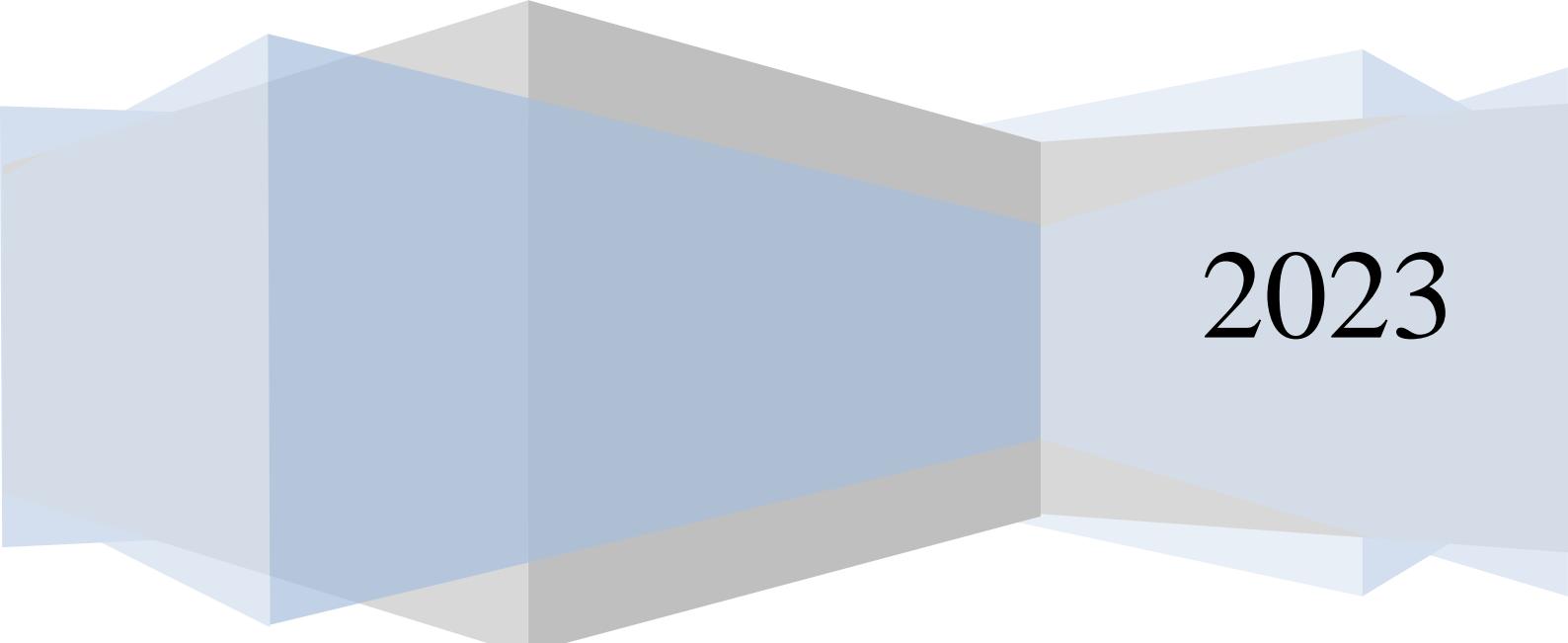


Algoritmia

Practica 1:

Hashsing

David Piñuel Bosque



2023

Índice

1. Código Fuente del Programa de Hashing Ejercicio 1	4
2. Código Fuente del Programa de Hashing Ejercicio 2	6
3. Código Fuente del Programa de Hashing Ejercicio 3	9
4. Código Fuente del Programa de Hashing Ejercicio 4	12
5. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 1.....	14
6. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 2.....	16
7. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 3.....	18
8. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 4.....	21
9. Preguntas	28

Índice Ilustración

Ilustración 1: Código Fuente Ejercicio 1 Parte 1	4
Ilustración 2: Código Fuente Ejercicio 1 Parte 2	4
Ilustración 3: Código Fuente Ejercicio 1 Parte 3	5
Ilustración 4: Código Fuente Ejercicio 1 Parte 4	5
Ilustración 5: Código Fuente Ejercicio 1 Parte 5	6
Ilustración 6: Código Fuente Ejercicio 2 Parte 1	6
Ilustración 7: Código Fuente Ejercicio 2 Parte 2	7
Ilustración 8: Código Fuente Ejercicio 2 Parte 3	7
Ilustración 9: Código Fuente Ejercicio 2 Parte 4	8
Ilustración 10: Código Fuente Ejercicio 2 Parte 5	8
Ilustración 11: Código Fuente Ejercicio 2 Parte 6	9
Ilustración 12: Código Fuente Ejercicio 3 Parte 1	9
Ilustración 13: Código Fuente Ejercicio 3 Parte 2	10
Ilustración 14: Código Fuente Ejercicio 3 Parte 3	10
Ilustración 15: Código Fuente Ejercicio 3 Parte 4	11
Ilustración 16: Código Fuente Ejercicio 3 Parte 5	11
Ilustración 17: Código Fuente Ejercicio 4 Parte 1	12
Ilustración 18: Código Fuente Ejercicio 4 Parte 2	12
Ilustración 19: Código Fuente Ejercicio 4 Parte 3	13
Ilustración 20: Código Fuente Ejercicio 4 Parte 4	13
Ilustración 21: Código Fuente Ejercicio 4 Parte 5	14
Ilustración 22: Prueba ejecución y compilación Ejercicio 1 Parte 1	14
Ilustración 23: Prueba ejecución y compilación Ejercicio 1 Parte 2	15
Ilustración 24: Prueba ejecución y compilación Ejercicio 1 Parte 3	15

<i>Ilustración 25: Prueba ejecución y compilación Ejercicio 1 Parte 4</i>	16
<i>Ilustración 26: Prueba ejecución y compilación Ejercicio 2 Parte 1</i>	16
<i>Ilustración 27: Prueba ejecución y compilación Ejercicio 2 Parte 2</i>	17
<i>Ilustración 28: Prueba ejecución y compilación Ejercicio 2 Parte 3</i>	17
<i>Ilustración 29: Prueba ejecución y compilación Ejercicio 2 Parte 4</i>	18
<i>Ilustración 30: Prueba ejecución y compilación Ejercicio 3 Parte 1</i>	18
<i>Ilustración 31: Prueba ejecución y compilación Ejercicio 3 Parte 2</i>	19
<i>Ilustración 32: Prueba ejecución y compilación Ejercicio 3 Parte 3</i>	19
<i>Ilustración 33: Prueba ejecución y compilación Ejercicio 3 Parte 4</i>	20
<i>Ilustración 34: Prueba ejecución y compilación Ejercicio 3 Parte 5</i>	20
<i>Ilustración 35: Prueba ejecución y compilación Ejercicio 3 Parte 6</i>	21
<i>Ilustración 36: Prueba ejecución y compilación Ejercicio 4 Parte 1</i>	21
<i>Ilustración 37: Prueba ejecución y compilación Ejercicio 4 Parte 2</i>	22
<i>Ilustración 38: Prueba ejecución y compilación Ejercicio 4 Parte 3</i>	22
<i>Ilustración 39: Prueba ejecución y compilación Ejercicio 4 Parte 4</i>	23
<i>Ilustración 40: Prueba ejecución y compilación Ejercicio 4 Parte 5</i>	23
<i>Ilustración 41: Prueba ejecución y compilación Ejercicio 4 Parte 6</i>	24
<i>Ilustración 42: Prueba ejecución y compilación Ejercicio 4 Parte 7</i>	24
<i>Ilustración 43: Prueba ejecución y compilación Ejercicio 4 Parte 8</i>	25
<i>Ilustración 44: Prueba ejecución y compilación Ejercicio 4 Parte 9</i>	25
<i>Ilustración 45: Prueba ejecución y compilación Ejercicio 4 Parte 10</i>	26
<i>Ilustración 46: Prueba ejecución y compilación Ejercicio 4 Parte 11</i>	26
<i>Ilustración 47: Prueba ejecución y compilación Ejercicio 4 Parte 12</i>	27
<i>Ilustración 48: Prueba ejecución y compilación Ejercicio 4 Parte 13</i>	27
<i>Ilustración 49: Prueba ejecución y compilación Ejercicio 4 Parte 14</i>	28

1. Código Fuente del Programa de Hashing Ejercicio 1.

The screenshot shows the Visual Studio Code interface with the file 'Ejercicio1.c' open. The code implements a hash table for storing actor information. It includes functions for initializing the hash table, inserting actors, searching for actors, removing items, calculating the load factor, displaying the table, and reading from a CSV file. The code uses structures and dynamic memory allocation.

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

#define TAM 500
#define LIBRE -1
#define BORRADO -2

typedef struct {
    char clave[510];
    char apellidos[255];
    char nombre[255];
    int edad;
    char goya[1];
    long cache;
} myreg;

void init(myreg t_hash[], int tam);
int H(char* k, int tam);
void insert(myreg r, myreg t_hash[], int tam);
int search(char* k, myreg t_hash[], int tam);
int remove_item(char* k, myreg t_hash[], int tam);
float loadfactor(myreg t_hash[], int tam);
void show(myreg t_hash[], int tam);
int leerFichero(char *nombre_fichero);

int main(int argc, char *argv[]) {
    leerFichero("listaActores.csv");
    system("PAUSE");
    return EXIT_SUCCESS;
}

void init(myreg t_hash[], int tam) {
    int i;
    for (i = 0; i < tam; i++) {
        strcpy(t_hash[i].clave, "");
        strcpy(t_hash[i].apellidos, "");
        strcpy(t_hash[i].nombre, "");
        t_hash[i].edad = LIBRE;
        strcpy(t_hash[i].goya, "");
        t_hash[i].cache = LIBRE;
    }
}

int H(char* k, int tam) {
    int sum = 0;
    int i;
    for (i = 0; k[i] != '\0'; i++) {
        sum += k[i];
    }
    return sum % tam;
}

void insert(myreg r, myreg t_hash[], int tam) {
    int pos = H(r.clave, tam);
    if (strcmp(t_hash[pos].clave, "") == 0 || strcmp(t_hash[pos].clave, "BORRADO") == 0) {
        t_hash[pos] = r;
    } else {
        printf("Colision\n");
    }
}

int search(char* k, myreg t_hash[], int tam) {
    int pos = H(k, tam);
    if (strcmp(t_hash[pos].clave, "") == 0) {
        return -1;
    } else if (strcmp(t_hash[pos].clave, k) == 0) {
        return pos;
    } else {

```

Ilustración 1: Código Fuente Ejercicio 1 Parte 1

The screenshot shows the Visual Studio Code interface with the file 'Ejercicio1.c' open, continuing from the previous part. This section contains the implementation of the hash table's search, insertion, and removal functions. It also includes a collision handling mechanism where it prints 'Colision' when a collision occurs during insertion.

```
    }
}

int search(char* k, myreg t_hash[], int tam) {
    int pos = H(k, tam);
    if (strcmp(t_hash[pos].clave, "") == 0) {
        return -1;
    } else if (strcmp(t_hash[pos].clave, k) == 0) {
        return pos;
    } else {

```

Ilustración 2: Código Fuente Ejercicio 1 Parte 2

```
C Ejercicio1.c > remove_item(char*, myreg [], int)
63     int search(char* k, myreg t_hash[], int tam) {
64         int pos = H(k, tam);
65         if (strcmp(t_hash[pos].clave, "") == 0) {
66             return -1;
67         } else if (strcmp(t_hash[pos].clave, k) == 0) {
68             return pos;
69         } else {
70             printf("Colision\n");
71             return -1;
72         }
73     }
74
75     int remove_item(char* k, myreg t_hash[], int tam) {
76         int pos = search(k, t_hash, tam);
77         if (pos == -1) {
78             return -1;
79         }
80         strcpy(t_hash[pos].clave, "BORRADO");
81         strcpy(t_hash[pos].nombre, "BORRADO");
82         strcpy(t_hash[pos].apellidos, "BORRADO");
83         t_hash[pos].edad = BORRADO;
84         strcpy(t_hash[pos].goya, "BORRADO");
85         t_hash[pos].cache = BORRADO;
86         return 1;
87     }
88
89     float loadfactor(myreg t_hash[], int tam) {
90         int n_elems = 0;
91         int i;
92         for (i = 0; i < tam; i++) {
93             if (strcmp(t_hash[i].clave, "") != 0 && strcmp(t_hash[i].clave, "BORRADO") != 0) {
94                 n_elems++;
95             }
96         }
97         return ((float)n_elems / tam);
98     }
99 
```

Ilustración 3: Código Fuente Ejercicio 1 Parte 3

```
C Ejercicio1.c > remove_item(char*, myreg [], int)
96     }
97     return ((float)n_elems / tam);
98 }
99
100 void show(myreg t_hash[], int tam) {
101     int i;
102     for (i = 0; i < tam; i++) {
103         printf("%s %s %d %d | ", t_hash[i].clave, t_hash[i].nombre, t_hash[i].apellidos, t_hash[i].edad, t_hash[i].goya, t_hash[i].cache);
104     }
105     printf("\n");
106 }
107
108 int leerFichero(char *nombre_fichero) {
109     myreg regI;
110     char linea[500];
111     char *token;
112
113     FILE *fp = fopen(nombre_fichero, "r");
114     if (fp == NULL) {
115         printf("Error de lectura del archivo\n");
116         return -1;
117     }
118
119     myreg myhash[TAM];
120     init(myhash, TAM);
121
122     while (fgets(linea, 500, fp) != NULL) {
123         token = strtok(linea, ";");
124         strcpy(regI.apellidos, token);
125
126         token = strtok(NULL, ";");
127         strcpy(regI.nombre, token);
128
129         strcpy(regI.clave, regI.apellidos);
130         strcat(regI.clave, regI.nombre);
131
132         token = strtok(NULL, ";");
133     }
134 }
```

Ilustración 4: Código Fuente Ejercicio 1 Parte 4

