1. Ganesh sesha sai Akhil

AP21110010931

CSE - 0

1. Hash implementation

Code :

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define HASH\_TABLE\_SIZE 100

struct Variable {

char name[50];

char data\_type[50];

char size[50];

char dimensions[50];

char address[50];

struct Variable\* next;

};

struct Variable\* hash\_table[HASH\_TABLE\_SIZE];

unsigned int hash(const char\* name) {

unsigned int hash\_value = 0;

while (\*name) {

hash\_value = (hash\_value << 5) + \*name;

name++;

}

return hash\_value % HASH\_TABLE\_SIZE;

}

void add\_variable(char name[], char data\_type[], char size[], char dimensions[], char address[]) {

struct Variable\* new\_variable = (struct Variable\*)malloc(sizeof(struct Variable));

strcpy(new\_variable->name, name);

strcpy(new\_variable->data\_type, data\_type);

strcpy(new\_variable->size, size);

strcpy(new\_variable->dimensions, dimensions);

strcpy(new\_variable->address, address);

new\_variable->next = NULL;

unsigned int index = hash(name);

new\_variable->next = hash\_table[index];

hash\_table[index] = new\_variable;

}

void display\_symbol\_table() {

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

printf("| %-20s | %-20s | %-20s | %-20s | %-20s |\n", "Name", "Type", "Size", "Dimensions", "Address");

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

for (int i = 0; i < HASH\_TABLE\_SIZE; i++) {

struct Variable\* current = hash\_table[i];

while (current != NULL) {

printf("| %-20s | %-20s | %-20s | %-20s | %-20s |\n",

current->name, current->data\_type, current->size, current->dimensions, current->address);

current = current->next;

}

}

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

}

int main() {

char choice;

char name[50], data\_type[50], size[50], dimensions[50], address[50];

while (1) {

printf("\nEnter '1' to add a variable, '2' to display the symbol table, or 'q' to quit: ");

scanf(" %c", &choice);

switch (choice) {

case '1':

printf("\nEnter the variable name: ");

scanf("%s", name);

printf("Enter the type (datatype, function, or array): ");

scanf("%s", data\_type);

printf("Enter the size (if applicable): ");

scanf("%s", size);

if (strcmp(data\_type, "array") == 0) {

printf("Enter the dimensions (if it's an array, comma-separated): ");

scanf("%s", dimensions);

} else {

strcpy(dimensions, "-");

}

printf("Enter the address: ");

scanf("%s", address);

add\_variable(name, data\_type, size, dimensions, address);

printf("Variable added to symbol table.\n");

break;

case '2':

display\_symbol\_table();

break;

case 'q':

for (int i = 0; i < HASH\_TABLE\_SIZE; i++) {

struct Variable\* current = hash\_table[i];

while (current != NULL) {

struct Variable\* temp = current;

current = current->next;

free(temp);

}

}

return 0;

default:

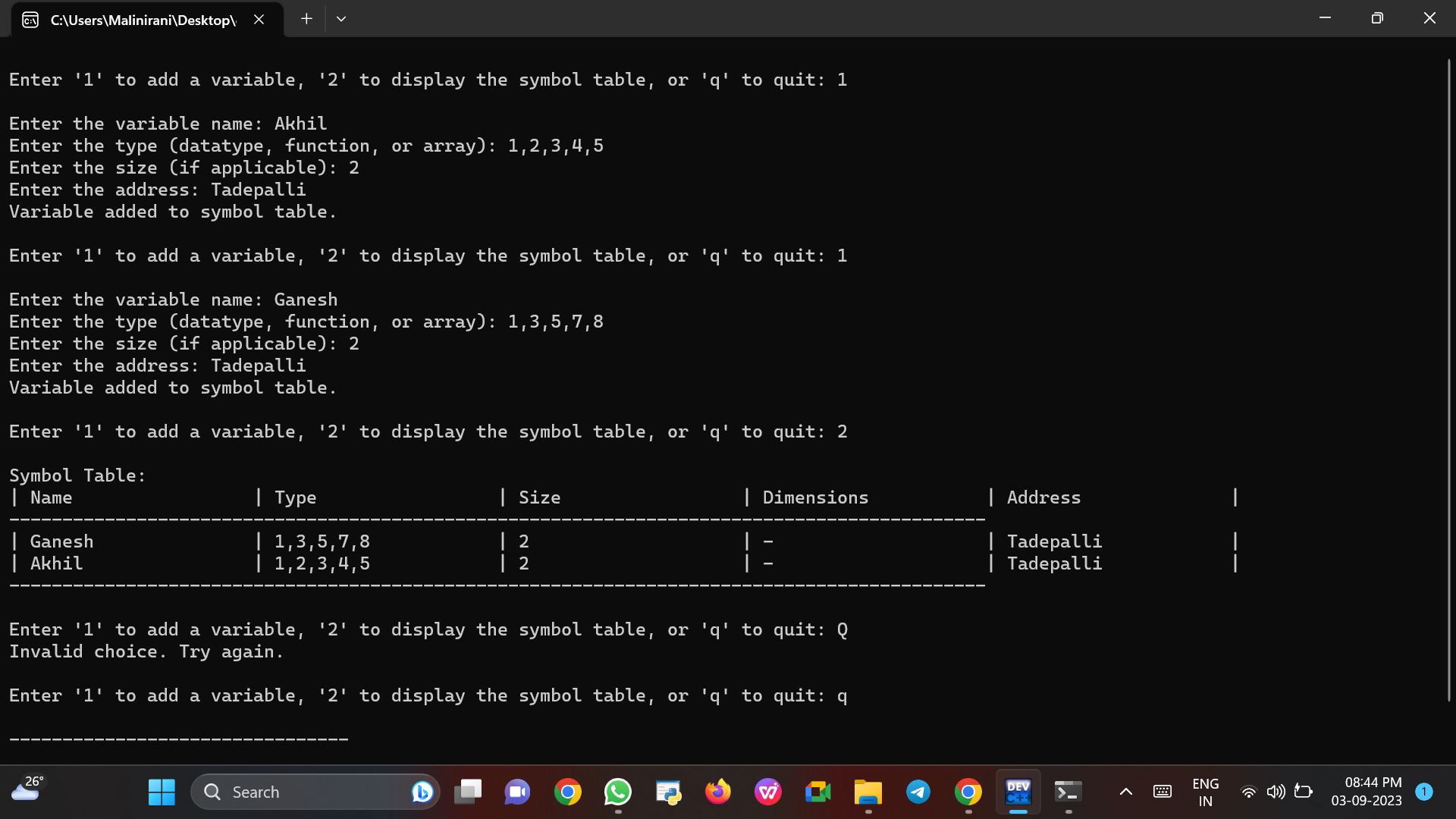
printf("Invalid choice. Try again.\n");

}

}

}

Output:



1. Linked List implementation

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Variable {

char name[50];

char data\_type[50];

char size[50];

char dimensions[50];

char address[50];

struct Variable\* next;

};

struct Variable\* head = NULL;

void add\_variable(char name[], char data\_type[], char size[], char dimensions[], char address[]) {

struct Variable\* new\_variable = (struct Variable\*)malloc(sizeof(struct Variable));

strcpy(new\_variable->name, name);

strcpy(new\_variable->data\_type, data\_type);

strcpy(new\_variable->size, size);

strcpy(new\_variable->dimensions, dimensions);

strcpy(new\_variable->address, address);

new\_variable->next = head;

head = new\_variable;

}

void display\_symbol\_table() {

struct Variable\* current = head;

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

printf("| %-20s | %-20s | %-20s | %-20s | %-20s |\n", "Name", "Type", "Size", "Dimensions", "Address");

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

while (current != NULL) {

printf("| %-20s | %-20s | %-20s | %-20s | %-20s |\n",

current->name, current->data\_type, current->size, current->dimensions, current->address);

current = current->next;

}

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

}

int main() {

char choice;

char name[50], data\_type[50], size[50], dimensions[50], address[50];

while (1) {

printf("\nEnter '1' to add a variable, '2' to display the symbol table, or 'q' to quit: ");

scanf(" %c", &choice);

switch (choice) {

case '1':

printf("\nEnter the variable name: ");

scanf("%s", name);

printf("Enter the type (datatype, function, or array): ");

scanf("%s", data\_type);

printf("Enter the size (if applicable): ");

scanf("%s", size);

if(data\_type == "array"){

printf("Enter the dimensions (if it's an array, comma-separated): ");

scanf("%s", dimensions);

}

else{

printf("-");

}

printf("Enter the address: ");

scanf("%s", address);

add\_variable(name, data\_type, size, dimensions, address);

printf("Variable added to symbol table.\n");

break;

case '2':

display\_symbol\_table();

break;

case 'q':

while (head != NULL) {

struct Variable\* temp = head;

head = head->next;

free(temp);

}

return 0;

default:

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

}

}

}

Output:

