

Algoritmia

Practica 2:

Backtraking

David Piñuel Bosque



2023

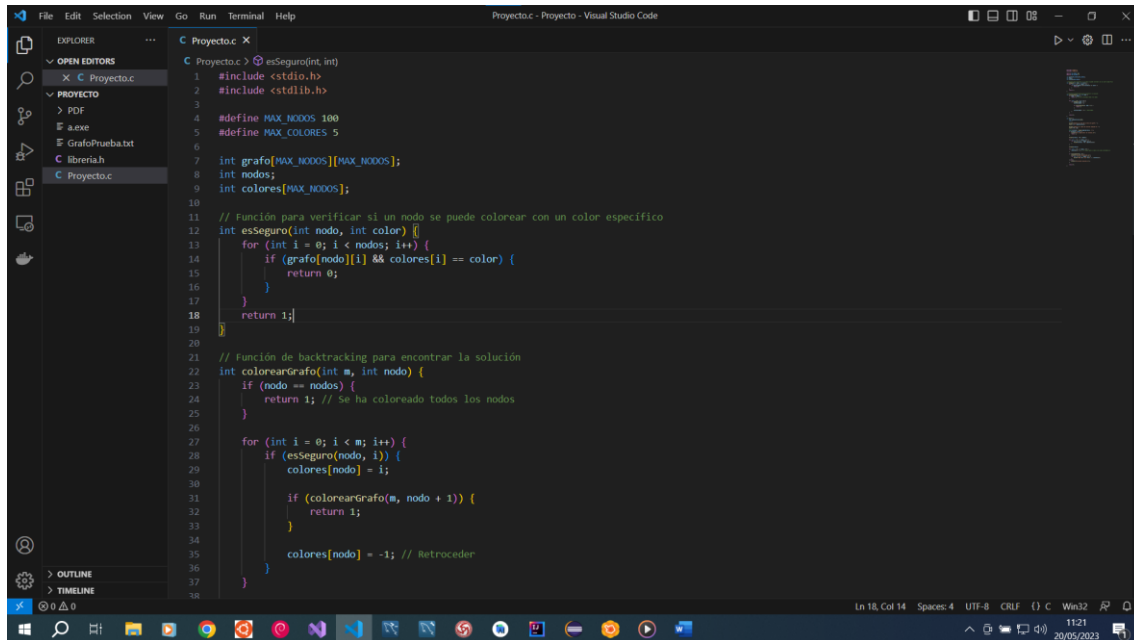
Índice

1. Código Fuente del Programa de backtraking.....	4
2. Ejemplo de prueba del programa compilado y ejecutado.	8
3. Aclaraciones y comentarios.	10

Índice Ilustración

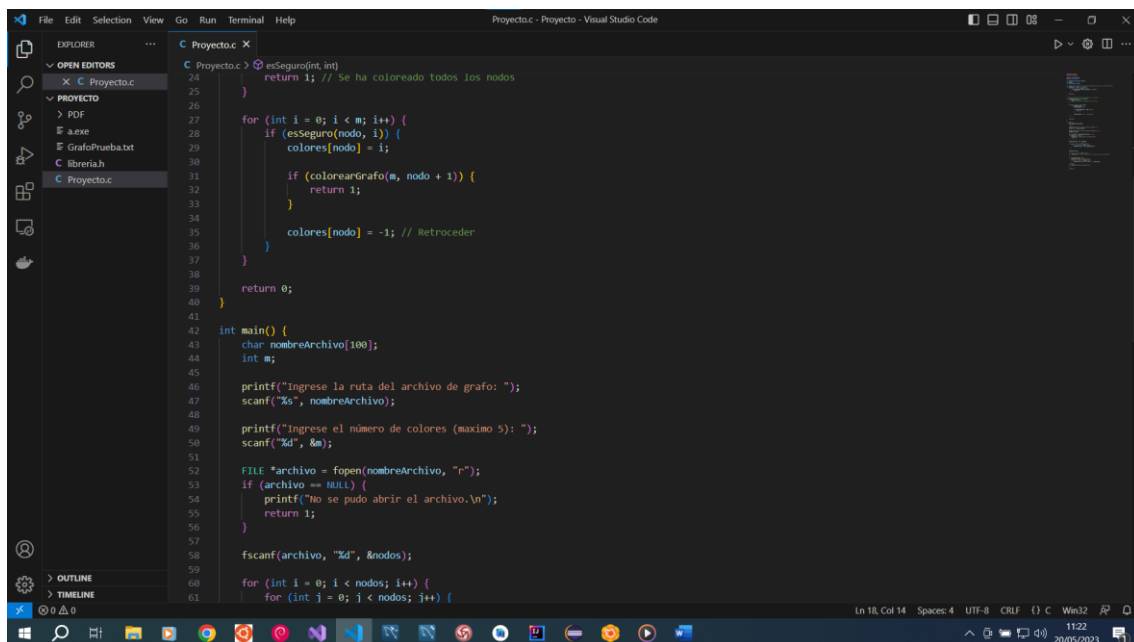
<i>Ilustración 1: Código Fuente Parte 1</i>	<i>3</i>
<i>Ilustración 2: Código Fuente Parte 2</i>	<i>3</i>
<i>Ilustración 3: Código Fuente Parte 3</i>	<i>4</i>
<i>Ilustración 4: Código Fuente Parte 4</i>	<i>4</i>
<i>Ilustración 5: Código 2 Fuente Parte 1</i>	<i>5</i>
<i>Ilustración 6: Código 2 Fuente Parte 2</i>	<i>5</i>
<i>Ilustración 7: Código 2 Fuente Parte 2</i>	<i>6</i>
<i>Ilustración 8: Código 2 Fuente Parte 3</i>	<i>6</i>
<i>Ilustración 9: Código 2 Fuente Parte 4</i>	<i>7</i>
<i>Ilustración 10: Ejemplo Prueba Parte 1</i>	<i>7</i>
<i>Ilustración 11: Ejemplo Prueba Parte 2</i>	<i>8</i>
<i>Ilustración 12: Ejemplo 2 Prueba Parte 1</i>	<i>8</i>
<i>Ilustración 13: Ejemplo 2 Prueba Parte 3</i>	<i>9</i>

1. Código Fuente del Programa de backtracking.



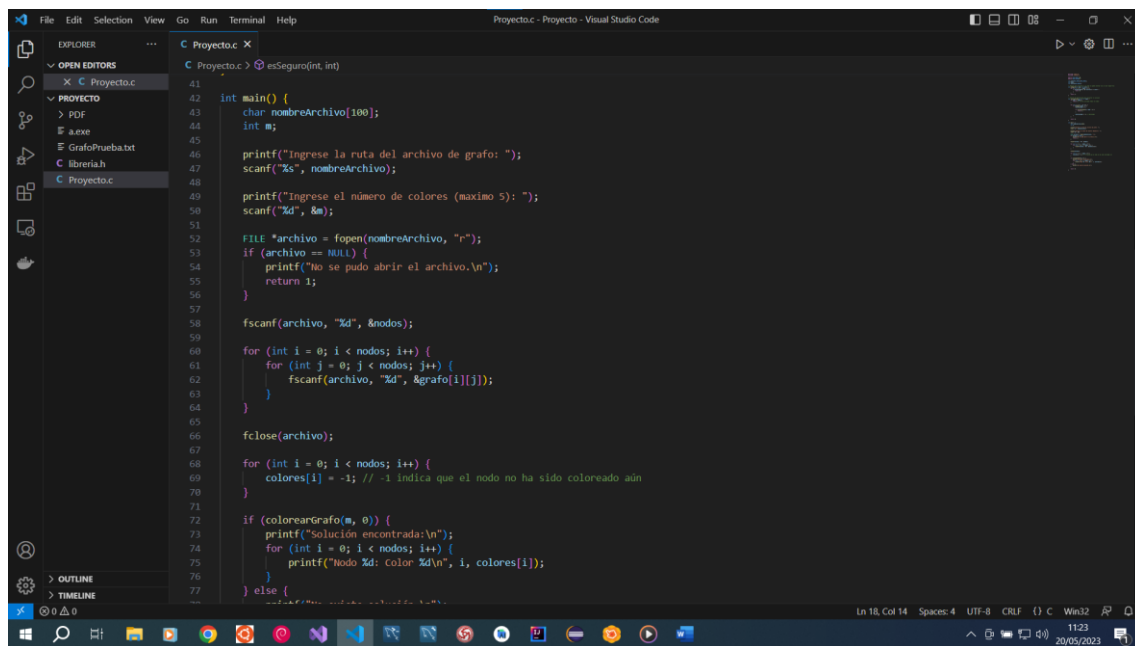
```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX_NODOS 100
5  #define MAX_COLORES 5
6
7  int grafo[MAX_NODOS][MAX_NODOS];
8  int nodos;
9  int colores[MAX_NODOS];
10
11 // Función para verificar si un nodo se puede colorear con un color específico
12 int esSeguro(int nodo, int color) {
13     for (int i = 0; i < nodos; i++) {
14         if (grafo[nodo][i] && colores[i] == color) {
15             return 0;
16         }
17     }
18     return 1;
19 }
20
21 // Función de backtracking para encontrar la solución
22 int colorearGrafo(int m, int nodo) {
23     if (nodo == nodos) {
24         return 1; // Se ha coloreado todos los nodos
25     }
26
27     for (int i = 0; i < m; i++) {
28         if (esSeguro(nodo, i)) {
29             colores[nodo] = i;
30
31             if (colorearGrafo(m, nodo + 1)) {
32                 return 1;
33             }
34
35             colores[nodo] = -1; // Retroceder
36         }
37     }
38 }
```

Ilustración 1: Código Fuente Parte 1



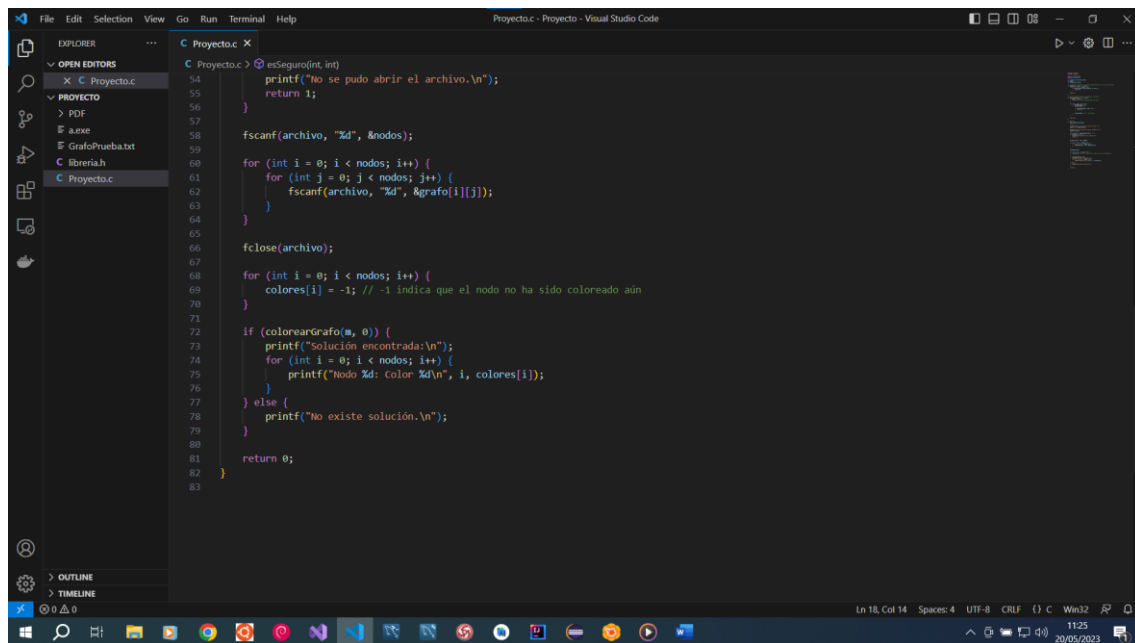
```
24     return 1; // Se ha coloreado todos los nodos
25 }
26
27 for (int i = 0; i < m; i++) {
28     if (esSeguro(nodo, i)) {
29         colores[nodo] = i;
30
31         if (colorearGrafo(m, nodo + 1)) {
32             return 1;
33         }
34
35         colores[nodo] = -1; // Retroceder
36     }
37 }
38
39 return 0;
40 }
41
42 int main() {
43     char nombreArchivo[100];
44     int m;
45
46     printf("Ingrese la ruta del archivo de grafo: ");
47     scanf("%s", nombreArchivo);
48
49     printf("Ingrese el número de colores (maximo 5): ");
50     scanf("%d", &m);
51
52     FILE *archivo = fopen(nombreArchivo, "r");
53     if (archivo == NULL) {
54         printf("No se pudo abrir el archivo.\n");
55         return 1;
56     }
57
58     fscanf(archivo, "%d", &nodos);
59
60     for (int i = 0; i < nodos; i++) {
61         for (int j = 0; j < nodos; j++) {
```

Ilustración 2: Código Fuente Parte 2



```
1  File Edit Selection View Go Run Terminal Help
2  Proyecto.c - Proyecto - Visual Studio Code
3
4  EXPLORER
5  ...
6  OPEN EDITORS
7  X C Proyecto.c
8  PROJECTO
9  > PDF
10 E a.exe
11 E GrafoPueba.txt
12 C libreria.h
13 C Proyecto.c
14
15 C Proyecto.c
16
17 41
18 42 int main() {
19 43     char nombreArchivo[100];
20 44     int m;
21 45
22 46     printf("Ingrese la ruta del archivo de grafo: ");
23 47     scanf("%s", nombreArchivo);
24 48
25 49     printf("Ingrese el número de colores (maximo 5): ");
26 50     scanf("%d", &m);
27 51
28 52     FILE *archivo = fopen(nombreArchivo, "r");
29 53     if (archivo == NULL) {
30 54         printf("No se pudo abrir el archivo.\n");
31 55         return 1;
32 56     }
33 57
34 58     fscanf(archivo, "%d", &nodos);
35 59
36 60     for (int i = 0; i < nodos; i++) {
37 61         for (int j = 0; j < nodos; j++) {
38 62             fscanf(archivo, "%d", &grafo[i][j]);
39 63         }
40 64     }
41 65
42 66     fclose(archivo);
43 67
44 68     for (int i = 0; i < nodos; i++) {
45 69         colores[i] = -1; // -1 indica que el nodo no ha sido coloreado aún
46 70     }
47 71
48 72     if (colorearGrafo(m, 0)) {
49 73         printf("Solución encontrada:\n");
50 74         for (int i = 0; i < nodos; i++) {
51 75             printf("Nodo %d: Color %d\n", i, colores[i]);
52 76         }
53 77     } else {
54 78         printf("No existe solución.\n");
55 79     }
56 80
57 81     return 0;
58 82 }
59 83
```

Ilustración 3: Código Fuente Parte 3



```
1  File Edit Selection View Go Run Terminal Help
2  Proyecto.c - Proyecto - Visual Studio Code
3
4  EXPLORER
5  ...
6  OPEN EDITORS
7  X C Proyecto.c
8  PROJECTO
9  > PDF
10 E a.exe
11 E GrafoPueba.txt
12 C libreria.h
13 C Proyecto.c
14
15 C Proyecto.c
16
17 54
18 55     printf("No se pudo abrir el archivo.\n");
19 56     return 1;
20 57 }
21 58
22 59     fscanf(archivo, "%d", &nodos);
23 60
24 61     for (int i = 0; i < nodos; i++) {
25 62         for (int j = 0; j < nodos; j++) {
26 63             fscanf(archivo, "%d", &grafo[i][j]);
27 64         }
28 65     }
29 66
30 67     fclose(archivo);
31 68
32 69     for (int i = 0; i < nodos; i++) {
33 70         colores[i] = -1; // -1 indica que el nodo no ha sido coloreado aún
34 71     }
35 72
36 73     if (colorearGrafo(m, 0)) {
37 74         printf("Solución encontrada:\n");
38 75         for (int i = 0; i < nodos; i++) {
39 76             printf("Nodo %d: Color %d\n", i, colores[i]);
40 77         }
41 78     } else {
42 79         printf("No existe solución.\n");
43 80     }
44 81
45 82     return 0;
46 83 }
```

Ilustración 4: Código Fuente Parte 4

The screenshot shows the Visual Studio Code editor with the file 'Proyecto2.c' open. The code includes standard headers, defines constants for MAX_ACTORES, MAX_ATRIBUTOS, and MAX_NOMBRE, and defines an Actor struct. It also includes a function leerActores that reads data from a CSV file into an array of Actor structs.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAX_ACTORES 10
6 #define MAX_ATRIBUTOS 5
7 #define MAX_NOMBRE 100
8
9 typedef struct {
10     char nombre[MAX_NOMBRE];
11     char apellido[MAX_NOMBRE];
12     int edad;
13     char premioGoya;
14     int cache;
15 } Actor;
16
17 void leerActores(Actor actores[MAX_ACTORES]) {
18     FILE *archivo;
19     char linea[MAX_NOMBRE * MAX_ATRIBUTOS];
20     char *token;
21     int i = 0;
22
23     archivo = fopen("listaActores.csv", "r");
24     if (archivo == NULL) {
25         printf("Error al abrir el archivo.\n");
26         exit(1);
27     }
28
29     while (fgets(linea, sizeof(linea), archivo)) {
30         token = strtok(linea, ",");
31         strcpy(actores[i].apellido, token);
32
33         token = strtok(NULL, ",");
34         strcpy(actores[i].nombre, token);
35
36         token = strtok(NULL, ",");
37         actores[i].edad = atoi(token);
38     }
39 }
```

Ilustración 5: Código 2 Fuente Parte 1

The screenshot shows the continuation of the C code in 'Proyecto2.c'. It includes a function cumpleRangoEdad that checks if an actor's age is within a specified range, and a function backtracking that performs a recursive search for the best casting combination based on age range and Goya awards.

```
28
29
30 while (fgets(linea, sizeof(linea), archivo)) {
31     token = strtok(linea, ",");
32     strcpy(actores[i].apellido, token);
33
34     token = strtok(NULL, ",");
35     strcpy(actores[i].nombre, token);
36
37     token = strtok(NULL, ",");
38     actores[i].edad = atoi(token);
39
40     token = strtok(NULL, ",");
41     actores[i].premioGoya = token[0];
42
43     token = strtok(NULL, ",");
44     actores[i].cache = atoi(token);
45     i++;
46 }
47 fclose(archivo);
48
49
50 int cumpleRangoEdad(int edad, int rangoEdadMin, int rangoEdadMax) {
51     return (edad >= rangoEdadMin && edad <= rangoEdadMax);
52 }
53
54 int backtracking(Actor actores[MAX_ACTORES], Actor seleccionados[MAX_ACTORES], int numActoresSeleccionados, int numActoresRestantes, int rangoEdadMin, int rangoEdadMax, int cacheTotal, int goyaAcumulados) {
55     if (numActoresSeleccionados == numActoresRestantes) {
56         if (cumpleRangoEdad(seleccionados[numActoresSeleccionados - 1].edad, rangoEdadMin, rangoEdadMax) && goyaAcumulados >= cacheTotal) {
57             *mejorCache = cacheTotal;
58             memcpy(mejorCasting, seleccionados, sizeof(Actor) * numActoresSeleccionados);
59             return 1;
60         }
61         return 0;
62     }
63
64     int i;
```

Ilustración 6: Código 2 Fuente Parte 2

The screenshot shows the Visual Studio Code editor with the file `Proyecto2.c` open. The code implements a backtracking function `backtracking` that explores different combinations of actors to find a valid solution. The function takes an array of actors, the current selection, the number of actors selected, the number of actors remaining, and the current age range. It checks if the current selection is valid (age range and total cost) and if it leads to a solution. If not, it backtracks and tries the next actor. The `mostrarCasting` function is also shown, which prints the details of the selected actors.

```
54 int backtracking(Actor actores[MAX_ACTORES], Actor seleccionados[MAX_ACTORES], int numActoresSeleccionados, int numActoresRestantes, int rangoEdadMin,
55                 int rangoEdadMax, int cacheTotal) {
56     if (numActoresSeleccionados == numActoresRestantes) {
57         *mejorCache = cacheTotal;
58         memcpy(mejorCasting, seleccionados, sizeof(Actor) * numActoresSeleccionados);
59         return 1;
60     }
61     return 0;
62 }
63
64 int i;
65 for (i = 0; i < MAX_ACTORES; i++) {
66     if (actores[i].nombre[0] != '\0') {
67         if (cumpleRangoEdad(actores[i].edad, rangoEdadMin, rangoEdadMax) && goyaAcumulados + actores[i].cache <= cacheTotal) {
68             seleccionados[numActoresSeleccionados] = actores[i];
69             if (backtracking(actores, seleccionados, numActoresSeleccionados + 1, numActoresRestantes, rangoEdadMin, rangoEdadMax, goyaAcumulados + actores[i].cache)) {
70                 return 1;
71             }
72             seleccionados[numActoresSeleccionados].nombre[0] = '\0'; // Deshacer la selección
73         }
74     }
75 }
76
77 return 0;
78 }
79
80 void mostrarCasting(Actor casting[MAX_ACTORES], int numActores) {
81     printf("Casting:\n");
82     int i;
83     for (i = 0; i < numActores; i++) {
84         printf("Actor %d: %s %s\n", i + 1, casting[i].nombre, casting[i].apellido);
85         if (casting[i].premioGoya == 'S') {
86             printf("Tiene premio Goya.\n");
87         } else {
88             printf("No tiene premio Goya.\n");
89         }
90     }
91 }
```

Ilustración 7: Código 2 Fuente Parte 2

The screenshot shows the Visual Studio Code editor with the file `Proyecto2.c` open. The code implements the `mostrarCasting` function, the `mostrarResultado` function, and the `main` function. The `mostrarCasting` function prints the details of the selected actors. The `mostrarResultado` function prints the result of the search, including the total cost and the number of actors selected. The `main` function initializes the variables and calls the `backtracking` function to find a solution.