```
C Ejercicio1.c > remove_item(char *, myreg [], int)
115     remove_item(char *, myreg [], int)
116     {
117         return -1;
118     }
119
120     myreg myhash[TAM];
121     init(myhash, TAM);
122
123     while (fgets(Linea, 500, fp) != NULL) {
124         token = strtok(Linea, ";");
125         strcpy(reg1.apellidos, token);
126
127         token = strtok(NULL, ";");
128         strcpy(reg1.nombre, token);
129
130         strcpy(reg1.clave, reg1.apellidos);
131         strcat(reg1.clave, reg1.nombre);
132
133         token = strtok(NULL, " ");
134         reg1.edad = atoi(token);
135
136         token = strtok(NULL, ";");
137         strcpy(reg1.goya, token);
138
139         token = strtok(NULL, ";");
140         reg1.cache = atoi(token);
141
142         insert(reg1, myhash, TAM);
143     }
144
145     show(myhash, TAM);
146     fclose(fp);
147     return 0;
148 }
```

Ilustración 5: Código Fuente Ejercicio 1 Parte 5

2. Código Fuente del Programa de Hashing Ejercicio 2.

The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** Ejercicio2.c - Ejercicio2 - Visual Studio Code
- Left Sidebar (Explorer Tab):**
 - OPEN EDITORS: C Ejercicio2.c
 - EJERCICIOS: C Ejercicio2.c
 - listaActores.csv
- Central Area (Editor):** The code editor displays the C program Ejercicio2.c. The code defines a structure for heroes, initializes a hash table, and implements functions for inserting, searching, removing items, calculating load factor, and displaying the table. It also includes a main function that reads from a CSV file and pauses execution.
- Right Sidebar:** Includes tabs for Problems, Tasks, and Run.
- Bottom Status Bar:** Line 1, Col 1, Spaces: 4, UTF-8, CR LF, F1, Win32, 13:11, 02/06/2023.

Ilustración 6: Código Fuente Ejercicio 2 Parte 1

```
File Edit Selection View Go Run Terminal Help Ejercicio2.c - Ejercicio2 - Visual Studio Code

OPEN EDITORS
  Ejercicio2.c
  Ejercicio2.c
EJERCICIO...
  Ejercicio2.c
listaActores.csv

void init(myreg t_hash[], int tam) {
    int i;
    for (i = 0; i < tam; i++) {
        strcpy(t_hash[i].clave, "");
        strcpy(t_hash[i].apellidos, "");
        strcpy(t_hash[i].nombre, "");
        t_hash[i].edad = LIBRE;
        strcpy(t_hash[i].goya, "");
        t_hash[i].cache = LIBRE;
    }
}

int H(char* k, int tam) {
    int sum = 0;
    int i;
    for (i = 0; k[i] != '\0'; i++) {
        sum += k[i];
    }
    return sum % tam;
}

int nextPos(int pos, int tam) {
    return (pos + 1) % tam;
}

void insert(myreg r, myreg t_hash[], int tam) {
    int pos = H(r.clave, tam);
    int i = 0;
    while (strcmp(t_hash[pos].clave, "") != 0 && strcmp(t_hash[pos].clave, "BORRADO") != 0) {
        pos = nextPos(pos, tam);
        i++;
        if (i >= tam) {
            printf("Tabla llena\n");
            return;
        }
    }
}

int search(char* k, myreg t_hash[], int tam) {
    int pos = H(k, tam);
    int i = 0;
    while (strcmp(t_hash[pos].clave, "") != 0) {
        if (strcmp(t_hash[pos].clave, k) == 0) {
            return pos;
        }
        pos = nextPos(pos, tam);
        i++;
        if (i >= tam) {
            break;
        }
    }
    return -1;
}

int remove_item(char* k, myreg t_hash[], int tam) {
    int pos = search(k, t_hash, tam);
    if (pos == -1) {
        return -1;
    }
    strcpy(t_hash[pos].clave, "BORRADO");
    strcpy(t_hash[pos].nombre, "BORRADO");
    strcpy(t_hash[pos].apellidos, "BORRADO");
    t_hash[pos].edad = BORRADO;
    strcpy(t_hash[pos].goya, "BORRADO");
    t_hash[pos].cache = BORRADO;
    return 1;
}

Ln 1, Col 1  Spaces: 4  UTF-8  CRLF  ⚡ C  Win32  ⌂  13:11  02/06/2023
```

Ilustración 7: Código Fuente Ejercicio 2 Parte 2

```
File Edit Selection View Go Run Terminal Help Ejercicio2.c - Ejercicio2 - Visual Studio Code

OPEN EDITORS
  Ejercicio2.c
  Ejercicio2.c
EJERCICIO...
  Ejercicio2.c
listaActores.csv

int search(char* k, myreg t_hash[], int tam) {
    int pos = H(k, tam);
    int i = 0;
    while (strcmp(t_hash[pos].clave, "") != 0) {
        if (strcmp(t_hash[pos].clave, k) == 0) {
            return pos;
        }
        pos = nextPos(pos, tam);
        i++;
        if (i >= tam) {
            break;
        }
    }
    return -1;
}

int remove_item(char* k, myreg t_hash[], int tam) {
    int pos = search(k, t_hash, tam);
    if (pos == -1) {
        return -1;
    }
    strcpy(t_hash[pos].clave, "BORRADO");
    strcpy(t_hash[pos].nombre, "BORRADO");
    strcpy(t_hash[pos].apellidos, "BORRADO");
    t_hash[pos].edad = BORRADO;
    strcpy(t_hash[pos].goya, "BORRADO");
    t_hash[pos].cache = BORRADO;
    return 1;
}

Ln 1, Col 1  Spaces: 4  UTF-8  CRLF  ⚡ C  Win32  ⌂  13:12  02/06/2023
```

Ilustración 8: Código Fuente Ejercicio 2 Parte 3

```
C Ejercicio2.c
C Ejercicio2.c ...
C Ejercicio2.c ...
97     ++<pos> --> 1;
98 }
99
100 strcpy(t_hash[pos].clave, "BORRADO");
101 strcpy(t_hash[pos].nombre, "BORRADO");
102 strcpy(t_hash[pos].apellidos, "BORRADO");
103 t_hash[pos].edad = BORRADO;
104 strcpy(t_hash[pos].goya, "BORRADO");
105 t_hash[pos].cache = BORRADO;
106
107 return 1;
108 }
109
110 float loadfactor(myreg t_hash[], int tam) {
111     int n_elems = 0;
112     int i;
113
114     for (i = 0; i < tam; i++) {
115         if ((strcmp(t_hash[i].clave, "") != 0 && strcmp(t_hash[i].clave, "BORRADO") != 0) ||
116             n_elems++);
117     }
118 }
119
120 return ((float)n_elems / tam);
121 }
122
123 void show(myreg t_hash[], int tam) {
124     int i;
125
126     for (i = 0; i < tam; i++) {
127         printf("%s %s %s %d | ", t_hash[i].clave, t_hash[i].nombre, t_hash[i].apellidos, t_hash[i].edad, t_hash[i].goya, t_hash[i].cache);
128     }
129
130     printf("\n");
131 }
132
133 int leerFichero(char *nombre_fichero) {
```

Ilustración 9: Código Fuente Ejercicio 2 Parte 4

```
File Edit Selection View Go Run Terminal Help Ejercicio2c - Ejercicio2 - Visual Studio Code

EXPLORER OPEN EDITORS EJERCICIO2C Ejercicio2.c ...
130     printf("\n");
131 }
132
133 int leerFichero(char *nombre_fichero) {
134     myreg regt;
135     char linea[500];
136     char *token;
137
138     FILE *fp = fopen(nombre_fichero, "r");
139     if (fp == NULL) {
140         printf("Error de lectura del archivo\n");
141         return -1;
142     }
143
144     myreg myhash[TAM];
145     init(myhash, TAM);
146
147     while (fgets(linea, 500, fp) != NULL) {
148         token = strtok(linea, ";");
149         strcpy(regt.apellidos, token);
150
151         token = strtok(NULL, ";");
152         strcpy(regt.nombre, token);
153
154         strcpy(regt.clave, regt.apellidos);
155         strcat(regt.clave, regt.nombre);
156
157         token = strtok(NULL, ";");
158         regt.edad = atoi(token);
159
160         token = strtok(NULL, ";");
161         strcpy(regt.goya, token);
162
163         token = strtok(NULL, ";");
164         regt.cache = atoi(token);
165
166         insert(regt, myhash, TAM);
167     }
168 }
```

Ilustración 10: Código Fuente Ejercicio 2 Parte 5

```
File Edit Selection View Go Run Terminal Help
Ejercicio2.c - Ejercicio2 - Visual Studio Code
OPEN EDITORS
EJERCICIO... Ejercicio2.c
Ejercicio2.c
listaActores.csv
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
```

The screenshot shows the Visual Studio Code interface with the file 'Ejercicio2.c' open. The code is a C program that reads a CSV file ('listaActores.csv') and inserts data into a hash table. The code uses strtok to parse the CSV file and struct Node to represent the linked list nodes. It includes definitions for TAM (table size), LIBRE (free), and NULL. The main function initializes the hash table, creates a node, inserts it, and then shows the table.

Ilustración 11: Código Fuente Ejercicio 2 Parte 6

3. Código Fuente del Programa de Hashing Ejercicio 3.

```
File Edit Selection View Go Run Terminal Help
Ejercicio3.c - Ejercicio3 - Visual Studio Code
OPEN EDITORS
EJERCICIO... Ejercicio3.c
Ejercicio3.c
listaActores.csv
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

The screenshot shows the Visual Studio Code interface with the file 'Ejercicio3.c' open. The code defines the structures for a Node, a linked list (LinkedList), and a hash table (HashTable). It also includes functions for initializing the hash table, creating a node, inserting data into the hash table, and showing the contents. The main function demonstrates how to use these components to read data from a CSV file and store it in a hash table.

Ilustración 12: Código Fuente Ejercicio 3 Parte 1

```
File Edit Selection View Go Run Terminal Help
OPEN EDITORS
  Ejercicio3.c
  Ejercicio3.h
Ejercicio3.c > ...
33 int main(int argc, char* argv[]) {
34     HashTable* hashtable;
35     init(&hashtable);
36     leerFichero("listaActores.csv");
37     show(&hashtable);
38     system("PAUSE");
39     return EXIT_SUCCESS;
40 }
41
42 void init(HashTable* hashtable) {
43     int i;
44     for (i = 0; i < TAM; i++) {
45         hashtable->table[i] = (LinkedList*)malloc(sizeof(LinkedList));
46         hashtable->table[i]->head = LIBRE;
47     }
48 }
49
50 int hash(char* k) {
51     int sum = 0;
52     int i;
53     for (i = 0; k[i] != '\0'; i++) {
54         sum += k[i];
55     }
56     return sum % TAM;
57 }
58
59 Node* createNode(char* clave, char* apellidos, char* nombre, int edad, char* goya, long cache) {
60     Node* newNode = (Node*)malloc(sizeof(Node));
61     strcpy(newNode->clave, clave);
62     strcpy(newNode->apellidos, apellidos);
63     strcpy(newNode->nombre, nombre);
64     newNode->edad = edad;
65     strcpy(newNode->goya, goya);
66     newNode->cache = cache;
67     newNode->next = NULL;
68     return newNode;
69 }
70
71 void insert(HashTable* hashtable, char* clave, char* apellidos, char* nombre, int edad, char* goya, long cache) {
72     int pos = hash(clave);
73     Node* newNode = createNode(clave, apellidos, nombre, edad, goya, cache);
74
75     if (hashtable->table[pos]->head == LIBRE) {
76         hashtable->table[pos]->head = newNode;
77     } else {
78         Node* temp = hashtable->table[pos]->head;
79         while (temp->next != NULL) {
80             temp = temp->next;
81         }
82         temp->next = newNode;
83     }
84 }
85
86 void show(HashTable* hashtable) {
87     int i;
88     for (i = 0; i < TAM; i++) {
89         printf("Posición %d: ", i);
90         Node* temp = hashtable->table[i]->head;
91         while (temp != NULL) {
92             printf("%s %s %s %d %s | ", temp->clave, temp->nombre, temp->apellidos, temp->edad, temp->goya, temp->cache);
93             temp = temp->next;
94         }
95         printf("\n");
96     }
97 }
98
99 int leerFichero(char* nombre_fichero) {
100    char linea[500];
101    char token;
```

Ilustración 13: Código Fuente Ejercicio 3 Parte 2

```
File Edit Selection View Go Run Terminal Help
OPEN EDITORS
  Ejercicio3.c
  Ejercicio3.h
Ejercicio3.c > ...
66     char linea[500];
67     char token;
68 }
69
70 void insert(HashTable* hashtable, char* clave, char* apellidos, char* nombre, int edad, char* goya, long cache) {
71     int pos = hash(clave);
72     Node* newNode = createNode(clave, apellidos, nombre, edad, goya, cache);
73
74     if (hashtable->table[pos]->head == LIBRE) {
75         hashtable->table[pos]->head = newNode;
76     } else {
77         Node* temp = hashtable->table[pos]->head;
78         while (temp->next != NULL) {
79             temp = temp->next;
80         }
81         temp->next = newNode;
82     }
83 }
84
85 void show(HashTable* hashtable) {
86     int i;
87     for (i = 0; i < TAM; i++) {
88         printf("Posición %d: ", i);
89         Node* temp = hashtable->table[i]->head;
90         while (temp != NULL) {
91             printf("%s %s %s %d %s | ", temp->clave, temp->nombre, temp->apellidos, temp->edad, temp->goya, temp->cache);
92             temp = temp->next;
93         }
94         printf("\n");
95     }
96 }
97
98 int leerFichero(char* nombre_fichero) {
99     char linea[500];
100    char token;
```

Ilustración 14: Código Fuente Ejercicio 3 Parte 3

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** Ejercicio3.c - Ejercicio3 - Visual Studio Code
- Explorer Panel (Left):**
 - OPEN EDITORS: Ejercicio3.c
 - EJERCICIOS:
 - Ejercicio3.c
 - listaActores.csv
- Editor Area (Center):** The code for `Ejercicio3.c` is displayed. The code reads from a file, tokenizes lines, and inserts data into a hash table. It includes functions for reading files, strtok, atoi, and inserting into a hash table.
- Bottom Status Bar:** Line 1, Col 1 | Spaces: 4 | UTF-8 | CRLF | Win32 | 13:15 | 02/06/2023
- Bottom Taskbar:** Icons for File, Find, Replace, Save, Run, Terminal, and others.

