y", 20);

Symbol\* symbol = search("x");

if (symbol) {

printf("Found: %s -> %d\n", symbol->name, symbol->value);

} else {

printf("Symbol not found: x\n");

}

delete("x");

symbol = search("x");

if (symbol) {

printf("Found: %s -> %d\n", symbol->name, symbol->value);

} else {

printf("Symbol not found: x\n");

}

return 0;

}