```
80 void mostrarCasting(Actor casting[MAX_ACTORES], int numActores) {
81     printf("Casting:\n");
82     int i;
83     for (i = 0; i < numActores; i++) {
84         printf("Actor %d: %s %s\n", i + 1, casting[i].nombre, casting[i].apellido);
85         if (casting[i].premioGoya == 'S') {
86             printf("Tiene premio Goya.\n");
87         } else {
88             printf("No tiene premio Goya.\n");
89         }
90     }
91 }
92
93 void mostrarResultado(Actor casting[MAX_ACTORES], int numActores, int cacheTotal) {
94     printf("\n*** Resultado ***\n");
95     if (numActores > 0) {
96         mostrarCasting(casting, numActores);
97         printf("coste total del casting: %d\n", cacheTotal);
98     } else {
99         printf("No hay solución que cumpla con las condiciones.\n");
100     }
101 }
102
103 int main() {
104     Actor actores[MAX_ACTORES];
105     Actor seleccionados[MAX_ACTORES];
106     int numActores;
107     int rangoEdadMin, rangoEdadMax;
108     int cacheTotal;
109     int mejorCache = -1;
110     Actor mejorCasting[MAX_ACTORES];
111     leerActores(actores);
112
113     printf("Ingrese el número de actores en el casting: ");
114     scanf("%d", &numActores);
```

Ilustración 8: Código 2 Fuente Parte 3

The screenshot shows the Visual Studio Code editor with the file 'Proyecto2.c' open. The code is a C program for a casting selection algorithm. It includes headers for 'stdio.h' and 'stdlib.h', and defines 'MAX_ACTORES' as 10. The 'main' function prompts the user for the number of actors, the minimum and maximum age range, the total number of Goya awards, and the best casting. It then calls 'backtracking' and 'mostrarResultado' functions. The code is as follows:

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

#define MAX_ACTORES 10

int main() {
    Actor actores[MAX_ACTORES];
    Actor seleccionados[MAX_ACTORES];
    int numActores;
    int rangoEdadMin, rangoEdadMax;
    int cacheTotal;
    int mejorCache = -1;
    Actor mejorCasting[MAX_ACTORES];
    leerActores(actores);

    printf("Ingrese el número de actores en el casting: ");
    scanf("%d", &numActores);

    printf("Ingrese el rango de edad mínimo: ");
    scanf("%d", &rangoEdadMin);

    printf("Ingrese el rango de edad máximo: ");
    scanf("%d", &rangoEdadMax);

    printf("Ingrese la cantidad total de premios Goya: ");
    scanf("%d", &cacheTotal);

    backtracking(actores, seleccionados, 0, numActores, rangoEdadMin, rangoEdadMax, 0, cacheTotal, &mejorCache, mejorCasting);

    mostrarResultado(mejorCasting, numActores, mejorCache);

    return 0;
}
```

Ilustración 9: Código 2 Fuente Parte 4

2. Ejemplo de prueba del programa compilado y ejecutado.

The screenshot shows the Visual Studio Code editor with the 'TERMINAL' view active. The terminal displays the output of the program execution. The user has run 'gcc .\Proyecto.c' and then './a.exe'. The program prompts for the number of actors (3), the minimum age (0), the maximum age (5), and the total number of Goya awards (3). It then displays the solution: 'Solución encontrada: Nodo 0: color 0, Nodo 1: color 0, Nodo 2: color 0, Nodo 3: color 0, Nodo 4: color 0'.

```
PS C:\Users\david\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programación_Avanzada\Algoritmia\Practicas\Tema 2\Proyecto> gcc .\Proyecto.c
PS C:\Users\david\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programación_Avanzada\Algoritmia\Practicas\Tema 2\Proyecto> .\a.exe
Ingrese la ruta del archivo de grafo: .\grafoPrueba.txt
Ingrese el n|mero de colores (maximo 5): 3
Soluci|n encontrada:
Nodo 0: color 0
Nodo 1: color 0
Nodo 2: color 0
Nodo 3: color 0
Nodo 4: color 0
PS C:\Users\david\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Experto_Programación_Avanzada\Algoritmia\Practicas\Tema 2\Proyecto>
```