Ilustración 15: Código Fuente Ejercicio 3 Parte 4

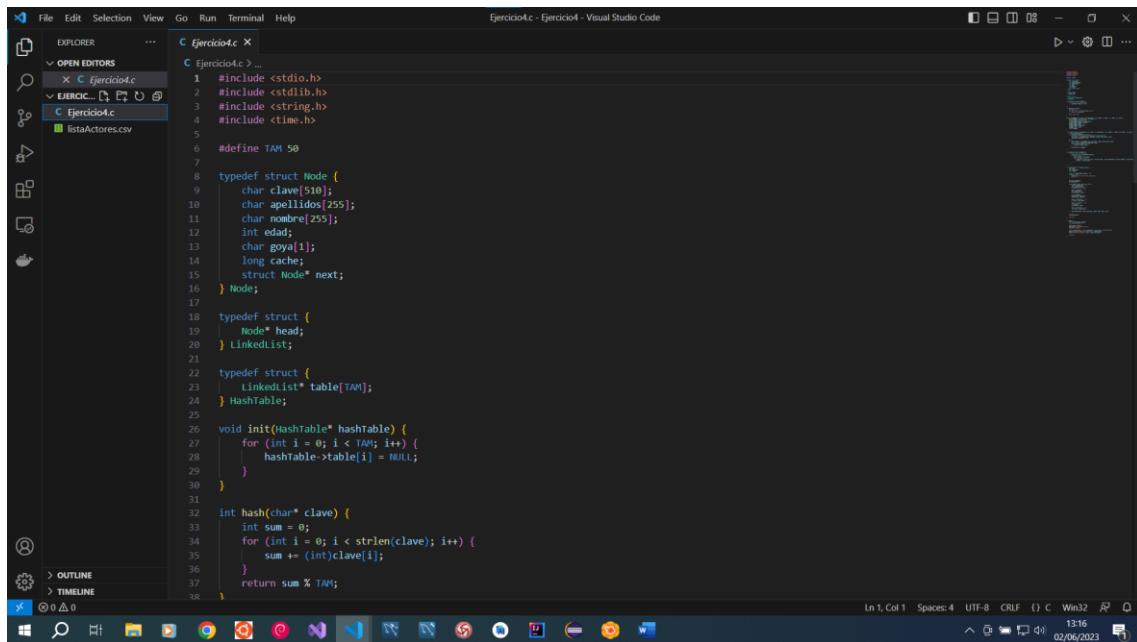
```
File Edit Selection View Go Run Terminal Help
Ejercicio3.c - Ejercicio3 - Visual Studio Code

EXPLORER OPEN EDITORS EJERCICIOS Ejercicio3.c listaActores.csv

C Ejercicio3.c x
115 strcpy(apellidos, token);
116
117 token = strtok(NULL, ";");
118 char nombre[255];
119 strcpy(nombre, token);
120
121 char clave[50];
122 strcpy(clave, apellidos);
123 strcat(clave, nombre);
124
125 token = strtok(NULL, ";");
126 int edad = atoi(token);
127
128 token = strtok(NULL, ";");
129 char goya[1];
130 strcpy(goya, token);
131
132 token = strtok(NULL, ";");
133 long cache = atoi(token);
134
135 insert(&hashTable, clave, apellidos, nombre, edad, goya, cache);
136 }
137
138 show(&hashTable);
139 fclose(fp);
140 return 0;
141 }
142 }
```

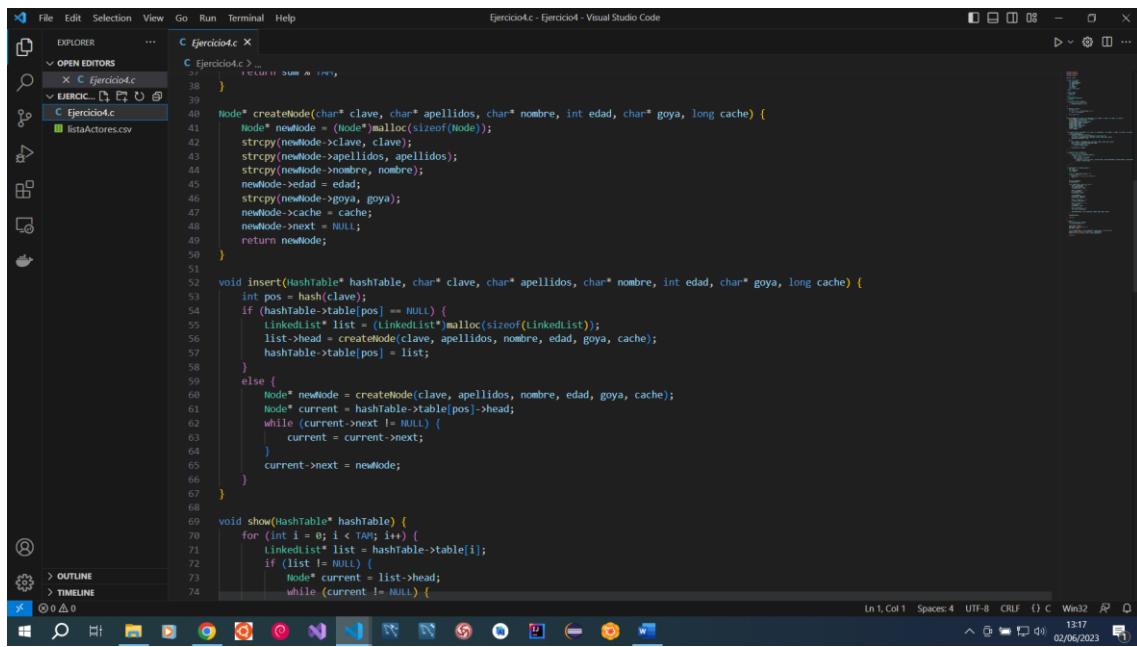
Ilustración 16: Código Fuente Ejercicio 3 Parte 5

4. Código Fuente del Programa de Hashing Ejercicio 4.



```
C:\ Ejercicio4.c
C:\ Ejercicio4.c > ...
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5
6 #define TAM 50
7
8 typedef struct Node {
9     char Clave[510];
10    char apellidos[255];
11    char nombre[255];
12    int edad;
13    char goya[1];
14    long cache;
15    struct Node* next;
16 } Node;
17
18 typedef struct {
19     Node* head;
20 } LinkedList;
21
22 typedef struct {
23     LinkedList* table[TAM];
24 } HashTable;
25
26 void init(HashTable* hashtable) {
27     for (int i = 0; i < TAM; i++) {
28         hashtable->table[i] = NULL;
29     }
30 }
31
32 int hash(char* clave) {
33     int sum = 0;
34     for (int i = 0; i < strlen(clave); i++) {
35         sum += (int)clave[i];
36     }
37     return sum % TAM;
38 }
```

Ilustración 17: Código Fuente Ejercicio 4 Parte 1



```
39
40 Node* createNode(char* clave, char* apellidos, char* nombre, int edad, char* goya, long cache) {
41     Node* newNode = (Node*)malloc(sizeof(Node));
42     strcpy(newNode->clave, clave);
43     strcpy(newNode->apellidos, apellidos);
44     strcpy(newNode->nombre, nombre);
45     newNode->edad = edad;
46     strcpy(newNode->goya, goya);
47     newNode->cache = cache;
48     newNode->next = NULL;
49     return newNode;
50 }
51
52 void insert(HashTable* hashtable, char* clave, char* apellidos, char* nombre, int edad, char* goya, long cache) {
53     int pos = hash(clave);
54     if (hashtable->table[pos] == NULL) {
55         LinkedList* list = (LinkedList*)malloc(sizeof(LinkedList));
56         list->head = createNode(clave, apellidos, nombre, edad, goya, cache);
57         hashtable->table[pos] = list;
58     }
59     else {
60         Node* newNode = createNode(clave, apellidos, nombre, edad, goya, cache);
61         Node* current = hashtable->table[pos]->head;
62         while (current->next != NULL) {
63             current = current->next;
64         }
65         current->next = newNode;
66     }
67 }
68
69 void show(HashTable* hashtable) {
70     for (int i = 0; i < TAM; i++) {
71         LinkedList* list = hashtable->table[i];
72         if (list != NULL) {
73             Node* current = list->head;
74             while (current != NULL) {
```

Ilustración 18: Código Fuente Ejercicio 4 Parte 2

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a tree view with "OPEN EDITORS" expanded, displaying "Ejercicio4.c".
- Editor:** The main pane contains the C code for "Ejercicio4.c".
- Code:**

```
C Ejercicio4.c
C Ejercicio4 > ...
67
68 void show(HashTable* hashTable) {
69     for (int i = 0; i < 700; i++) {
70         linkedList* list = hashTable->table[i];
71         if (list != NULL) {
72             Node* current = list->head;
73             while (current != NULL) {
74                 printf("%s %s %s %d %d\n", current->clave, current->apellidos, current->nombre, current->edad, current->goya, current->cache);
75                 current = current->next;
76             }
77         }
78     }
79 }
80
81 int leerFichero(char* nombre_fichero) {
82     Node reg1;
83     char linea[500];
84     char* token;
85
86     FILE* fp = fopen(nombre_fichero, "r");
87     if (fp == NULL) {
88         printf("Error de lectura del archivo\n");
89         return -1;
90     }
91
92     HashTable hashTable;
93     init(&hashTable);
94
95     while (fgets(linea, 500, fp) != NULL) {
96         token = strtok(linea, ";");
97         char apellidos[255];
98         strcpy(apellidos, token);
99
100        token = strtok(NULL, ";");
101        char nombre[255];
102        strcpy(nombre, token);
103
104        token = strtok(NULL, ";");
105        char edad[255];
106        strcpy(edad, token);
107
108        token = strtok(NULL, ";");
109        char goya[255];
110        strcpy(goya, token);
111
112        token = strtok(NULL, ";");
113        char cache[255];
114        strcpy(cache, token);
115
116        reg1.clave = apellidos;
117        reg1.apellidos = nombre;
118        reg1.nombre = edad;
119        reg1.edad = goya;
120        reg1.goya = cache;
121
122        insert(hashTable, &reg1);
123    }
124 }
```
- Status Bar:** Shows "ln 1, Col 1" and other standard status bar information.

Ilustración 19: Código Fuente Ejercicio 4 Parte 3

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows files like "Ejercicio4.c", "Ejercicio4.h", and "Ejercicio4.h" in the "OPEN EDITORS" section.
- Code Editor:** Displays a C program named "Ejercicio4.c". The code reads from a CSV file ("listaActores.csv") and inserts data into a hash table. It uses strtok to parse tokens and atoi to convert strings to integers.
- Output:** Shows the output "Ejercicio4 - Ejercicio4 - Visual Studio Code".
- Status Bar:** Shows "Ln 1, Col 1" and other standard status bar information.

```
C Ejercicio4.c
C Ejercicio4.c ...
C Ejercicio4.c ...
91 }
92
HashTable hashTable;
init(&hashTable);
93
while (fgets(linea, 500, fp) != NULL) {
94     token = strtok(linea, ",");
95     char apellidos[255];
96     strcpy(apellidos, token);
97
98     token = strtok(NULL, ",");
99     char nombre[255];
100    strcpy(nombre, token);
101
102    char clave[510];
103    strcpy(clave, apellidos);
104    strcat(clave, nombre);
105
106    token = strtok(NULL, ",");
107    int edad = atoi(token);
108
109    token = strtok(NULL, ",");
110    char goya[1];
111    strcpy(goya, token);
112
113    token = strtok(NULL, ",");
114    long cache = atoi(token);
115
116    token = strtok(NULL, ",");
117    insert(&hashTable, clave, apellidos, nombre, edad, goya, cache);
118
119    show(&hashTable);
120
121    return 0;
122}
123
124
125
126
127 int main() {
```

Ilustración 20: Código Fuente Ejercicio 4 Parte 4

```
C:\Ejercicio4> x
File Edit Selection View Go Run Terminal Help
Ejercicio4 - Ejercicio4 - Visual Studio Code

EXPLORER ...
OPEN EDITORS ...
C Ejercicio4.c ...
C Ejercicio4.c ...
114     strcpy(goya, token);
115
116     token = strtok(NULL, ";");
117     long cache = atol(token);
118
119     insert(&hashTable, clave, apellidos, nombre, edad, goya, cache);
120 }
121
122 show(&hashTable);
123
124 return 0;
125 }
126
127 int main() {
128     clock_t start_time, end_time;
129     int total_collisions = 0;
130
131     start_time = clock();
132     leerFichero("listaActores.csv");
133     end_time = clock();
134
135     double elapsed_time = ((double)(end_time - start_time) / CLOCKS_PER_SEC);
136     printf("Tiempo empleado: %lf segundos\n", elapsed_time);
137     printf("Total de colisiones: %d\n", total_collisions);
138
139     return 0;
140 }
141
```

Ilustración 21: Código Fuente Ejercicio 4 Parte 5

5. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 1.

Ilustración 22: Prueba ejecución y compilación Ejercicio 1 Parte 1

Ilustración 23: Prueba ejecución y compilación Ejercicio 1 Parte 2

The screenshot shows a Microsoft Visual Studio Code interface with several tabs open. The main tabs include 'Ejercicio1.c' (C) and 'Ejercicio1.cs' (C#). Other tabs like 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL' are also visible. There are multiple terminal windows at the bottom, each running a PowerShell session. One terminal shows the output of a C program, another shows the output of a C# program, and others show standard PowerShell prompts. The status bar at the bottom indicates the current file is 'Ejercicio1.c' and shows other details like file count, space usage, and system information.

Ilustración 24: Prueba ejecución y compilación Ejercicio 1 Parte 3

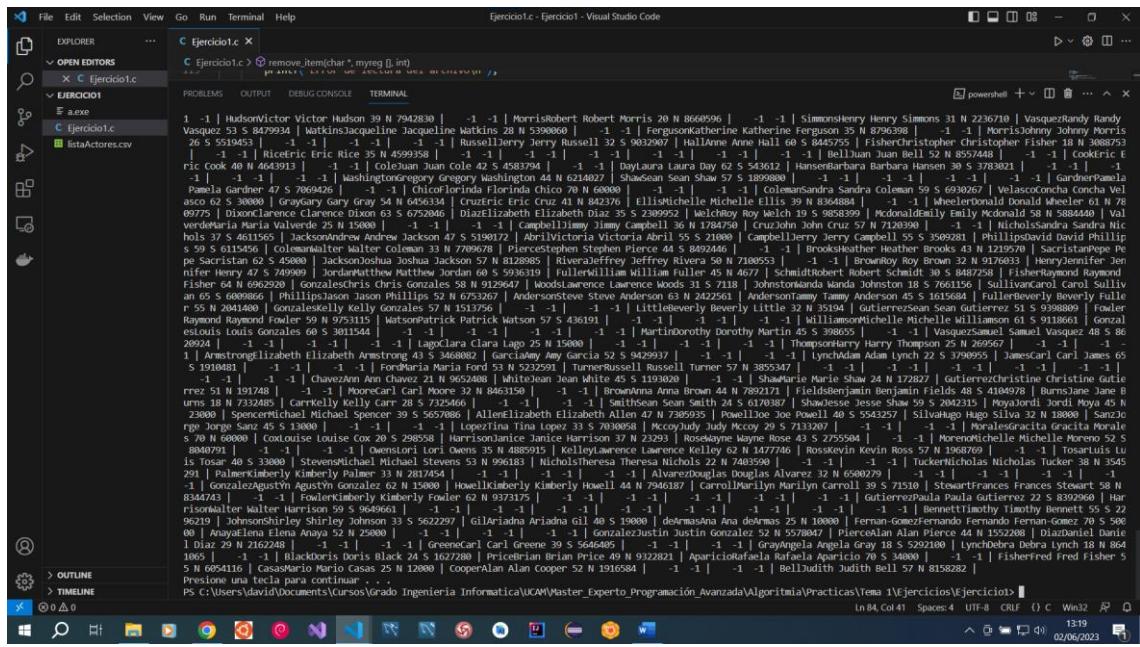


Ilustración 25: Prueba ejecución y compilación Ejercicio 1 Parte 4

6. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 2.

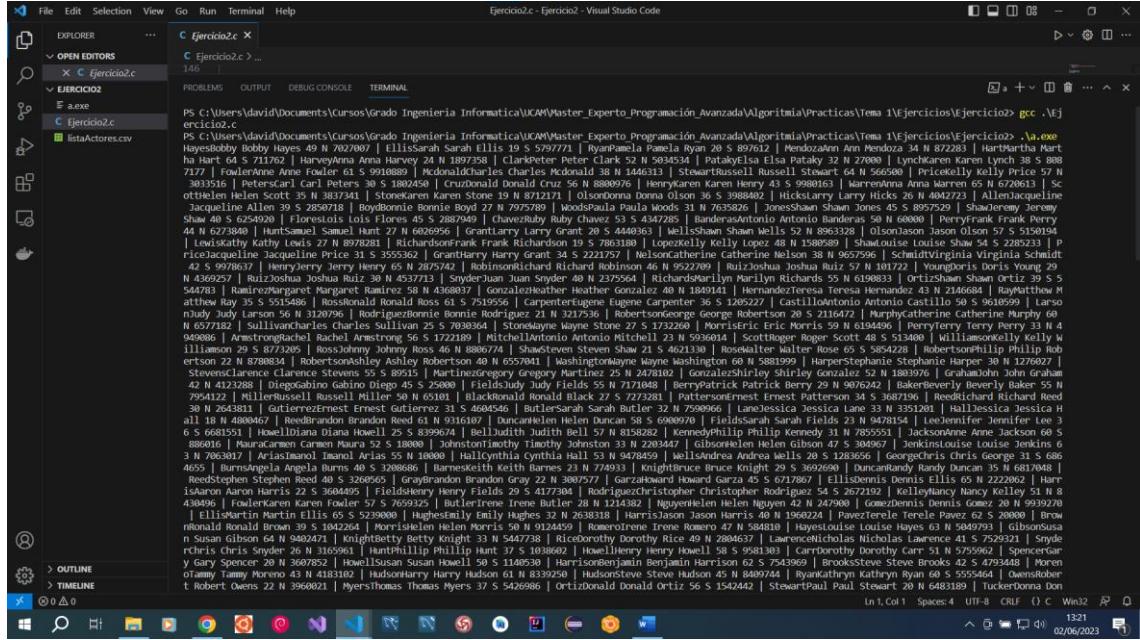


Ilustración 26: Prueba ejecución y compilación Ejercicio 2 Parte 1

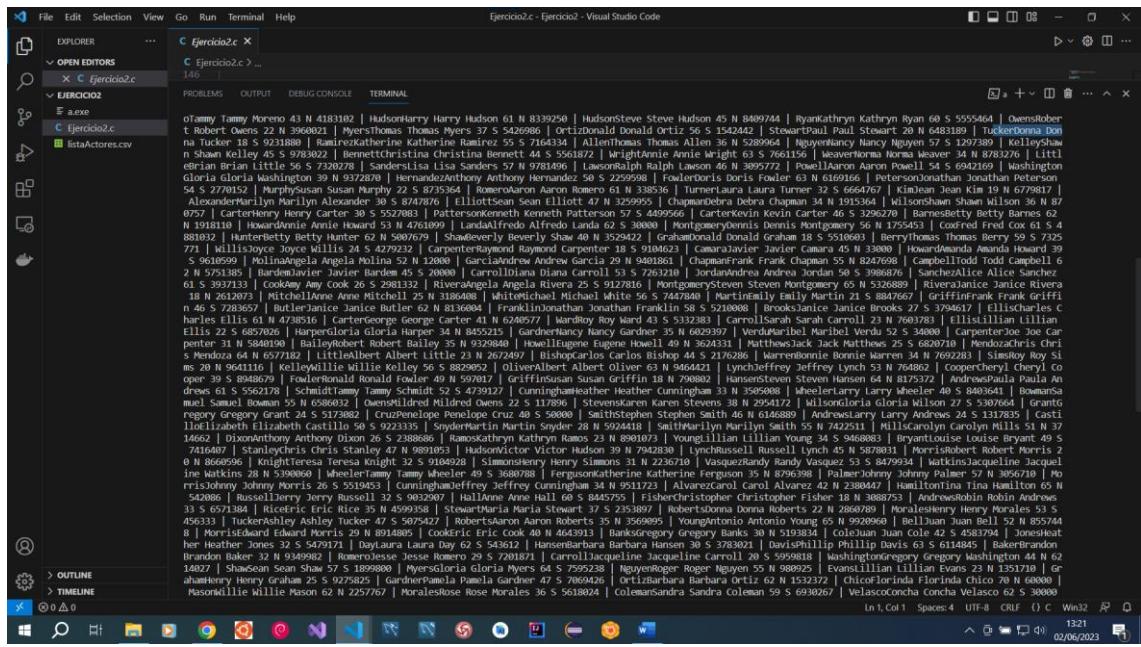


Ilustración 27: Prueba ejecución y compilación Ejercicio 2 Parte 2

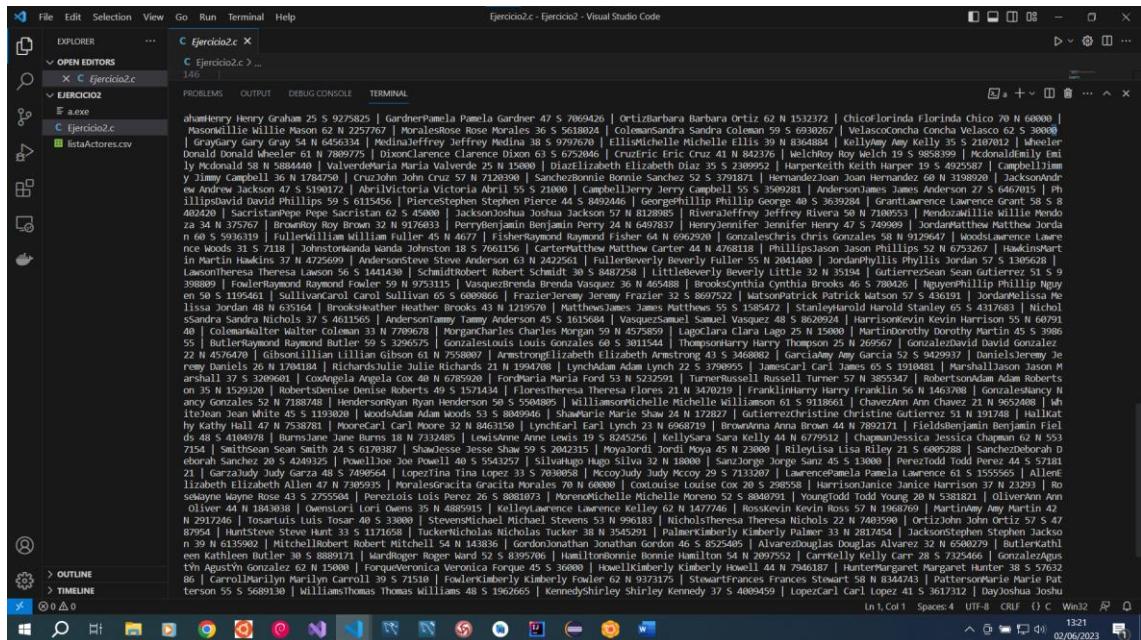


Ilustración 28: Prueba ejecución y compilación Ejercicio 2 Parte 3

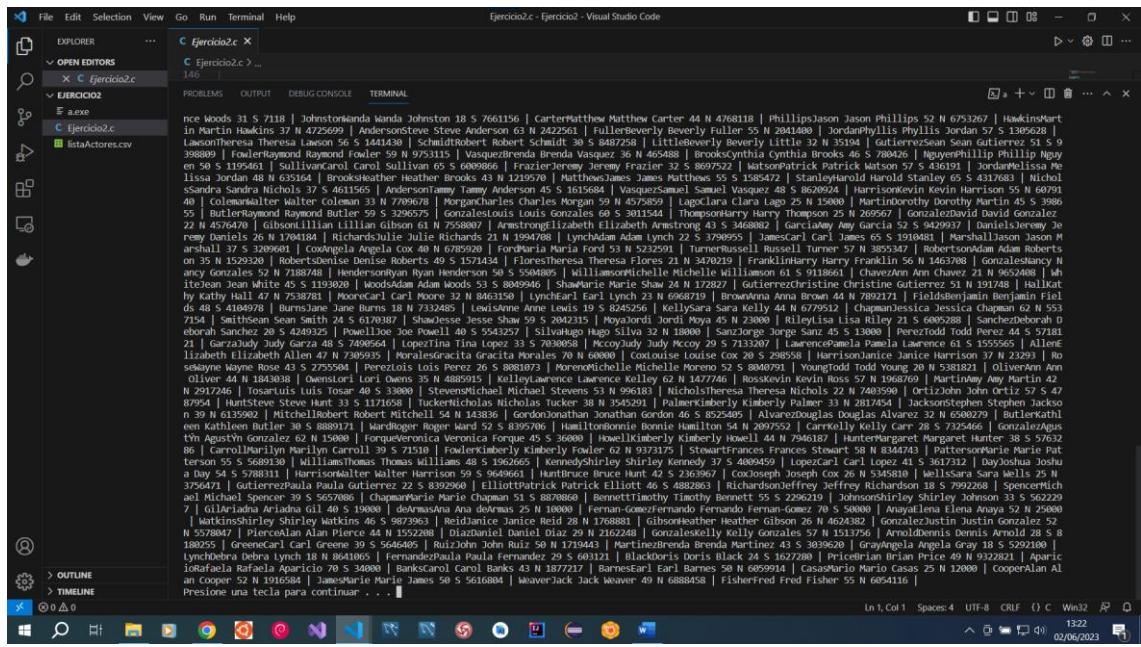


Ilustración 29: Prueba ejecución y compilación Ejercicio 2 Parte 4

7. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 3.

