**Add New City**

#define MAX\_CITIES 30

void addCity(char cities[][50], int \*count);

void addCity(char cities[][50], int \*count) {

if (\*count >= MAX\_CITIES) {

printf("City list is full!\n");

}else{

char name[50];

int unique = 1;

printf("Enter city name: ");

scanf("%s", name);

for (int i = 0; i < \*count; i++) {

int notUnique = 1;

int j = 0;

while (cities[i][j] != '\0' && name[j] != '\0') {

if (cities[i][j] != name[j]) {

notUnique = 0;

break;

}

j++;

}

if (cities[i][j] != name[j]){

notUnique = 0;

}

if (notUnique){

unique = 0;

break;

}

}

if (unique) {

int k = 0;

while (name[k] != '\0') {

cities[\*count][k] = name[k];

k++;

}

cities[\*count][k] = '\0';

(\*count)++;

printf("City added successfully!\n");

}else{

printf("City name already exists!\n");

}

}

A computer screen shot of a black screen

Description automatically generated}

**Purpose:**  
The purpose of this function is to **add a new city to the city list**.

**Prototype:**

void addCity(char cities[][50], int \*count);

**Parameters:**

* cities[][50] – A 2D character array used to store city names.
* \*count – A pointer to the current number of cities in the list.

**How It Works:**

1. **Check capacity**
   * First, it checks if the city list is full (\*count >= MAX\_CITIES).
   * If full, it displays the message: “City list is full!”.
2. **Input city name**
   * It takes the city name from the user and stores it in the name array.
3. **Check uniqueness**
   * Compares the new name with existing cities in the list.
   * If any character does not match, notUnique = 0.
   * If the name matches an existing city, unique = 0.
4. **Add city**
   * If the name is unique, it copies the name into cities[\*count].
   * Increments the city count (\*count)++.
   * Displays: “City added successfully!”.
5. **Duplicate case**
   * If the name already exists in the list, it displays: “City name already exists!”.