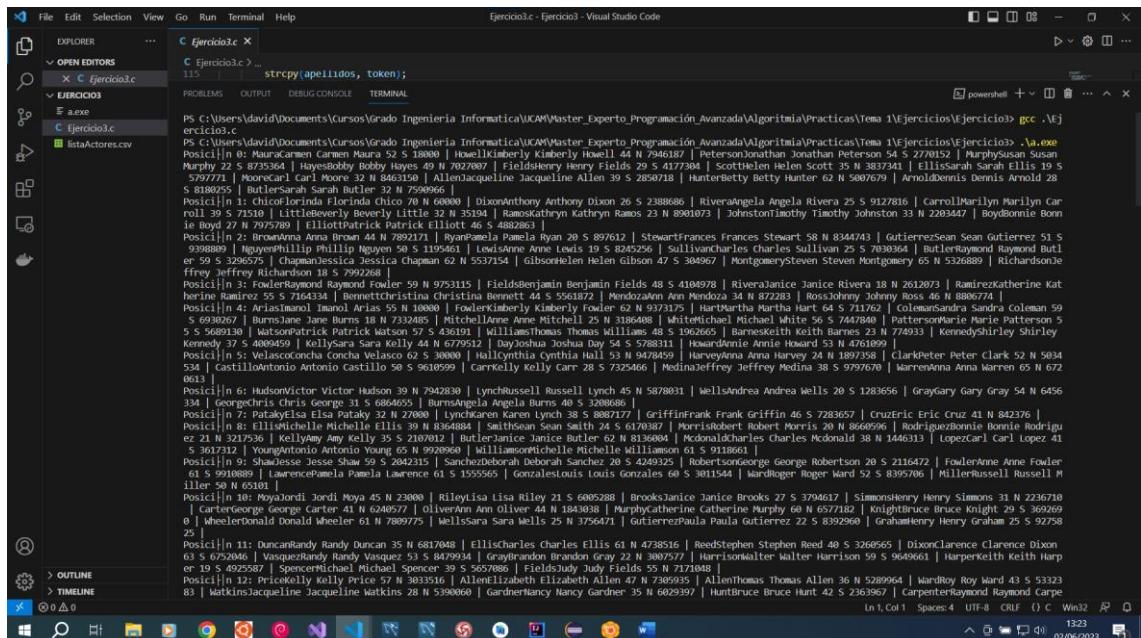


Ilustración 30: Prueba ejecución y compilación Ejercicio 3 Parte 1

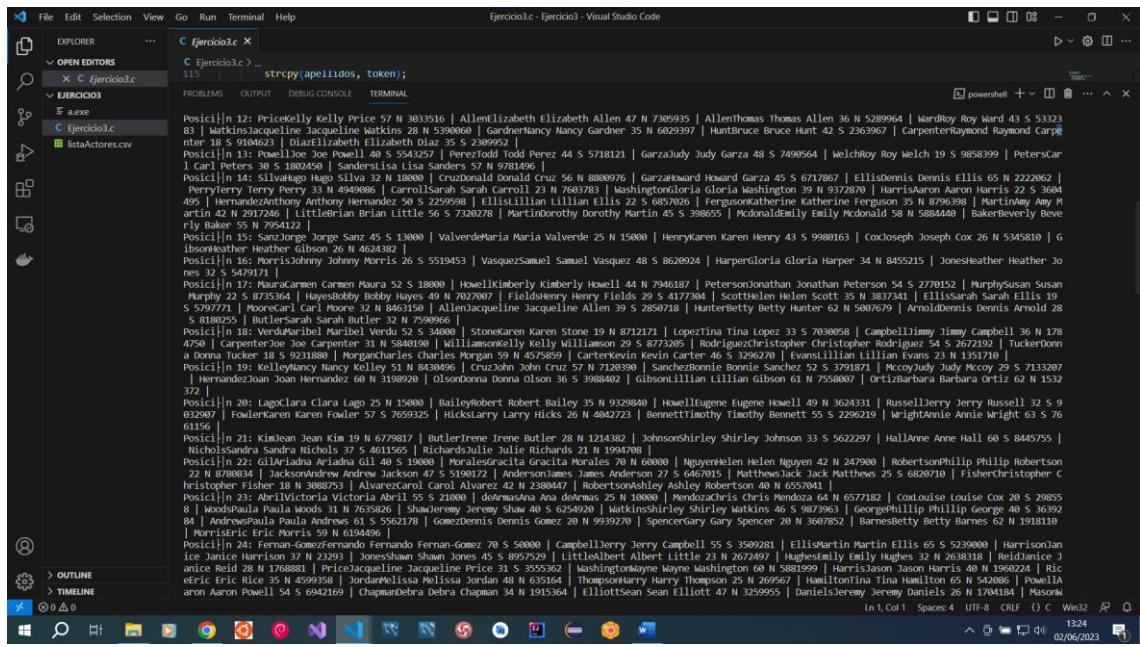


Ilustración 31: Prueba ejecución y compilación Ejercicio 3 Parte 2

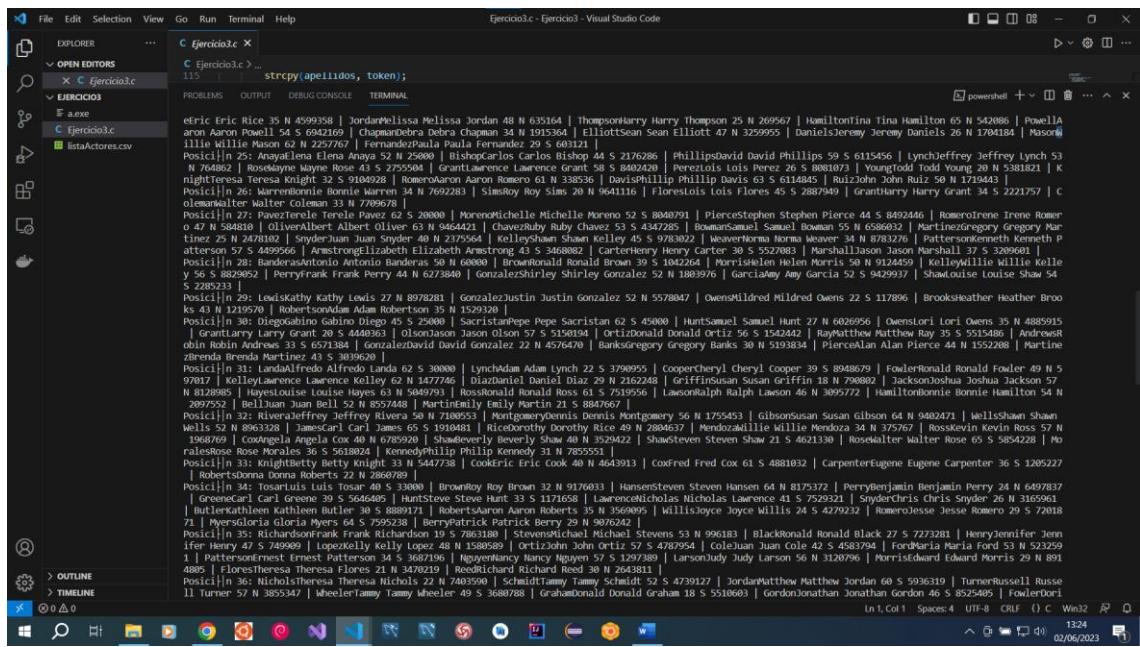


Ilustración 32: Prueba ejecución y compilación Ejercicio 3 Parte 3

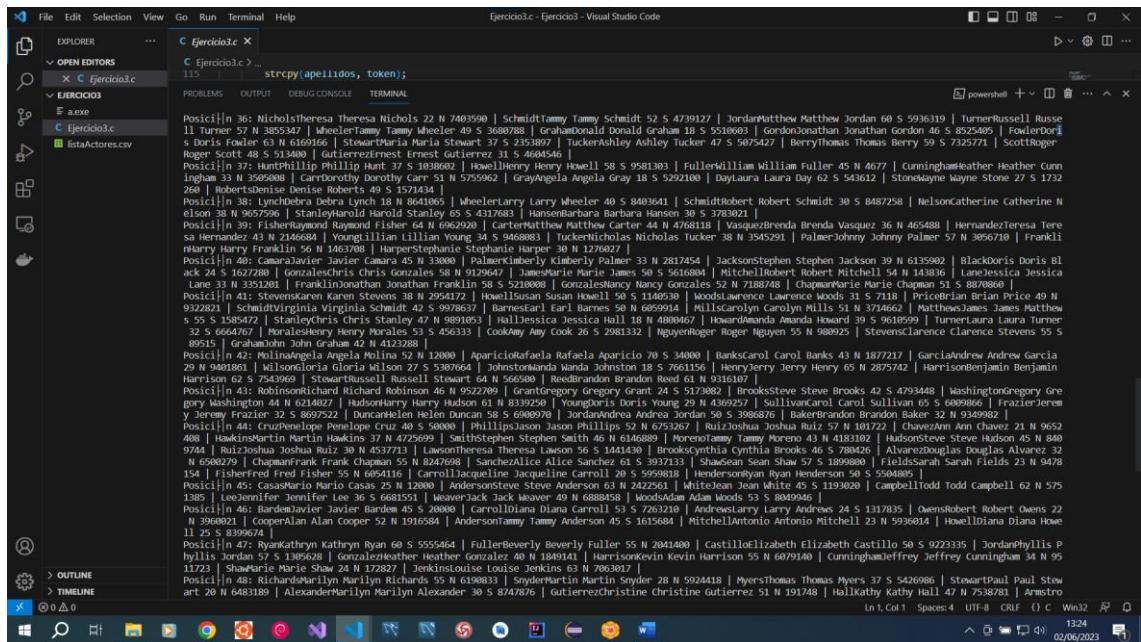


Ilustración 33: Prueba ejecución y compilación Ejercicio 3 Parte 4

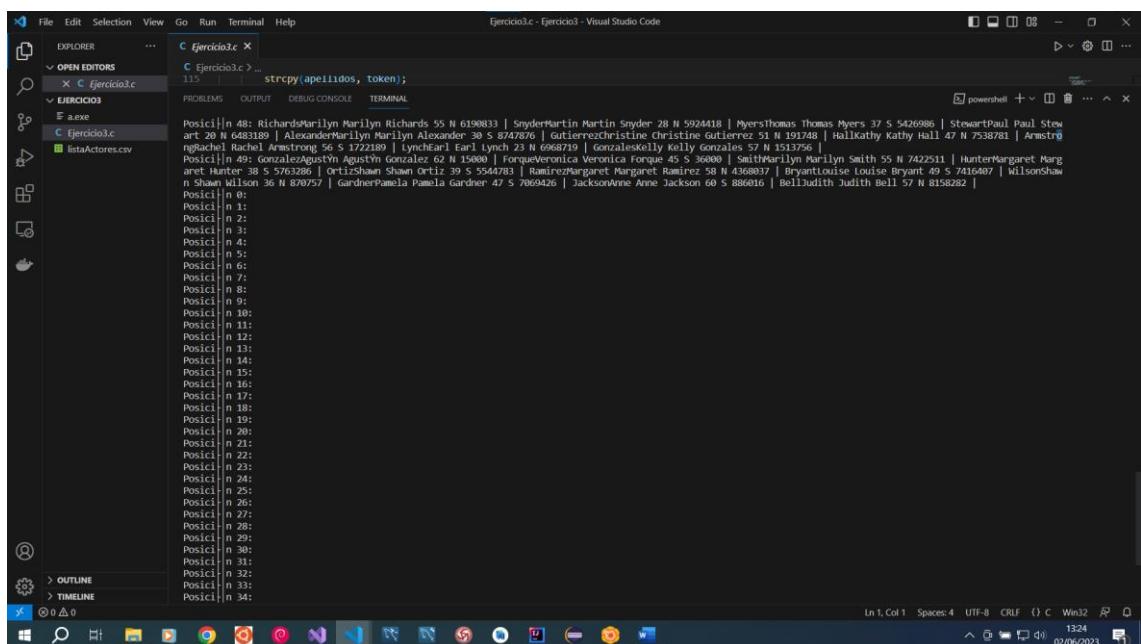


Ilustración 34: Prueba ejecución y compilación Ejercicio 3 Parte 5

```

File Edit Selection View Go Run Terminal Help Ejercicio3.c - Ejercicio3 - Visual Studio Code
OPEN EDITORS Ejercicio3.c
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
EJERCICIOS Ejercicio3.c
a.exe
listaActores.csv
Posici[n] h 11;
Posici[n] h 12;
Posici[n] h 13;
Posici[n] h 14;
Posici[n] h 15;
Posici[n] h 16;
Posici[n] h 17;
Posici[n] h 18;
Posici[n] h 19;
Posici[n] h 20;
Posici[n] h 21;
Posici[n] h 22;
Posici[n] h 23;
Posici[n] h 24;
Posici[n] h 25;
Posici[n] h 26;
Posici[n] h 27;
Posici[n] h 28;
Posici[n] h 29;
Posici[n] h 30;
Posici[n] h 31;
Posici[n] h 32;
Posici[n] h 33;
Posici[n] h 34;
Posici[n] h 35;
Posici[n] h 36;
Posici[n] h 37;
Posici[n] h 38;
Posici[n] h 39;
Posici[n] h 40;
Posici[n] h 41;
Posici[n] h 42;
Posici[n] h 43;
Posici[n] h 44;
Posici[n] h 45;
Posici[n] h 46;
Posici[n] h 47;
Posici[n] h 48;
Posici[n] h 49;
Presione una tecla para continuar . .
PS C:\Users\David\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programacion_Avanzada\Practicas\Tema 1\Ejercicios\Ejercicio3>

```

Ilustración 35: Prueba ejecución y compilación Ejercicio 3 Parte 6

8. Ejemplo de prueba del programa compilado y ejecutado Ejercicio 4.

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
EJERCICIOS Ejercicio4.c
a.exe
listaActores.csv
PS C:\Users\David\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programacion_Avanzada\Practicas\Tema 1\Ejercicios\Ejercicio4> gcc .\Ejercicio4.c
PS C:\Users\David\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programacion_Avanzada\Practicas\Tema 1\Ejercicios\Ejercicio4> .\a.exe
LandalAlfredo Landa Alfredo 62 S 30000
LandalAlfredo Landa Alfredo 62 S 30000
LynchAdam Lynch Adam 22 S 3790955
CooperCheryl Cooper Cheryl 39 S 8948679
HayesBobby Hayes Bobby 49 M 7027007
FowlerRonald Fowler Ronald 49 M 597017
KelleyLawrence Kelley Lawrence 62 N 1477746
DishonBeverly Dishon Beverly 29 F 201570
GriffithsonGriffin Griffithson Griffin 59 N 799882
JacksonJoshua Jackson Joshua 57 N 8128985
HayesLouise Hayes Louise 63 N 5049793
RossRonald Ross Ronald 61 S 5159556
EllisSarah Ellis Sarah 19 S 7797771
LaemmleRalph Laemmle Ralph 48 M 3695772
HanniganKathy Hannigan Kathy 54 N 2097552
MooreCarl Moore Carl 32 N 8463158
BellmanBelli Juan 52 N 8557448
HunterBetty Hunter Betty 62 N 5607679
MartinEmily Martin Emily 21 S 8847667
ChicotFlorinda Chico Florida 70 N 68000
DiazLuis Diaz Luis 27 M 5024568
RiveraAngela Rivera Angela 25 S 9127816
CarrollMarilyn Carroll Marilyn 39 S 71510
LittleBeverly Little Beverly 32 N 35194
RamosKathy Ramos Kathryn 23 N 8961073
JohnstonTimothy Johnston Timothy 33 N 2203447
BoyceDebbie Boyce Debbie 27 F 5024579
ElliotTricia Elliot Tricia 46 S 4882863
BrownAnna Brown Anna 44 N 7892171
RyanPamela Ryan Pamela 20 S 897012
StewartFrances Stewart Frances 58 N 8344743
GutierrezSean Gutierrez Sean 51 S 9398899
NguyenPhuoc Nguyen Phuoc 45 M 5119561
LozanoLeticia Lozano Leticia 19 S 8245226
SullivanCharles Sullivan Charles 25 S 7030364
ButlerRaymond Butler Raymond 59 S 3296575
ChapmanJessica Chapman Jessica 62 N 5537154
GibsonHelen Gibson Helen 47 S 364967
MontgomerySteven Montgomery Steven 65 N 5326889

```

Ilustración 36: Prueba ejecución y compilación Ejercicio 4 Parte 1

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
GibsonHelen Gibson Helen 47 S 304967
MontgomerySteven Montgomery Steven 65 N 5326889
RichardsonJeffrey Richardson Jeffrey 18 S 7992268
FowlerRaymond Fowler Raymond 59 N 9753115
FieldsBenjamin Fields Benjamin 48 S 8104976
RiveraJanice Rivera Janice 18 N 2612073
RamirezAudrey Ramirez Audrey 55 S 7164334
BennettChristina Bennett Christina 44 S 5561872
MendozaAnn Mendoza Ann 34 N 872283
RossJohnny Ross Johnny 46 N 8806774
AriasImanol Arias Imanol 55 N 10000
FowlerKimberly Fowler Kimberly 62 N 9373175
HartMartha Hart Martha 50 N 8761762
CollierLorraine Collier Lorraine 59 N 5930267
BurnsJane Burns Jane 18 N 7332485
MitchellAnne Mitchell Anne 25 N 3186408
WhiteMichael White Michael 56 S 7447840
PattersonMarie Patterson Marie 55 S 5689130
WatsonPatrick Watson Patrick 55 S 436191
MilliganDawn Milligan Dawn 48 S 8962665
BaronKathy Baron Kathy 23 N 774933
KennedyShirley Kennedy Shirley 37 S 4009459
KellySara Kelly Sara 44 N 6779512
DayJoshua Day Joshua 54 S 5788311
HowardAnnie Howard Annie 53 N 4761099
WolfeCynthia Wolfe Cynthia 53 N 3000000
HallCynthia Hall Cynthia 53 N 9478459
HarveyAnna Harvey Anna 24 N 3897358
ClarkPeter Clark Peter 52 N 9034534
CastilloAntonio Castillo Antonio 50 S 9610599
CarrKelly Carr Kelly 28 S 7325466
MedinaLorey Medina Lorey 38 S 3977670
RanneyLorraine Ranney Lorraine 65 N 602613
HudsonVictor Hudson Victor 39 N 7942830
LynchRussell Lynch Russell 45 N 5878031
WellsAndrea Wells Andrea 20 S 1283656
GrayGary Gray Gary 54 N 6456334
GeorgeChris George Chris 31 S 6864655
BurnsAngela Burns Angela 40 S 3288086
PatakyElsa Pataky Elsa 32 N 70000
LynchKaren Lynch Karen 38 S 8807177

```

Ilustración 37: Prueba ejecución y compilación Ejercicio 4 Parte 2

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PatakyElsa Pataky Elsa 32 N 70000
LynchKaren Lynch Karen 38 S 8807177
GriffinFrank Griffin Frank 46 S 7283657
CruzEric Cruz Eric 41 N 842376
ElizMiguel Eliz Miguel 40 N 8364884
SmithsonSmith Sean 24 S 6170287
MorrisRobert Morris Robert 20 N 8660596
RodriguezBonni Rodriguez Bonnie 21 N 3217536
KellyAmy Kelly Amy 35 S 2107012
ButlerJanice Butler Janice 62 N 8136004
McDonaldPaula McDonald Paula 38 N 1446313
LopezEdgar Lopez Edgar 41 N 3811272
YoungAntonio Young Antonio 65 N 9920969
WilliamsonMichelle Williamson Michelle 61 S 9118661
ShawDessie Shaw Dessie 59 S 2023235
SanchezDeborah Sanchez Deborah 20 S 4249325
RobertsonGeorge Robertson George 29 S 2116472
FowlerAudrey Fowler Audrey 55 S 9753115
LawrenceCelia Lawrence Celia 61 S 1555565
GonzalesLouis Gonzales Louis 60 S 3011544
WardRoger Ward Roger 52 S 8395706
MillerRussell Miller Russell 50 N 65101
MoyaJordi Moya Jordi 45 N 23000
KileyJill Kiley Jill 44 S 8000088
BrooksJanice Brooks Janice 55 S 3794617
SimonsHenry Simons Henry 31 N 2336710
CarterGeorge Carter George 41 N 6240577
OliverAnn Oliver Ann 44 N 1843038
MurphyCatherine Murphy Catherine 60 N 6577182
KnightBruce Knight Bruce 29 N 3692990
AndersonStephen Anderson Stephen 50 N 7889775
WellsSara Wells Sara 25 N 3756471
GutierrezPaula Gutierrez Paula 22 S 8392960
GrahamHenry Graham Henry 25 S 9275825
DuncanRandy Duncan Randy 35 N 6817048
EllisStephen Ellis Stephen 61 S 4730616
ReedStephen Reed Stephen 54 S 3286559
DixonLawrence Dixon Lawrence 63 S 6752046
VasquezRandy Vasquez Randy 53 S 8479934
GrayBrandon Gray Brandon 22 N 3007577
HarrisonWalter Harrison Walter 59 S 9649661

```

Ilustración 38: Prueba ejecución y compilación Ejercicio 4 Parte 3

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell + x ... x
GrayBrandon Gray Brandon 22 N 3007577
HarrisonWalter Harrison Walter 59 S 0649661
HarperKeith Harper Keith 19 S 4925587
SpencerMichael Spencer Michael 39 S 5657086
FieldsJudy Fields Judy 55 N 7171048
PriceKelly Price Kelly 57 N 8033516
AllenThomas Allen Thomas 36 N 5839935
AllenThomas Allen Thomas 36 N 5839935
AllenThomas Allen Thomas 36 N 5839935
WardRoy Ward Roy 43 S 5332383
WatkinsJacqueline Watkins Jacqueline 28 N 5390060
GardnerNancy Gardner Nancy 35 N 6029397
HuntBruce Hunt Bruce 42 S 2363967
CarpenterRaymond Carpenter Raymond 18 S 9104623
DillLorraine Dill Lorraine 35 N 5399952
PowellJoe Powell Joe 40 S 5543257
PerezTodd Perez Todd 44 S 5718121
GarzaJudy Garza Judy 48 S 7498564
WelchRoy Welch Roy 19 S 9858399
PeterCarl Peter Carl 35 S 880459
SandersonPeter Sanderson Peter 18 S 9781496
SilvaHugo Silva Hugo 32 N 480000
CruzDonald Cruz Donald 56 N 8800976
GarzaHoward Garza Howard 45 S 6717867
EllisDennis Ellis Dennis 65 N 2222062
PerryTerry Perry Terry 33 N 0495086
CarrollJohn Carroll John 39 N 5398783
WashingtonFlorida Washington Florida 39 N 9372870
HarrisAaron Harris Aaron 22 S 3644495
HernandezAnthony Hernandez Anthony 50 S 2259598
EllisLillian Ellis Lillian 22 S 6857026
FergusonKatherine Ferguson Katherine 35 N 8796398
MarkhamMartha Markham Martha 45 N 2917248
LittleJohn LittleJohn 58 S 5390278
MartinDorothy Martin Dorothy 45 S 988655
McDonaldEmily McDonald Emily 58 N 5884440
BakerBeverly Baker Beverly 55 N 7954122
SanJorgeSanz Jorge 45 S 13000
ValverdeMaria Valverde Maria 25 N 15800
HenryKaren Henry Karen 43 S 9988163
CoxJoseph Cox Joseph 26 N 9345840
GibsonHeather Gibson Heather 26 N 4624382

```

Ilustración 39: Prueba ejecución y compilación Ejercicio 4 Parte 4

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell + x ... x
FoxJoseph Fox Joseph 26 N 9345810
GibsonHeather Gibson Heather 26 N 4624382
MorrisJohnny Morris Johnny 26 S 5519453
VasquezSamuel Vasquez Samuel 48 S 8620924
HarperGloria Harper Gloria 34 N 8455215
JonesKeather Jones Keather 32 S 5479171
PetersonJonathan Peterson Jonathan 54 S 2770152
AppleJohn Apple John 35 S 8739149
FieldsHenry Fields Henry 29 S 4177308
ScotthielenScott Scotthielen Scott 35 N 3837341
AllenJacqueline Allen Jacqueline 39 S 2850718
ArnoldDennis Arnold Dennis 28 S 8180255
ButlerSarah Butler Sarah 32 S 7590966
VerduzcoVeronica Verduzco Veronica 50000
StockmanStockman Karen 19 N 8712171
LopezTina Lopez Tina 33 S 7030058
CampbellJimmy Campbell Jimmy 36 N 1784750
CarpenterJoe Carpenter Joe 31 N 5840190
WilliamsonKelly Williamson Kelly 29 S 8773205
RodriguezZach Rodriguez Zach 54 S 2672192
HernandezTucker Hernandez Tucker 20 N 9211880
MorganCharles Morgan Charles 59 N 4575859
CarterKevin Carter Kevin 46 S 3296270
EvansLillian Evans Lillian 23 N 1351710
KelleyNancy Kelley Nancy 51 N 8430496
CruzJohn Cruz John 57 N 7120398
SamuelLinda Samuel Linda 50 S 3791871
McCoyJudy McCoy Judy 29 S 7132807
HernandezJoan Hernandez Joan 68 N 3189820
OlsonDonna Olson Donna 36 S 3984482
GibsonLillian Gibson Lillian 61 N 7558007
OrtizBarbara Ortiz Barbara 62 N 1523272
LagoLarry Lago Larry 18 N 1360787
HallRobert Hall Robert 40 N 9329840
HowellIrene Howell Irene 49 N 3624331
RussellJerry Russell Jerry 32 S 9032907
FowlerKaren Fowler Karen 57 S 7659325
HicksLarry Hicks Larry 26 N 4042723
BennettTimothy Bennett Timothy 55 S 2296219
WrightAnnie Wright Annie 63 S 7661156
KimJean Kim Jean 19 N 6779623

```

Ilustración 40: Prueba ejecución y compilación Ejercicio 4 Parte 5

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EXPLORER Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c Ejercicio4.c ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
a.exe Ejercicio4.c listaActores.csv
WrightAnnie Wright Annie 61 S 7661156
KimJean Kim Jean 19 N 6779817
ButlerIrene Butler Irene 28 N 1214382
JohnsonShirley Johnson Shirley 33 S 5622297
HallAnne Hall Anne 68 S 8445755
NicholsSandra Nichols Sandra 37 S 4611565
AtkinsLinda Atkins Linda 44 N 1994708
GilAriadna Gil Ariadna 49 S 19080
MoralesGraciela Morales Graciela 78 N 69000
NguyenHelen Nguyen Helen 42 N 247900
RobertsonPhilip Robertson Philip 22 N 8788834
JacksonAndrew Jackson Andrew 47 S 5190172
AndersonJames Anderson James 46 S 5467015
MatthewJack Matthew Jack 25 S 600100
FisherChristopher Fisher Christopher 18 N 3088753
AlvarezCarol Alvarez Carol 42 N 238047
RobertsonAshley Robertson Ashley 40 N 6557041
AbrilVictoria Abril Victoria 55 S 21800
deMashina deArmas Ana 25 N 00000
HenryDiana Henry Diana 60 N 6577182
Costoune Cox Tardie 20 S 29058
WoodPaula Woods Paula 31 N 7635826
ShawJeremy Shaw Jeremy 40 S 254920
WatkinsShirley Watkins Shirley 46 S 9873963
GeorgePhilip George Phillip 40 S 3639284
AndersonAudrey Anderson Audrey 40 S 5937278
CoxDennis Cox Dennis 29 N 9939270
SpencerGary Spencer Gary 20 N 3607852
BarnesBetty Barnes Betty 62 N 1918110
MorrisEric Morris Eric 59 N 6194496
Fernan-GomezFernando Fernan-Gomez Fernando 70 S 50000
CampbellDeneen Campbell Deneen 35 N 3509281
EllisSarah Ellis Martin 65 S 5339000
HarrisonJanice Harrison Janice 37 N 23293
JonesShawn Jones Shawn 45 S 8957529
LittleAlbert Little Albert 23 N 2672497
HughesEmily Hughes Emily 32 N 2638318
ReidJanice Reid Janice 28 N 768881
PriceJacqueline Price Jacqueline 31 S 3555362
WashingtonWayne Washington Wayne 60 N 5881999
HarrisonHarris Jason 40 N 156224

```

Ilustración 41: Prueba ejecución y compilación Ejercicio 4 Parte 6

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EXPLORER Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c Ejercicio4.c ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
a.exe Ejercicio4.c listaActores.csv
WashingtonWayne Washington Wayne 60 N 5881999
HarrisJason Harris Jason 40 N 156224
RiceEric Rice Eric 35 N 4599358
JordanMelissa Jordan Melissa 48 N 533164
ThompsonTina Thompson Tina 40 N 269567
HamillTina Hamill Tina 65 N 542886
PowellAaron Powell Aaron 54 S 6942169
ChapmanDebra Chapman Debra 34 N 1915364
ElliottSean Elliott Sean 47 N 3259955
DanielsJeremy Daniels Jeremy 26 N 1784184
MasoneWillie Masone Willie 62 N 2257761
FernandezKathy Fernandez Kathy 55 S 683121
AyayelenoAnaya Elena 52 N 25000
BishopCarlos Bishop Carlos 44 S 2176286
PhillipsDavid Phillips David 59 S 6115456
LynchJeffrey Lynch Jeffrey 53 N 764862
Rosewayne Rose Wayne 55 S 22575
GrantLorraine Grant Lorraine 58 S 8482420
ReyesLois Reyes Lois 26 S 8001073
YoungTodd Young Todd 20 N 5381821
KnightTeresa Knight Teresa 32 S 9104928
RomeroAaron Romero Aaron 61 N 338536
DavisPhilip Davis Phillip 63 S 6114845
HallZJohn Hall Z John 54 S 2176283
KerryConnie Warren Bonnie 34 N 7692283
SimsRoy Sims Roy 20 N 9641116
FloresLuis Flores Luis 45 S 2887949
GrantHarry Grant Harry 34 S 2221757
ColemanWalter Coleman Walter 33 N 7789678
PavezTeresa Pavez Teresa 62 S 280000
AyalaStephen Ayala Stephen 55 S 8069791
PierceStephen Pierce Stephen 44 S 8492446
RomeroIrene Romero Irene 47 N 584810
OliverAlbert Oliver Albert 63 N 9464421
ChavezRuby Chavez Ruby 53 S 4347285
BowmanSamuel Bowman Samuel 55 S 868632
KarpinskiJohn Karpinski John 25 N 2478102
SnyderJuan Snyder Juan 40 N 2175564
KelleyShawn Kelley Shawn 45 S 9783022
WeaverNorma Weaver Norma 34 N 8783276
PattersonKenneth Patterson Kenneth 57 S 4499566

```

Ilustración 42: Prueba ejecución y compilación Ejercicio 4 Parte 7

```

File Edit Selection View Go Run Terminal Help Ejercicio4c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4c Ejercicio4c > ...
EJERCICIO Ejercicio4c listaActores.csv
a.exe
WeaverNorma Weaver Norma 34 N 8783276
PattersonKenneth Patterson Kenneth 57 S 4499566
ArmstrongElizabeth Armstrong Elizabeth 43 S 3468082
CarterHenry Carter Henry 38 S 5527083
MarshallJason Marshall Jason 37 S 3289601
BanderasAntonio Banderas Antonio 59 H 60000
BrownstoneBrownstone Brownstone 39 S 5496490
KerryLorraine Kerry Lorraine 59 9124459
KelleWillie Kelley Willie 56 S 8829052
PerryFrank Perry Frank 44 N 6273840
GonzalezShirley Gonzalez Shirley 52 N 1803976
GarciaAmy Garcia Amy 52 S 5429937
ShawLori Shaw Lori 35 N 8285203
LewisKathy Lewis Kathy 77 H 6002931
GonzalezJustin Gonzalez Justin 52 N 5578047
OwensHildred Owens Mildred 22 S 117896
BrookWeather Brooks Heather 43 N 1219578
RobertsonAdam Robertson Adam 35 N 159320
DiegoGabino Diego Gabino 45 S 54000
SantosSamuel Santos Samuel 62 S 45800
HuntSamuel Hunt Samuel 27 N 6026956
OwenLori Owen Lori 35 N 4885915
GrantLarry Grant Larry 20 S 5440363
OlsonJason Olson Jason 57 S 5156194
OrtizDonald Ortiz Donald 35 S 1542442
RoyPatricia Roy Patricia 25 S 54000
AndreaRobin Andrews Robin 33 S 6571384
GonzalezDavid Gonzalez David 22 N 6576470
BanksGregory Banks Gregory 22 N 5193834
PierceAlan Pierce Alan 44 N 5552208
MartinezBrenda Martinez Brenda 43 S 3039620
LandaAlfredo Landa Alfredo 62 S 30000
HazelAudrey Hazel Audrey 44 N 7946187
LynchAdam Lynch Adam 22 S 3790995
CopperCheryl Copper Cheryl 39 S 8948679
HayesBobby Hayes Bobby 49 H 7027007
FowlerRonald Fowler Ronald 49 N 597017
KelleLawrence Kelle Lawrence 62 S 5477746
DiazDaniel Diaz Daniel 29 N 21628
GriffithSusan Griffin Susan 18 N 790802
JacksonJoshua Jackson Joshua 57 N 8128985

```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 13:28 02/06/2023

Ilustración 43: Prueba ejecución y compilación Ejercicio 4 Parte 8

```

File Edit Selection View Go Run Terminal Help Ejercicio4c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4c Ejercicio4c > ...
EJERCICIO Ejercicio4c listaActores.csv
a.exe
GriffinSusan Griffin Susan 18 N 790802
JacksonJoshua Jackson Joshua 57 N 8128985
HayesLouise Hayes Louise 63 N 5640973
RossRonald Ross Ronald 61 S 5151956
EllisSarah Ellis Sarah 19 S 5579771
LawsonRalph Lawson Ralph 46 N 3695772
HamiltonWilliam Hamilton William 54 N 2097552
HayesBobby Hayes Bobby 49 H 7027007
BellmanBell Juan Bell Juan 52 N 8557448
HunterBetty Hunter Betty 62 N 5807679
MartinEmily Martin Emily 21 S 8847667
RiveraJeffrey Rivera Jeffrey 50 N 710953
MontgomeryDennis Montgomery Dennis 56 N 1755453
EllisSarah Ellis Sarah 19 S 5579771
WellsSharon Wells Sharon 52 N 8963238
JamesCarl James Carl 65 S 1910481
RiceDorothy Rice Dorothy 49 N 2804637
MendozaWillie Mendoza Willie 34 N 375767
RossKevin Ross Kevin 57 N 1968769
LoxAngela Lox Angela 20 S 5447380
ShawSteven Shaw Steven 21 S 4621330
RoseWalter Rose Walter 65 S 5854228
MoralesRose Morales Rose 36 S 5618024
KennedyPhilip Kennedy Philip 31 N 7855591
KnightBetty Knight Betty 33 S 5447738
CoxFred Cox Fred 61 S 4881032
CoxFred Cox Fred 61 S 4881032
CarpenterEugene Carpenter Eugene 36 S 1205227
RobertsDonna Roberts Donna 22 N 2860789
TosarLuis Tosar Luis 40 S 33000
BrownRoy Brown Roy 32 N 9176033
HansenSteven Hansen Steven 64 S 8175372
ReynoldsPeter Reynolds Peter 24 N 6497837
GreenCarl Green Carl 39 S 56640495
HuntSteve Hunt Steve 33 S 1171658
LawrenceNicholas Lawrence Nicholas 41 S 7529321
SnyderChris Snyder Chris 20 N 3165961
ButlerKathleen Butler Kathleen 36 S 8889171
RobertsAaron Roberts Aaron 35 N 3569095
WillisJoyce Willis Joyce 24 S 4279232

```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 13:28 02/06/2023

Ilustración 44: Prueba ejecución y compilación Ejercicio 4 Parte 9

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell + v ... ^ x
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 ⌂ 13:28 02/06/2023

```

Output from the terminal:

```

RobertsSharon Roberts Aaron 35 N 3569095
WillisJoyce Willis Joyce 24 S 4279232
RomeroJesse Romero Jesse 29 S 7201671
MyersGloria Myers Gloria 64 S 7595238
BerryPatrick Berry Patrick 29 N 9076242
RichardsonFrank Richardson Frank 19 S 7863180
StevensDonald Richardson Donald 19 S 7961683
BlackDonald Black Ronald 27 S 7733281
HenryJennifer Henry Jennifer 47 S 749999
LopezKelly Lopez Kelly 48 N 1580589
OrtizJohn Ortiz John 57 S 4787954
ColeJuan Cole Juan 42 S 4583794
FordMarie Ford Marie 50 S 6501691
HallJennifer Hall Jennifer 34 S 3687196
NguyenNancy Nguyen Nancy 57 S 1297389
LarsonJudy Larson Judy 56 N 3126796
MorrisEdward Morris Edward 29 N 8914805
FloresTheresa Flores Theresa 21 N 3470219
ReedRichard Reed Richard 38 N 2643800
AttalaTheresa Attala Theresa 38 N 7403590
SchmidtTammy Schmidt Tammy 52 S 4739127
JordanMatthew Jordan Matthew 68 S 5936319
TurnerRussell Turner Russell 57 N 3855347
WheelerTammy Wheeler Tammy 49 S 3680788
GrahamDonald Graham Donald 18 S 5510603
GordonDonald Gordon Donald 18 S 5525405
ReedDonald Reid Donald 18 S 6369166
StewartMaria Stewart Maria 37 S 2353897
TuckerAshley Tucker Ashley 47 S 5075427
BerryThomas Berry Thomas 59 S 7325771
ScottRoger Scott Roger 48 S 513400
GutierrezEnriqu Gutierrez Enriqu 51 S 4604546
HammondCarol Hammond Carol 62 S 5830692
HowellHenry Howell Henry 58 S 9581303
FullerWilliam Fuller William 45 N 4677
CunninghamHeather Cunningham Heather 33 N 3505008
CarrDorothy Carr Dorothy 51 N 5755962
GrayAngela Gray Angela 18 S 5292100
DayLaura Day Laura 62 S 586013
StoneWayne Stone Wayne 27 S 1732260
RobertsDenise Roberts Denise 49 S 1571434

```

Ilustración 45: Prueba ejecución y compilación Ejercicio 4 Parte 10

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell + v ... ^ x
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 ⌂ 13:28 02/06/2023

```

Output from the terminal:

```

MunozLorena Munoz Lorena 27 S 4722960
RobertsDenise Roberts Denise 49 S 4571434
LynchDebra Lynch Debra 18 N 8641065
WheelerLarry Wheeler Larry 40 S 8463641
SchmidtRobert Schmidt Robert 30 S 8487258
NelsonCatherine Nelson Catherine 38 N 9657596
StanleyHarold Stanley Harold 65 S 4317683
HammondCarol Hammond Carol 62 S 5830692
FisherRaymond Fisher Raymond 64 N 6962930
CarterMatthew Carter Matthew 44 N 4768118
VasquezBrenda Vasquez Brenda 36 N 465488
HernandezTeresa Hernandez Teresa 43 N 2146684
YoungLillian Young Lillian 34 S 9465883
TuckerAshley Tucker Ashley 47 S 5075427
PalmerDoris Palmer Doris 54 S 4627208
PalmerJohnny Palmer Johnny 57 N 3056718
FranklinHarry Franklin Harry 56 N 1463708
HarperStephanie Harper Stephanie 30 N 1276027
CamaraJavier Camara Javier 45 N 33000
PalmerKimberly Palmer Kimberly 33 N 2817454
JonesLinda Jones Linda 40 N 3135902
BlackDoris Black Doris 24 S 4627208
GonzalesChris Gonzales Chris 58 S 9129647
JamesMarie James Marie 50 S 5016804
MitchellRobert Mitchell Robert 54 N 143836
LaneJessica Lane Jessica 33 N 3351201
FranklinJonathan Franklin Jonathan 58 S 5210008
LawrenceWoods Lawrence Woods 31 S 7118
PriceBrian Price Brian 49 N 3324824
SchmidtLaura Schmidt Laura 42 S 9978637
HansenBarbara Hansen Barbara 50 N 6044694
MillsCarolyn Mills Carolyn 51 N 3714662
MatthewsJames Matthews James 55 S 1585472
StanleyChris Stanley Chris 47 N 9810153
HallJessica Hall Jessica 18 N 4800467
HowardAmanda Howard Amanda 39 S 5010599
TurnerLaura Turner Laura 32 S 6664767
HoralesHenry Horales Henry 53 S 456333

```

Ilustración 46: Prueba ejecución y compilación Ejercicio 4 Parte 11

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
TurnerLaura Turner Laura 32 S 6664767
MoralesHenry Morales Henry 53 S 456333
CookKey Cook Amy 26 S 2981332
NguyenRoger Nguyen Roger 55 N 980925
StevensClarence Stevens Clarence 55 S 89515
GrahamJohn Graham John 42 N 4123288
MohammedAngelique MohammedAngelique 39 S 7661159
AparicioLorena AparicioLorena Rafaela 70 S 34000
BanksCarol Banks Carol 43 N 8877217
GarciaAndrea Garcia Andrea 29 N 9401861
WilsonGloria Wilson Gloria 27 S 5307664
JohnstonWanda Johnston Wanda 18 S 7661156
HenryJerry Henry Jerry 61 N 8274742
HudsonHudson Hudson Harry 61 N 8339250
StewartRussell Stewart Russell 64 N 5665980
ReedBrandon Reed Brandon 61 N 9316107
RobinsonRichard Robinson Richard 46 N 9522709
GrantGregory Grant Gregory 24 S 517382
BrooksSteve Brooks Steve 42 S 4734488
HudsonHudson Hudson Harry 61 N 8339250
YoungDoris Young Doris 29 N 9401861
SullivanCarol Sullivan Carol 65 S 6009866
FrazierJeremy Frazier Jeremy 32 S 8697522
DuncanLeAnn Duncan LeAnn 58 S 6900970
JordanAndres Jordan Andres 50 S 398876
BaldwinCynthia Baldwin Cynthia 32 S 8249982
CruzPenelope Cruz Penelope 48 S 50000
PhillipsJason Phillips Jason 52 N 6753267
RuizJoshua Ruiz Joshua 57 N 801722
ChavezAnn Chavez Ann 21 N 9652468
HawkinsMartha Hawkins Martha 50 S 4725699
EstebanLorraine Esteban Lorraine 46 N 6168089
MorenoTamy Moreno Tammy 43 N 4183102
HudsonSteve Hudson Steve 45 N 8409744
RuizJoshua Ruiz Joshua 30 N 8537713
LawsonTheresa Lawson Theresa 52 S 1441430
BrooksCynthia Brooks Cynthia 46 S 789426
AlvarezDouglas Alvarez Douglas 32 S 6500279
ChapmanFrank Chapman Frank 55 N 8247698
SanchezAlice Sanchez Alice 61 S 3937133

```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 13:28 02/06/2023

Ilustración 47: Prueba ejecución y compilación Ejercicio 4 Parte 12

```

File Edit Selection View Go Run Terminal Help Ejercicio4.c - Ejercicio4 - Visual Studio Code
OPEN EDITORS Ejercicio4.c Ejercicio4.c ...
EJERCICIO Ejercicio4.c listaActores.csv
a.exe
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
ChapmanFrank Chapman Frank 55 N 8247698
SanchezAlice Sanchez Alice 61 S 3937133
ShawSean Shaw Sean 57 S 1899800
FieldsSarah Fields Sarah 23 N 9478154
FisherFred Fisher Fred 55 N 6054116
CarrollJacqueline Carroll Jacqueline 20 S 5959818
Hendersontonyan Hendersontonyan Ryan 50 S 5584895
LeeJennifer Lee Jennifer 21 N 11800
AndersonStan Anderson Stan 63 N 2422561
WhiteJean White Jean 45 S 1193020
CampbellTodd Campbell Todd 62 N 5751385
LeeJennifer Lee Jennifer 36 S 6681551
WeaverJack Weaver Jack 49 N 6884548
WoodburyDoreen Woodbury Doreen 46 N 11800
BurgosJavier Burgos Javier 45 S 20000
CarrollDiana Carroll Diana 53 S 2263210
AndrewsLarry Andrews Larry 24 S 1317835
OwenRobert Owen Robert 22 N 3960821
CooperAlan Cooper Alan 52 N 1916584
AndersonKathy Anderson Kathy 55 N 595684
MitchellAntonio Mitchell Antonio 23 N 5936014
HowellDiana Howell Diana 25 S 8399674
RyanKathryn Ryan Kathryn 60 S 5555464
FullerBeverly Fuller Beverly 55 N 2041400
CastilloElizabeth Castillo Elizabeth 50 S 9223335
JordanPhyllis Jordan Phyllis 57 S 1386264
ConradLorraine Conrad Lorraine 46 S 480141
HarrisonKevin Harrison Kevin 55 N 6079140
CunninghamJeffrey Cunningham Jeffrey 34 N 951723
ShawMarie Shaw Marie 24 N 172827
JenkinsLouise Jenkins Louise 63 N 7063017
RichardsMarilyn Richards Marilyn 55 N 6190833
SnyderMartin Snyder Martin 50 S 5552448
MyersThomas Myers Thomas 37 S 5426989
StewartPaul Stewart Paul 20 N 6483189
AlexanderMarilyn Alexander Marilyn 30 S 8747867
GutierrezChristine Gutierrez Christine 51 N 191748
HallKathy Hall Kathy 47 N 7538781
ArmstrongRachel Armstrong Rachel 56 S 1722189
LynchEarl Lynch Earl 23 N 6968719
GonzalesKelly Gonzales Kelly 57 N 1513756

```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⌂ C Win32 13:29 02/06/2023

Ilustración 48: Prueba ejecución y compilación Ejercicio 4 Parte 13

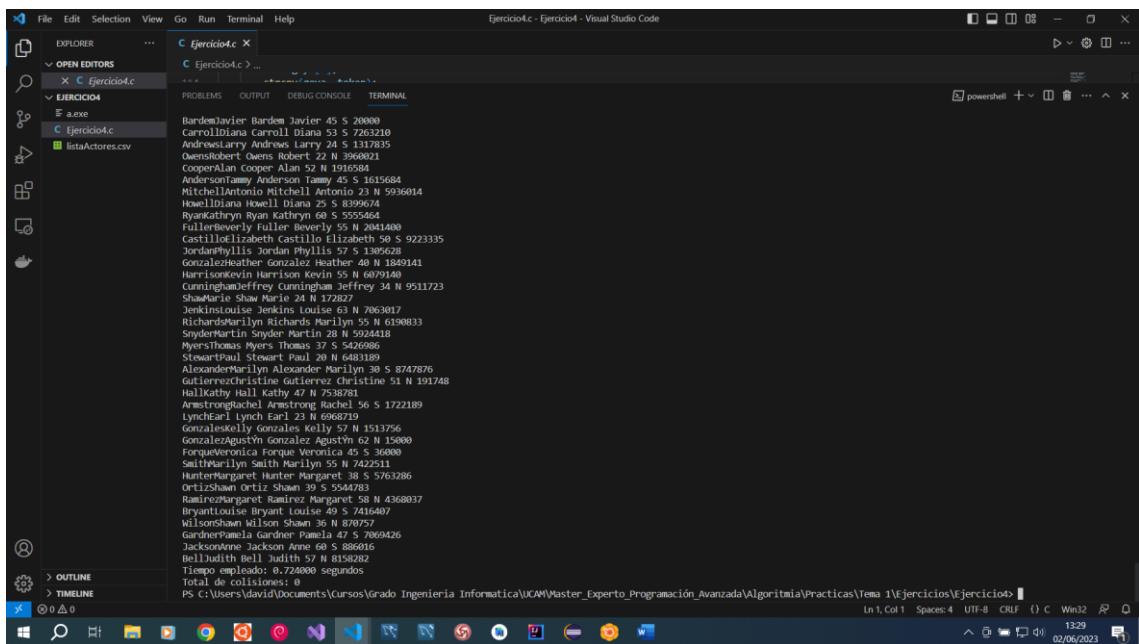


Ilustración 49: Prueba ejecución y compilación Ejercicio 4 Parte 14

9. Preguntas

I. Explicar las decisiones y el código más relevante de cada ejercicio:

Ejercicio 1: En este ejercicio se implementó un algoritmo de dispersión perfecta utilizando la función hash. Se utilizó una tabla hash con un tamaño fijo de 50 elementos. Se decidió utilizar una estructura de datos tipo arreglo para representar la tabla hash. El código más relevante en este ejercicio es la función hash, que calcula la posición en la tabla hash para cada clave.

Ejercicio 2: En este ejercicio se implementó un algoritmo de prueba dependiente de clave utilizando la función hash. Se utilizó una tabla hash con un tamaño fijo de 50 elementos. El código más relevante en este ejercicio es la función search, que busca un registro en la tabla hash utilizando la función hash y realizando una comparación de claves.

Ejercicio 3: En este ejercicio se implementó un algoritmo de encadenamiento utilizando listas enlazadas para resolver las colisiones. Se utilizó una tabla hash con un tamaño fijo de 50 elementos, donde cada elemento es una lista enlazada que almacena los registros que colisionan en la misma posición de la tabla. El código más relevante en este ejercicio es la función insert, que inserta un registro en la tabla hash y maneja las colisiones creando una nueva lista enlazada o agregando el registro a una lista existente.

Ejercicio 4: En este ejercicio se incluyen el uso de una estructura de datos para representar los registros, la implementación de una función hash simple, el empleo de la técnica de prueba dependiente de clave para manejar colisiones y asegurar un recorrido completo del algoritmo, y la inclusión de funciones de inicialización y visualización para facilitar la verificación y seguimiento del proceso.

II. ¿Qué función H(x) se ha utilizado? ¿Y qué función G(x)?

En los ejercicios anteriores, se utilizó la función H(x) para calcular la posición en la tabla hash. La función H(x) es una función de dispersión que toma la clave del registro como entrada y devuelve un valor entero que representa la posición en la tabla hash. La implementación exacta de la función H(x) no se muestra en el código proporcionado, por lo que no se puede determinar con certeza qué función específica se utilizó.

III. ¿Cuántos accesos a la tabla hay que realizar para recuperar los registros de los siguientes actores: "Javier Bardem"; "Dennis Gomez"; "Louise Jenkins"?

Para recuperar los registros de los actores "Javier Bardem", "Dennis Gomez" y "Louise Jenkins", se debe utilizar la función de búsqueda en la tabla hash. En el caso de una función de dispersión perfecta o una prueba dependiente de clave, solo se realizará un acceso a la tabla para recuperar el registro si se encuentra en la posición calculada por la función hash. En el caso del encadenamiento, se puede requerir más de un acceso a la tabla si los registros colisionan en la misma posición y se encuentran en la misma lista enlazada.

IV. Eliminar un registro que se haya insertado sin colisión. Probar a recuperar un registro que sí haya producido colisión con el que se acaba de eliminar. ¿Se recupera correctamente?

En el caso de eliminar un registro que se haya insertado sin colisión, simplemente se marca el registro como eliminado o se elimina de la tabla hash, dependiendo de la implementación específica. La eliminación de un registro sin colisión no debería tener ningún efecto en los demás registros.

Si se intenta recuperar un registro que haya producido colisión con el registro eliminado, la recuperación dependerá de la técnica utilizada para resolver las colisiones. En el caso del encadenamiento, el registro colisionado se mantendrá en la lista enlazada correspondiente y se podrá recuperar correctamente utilizando la función de búsqueda.

Los resultados empíricos obtenidos pueden variar dependiendo de los datos de entrada y las implementaciones específicas de las funciones hash y las estructuras de datos utilizadas. Es importante realizar múltiples pruebas y obtener promedios para obtener resultados más representativos. En general, los resultados empíricos deberían estar en línea con los conceptos y propiedades estudiados en el tema de Hashing, como la resolución de colisiones y la eficiencia en términos de tiempo de acceso y espacio utilizado.