Ilustración 10: Ejemplo Prueba Parte 1

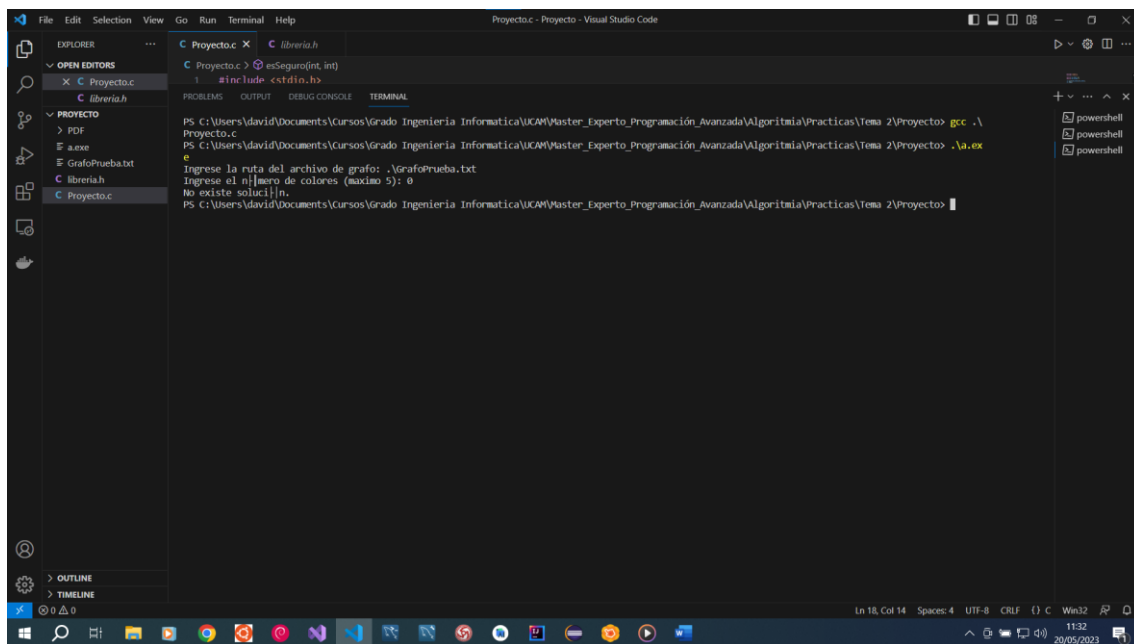


Ilustración 11: Ejemplo Prueba Parte 2

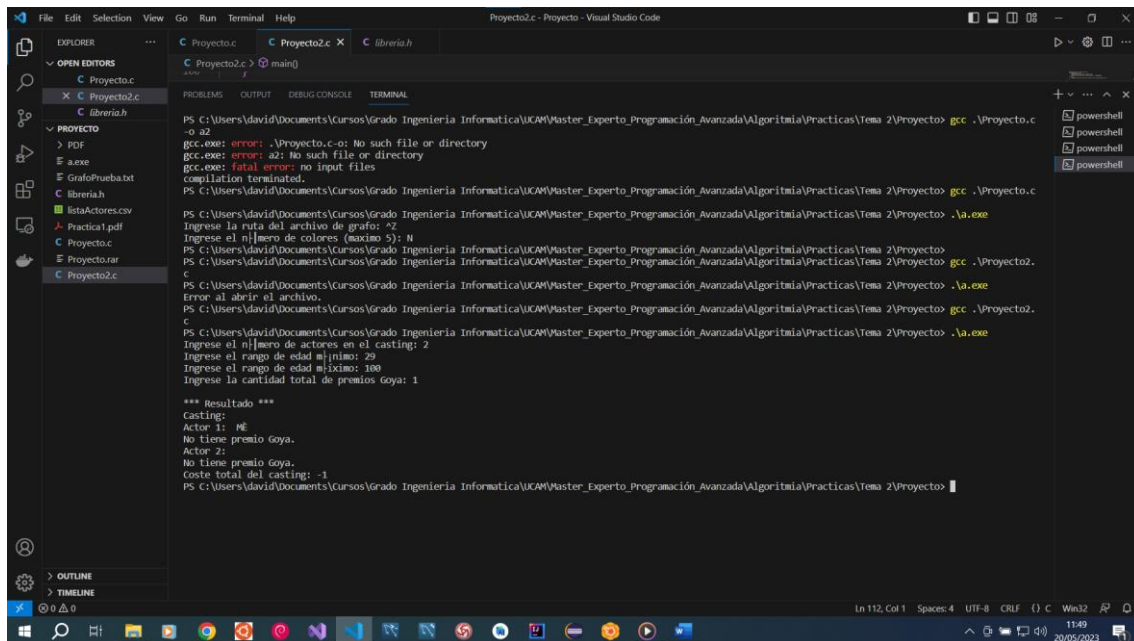


Ilustración 12: Ejemplo 2 Prueba Parte 1

The screenshot shows the Visual Studio Code interface with a C++ project named 'Proyecto2.c'. The Explorer sidebar on the left shows the project structure, including files like 'Proyecto2.c', 'libreria.h', 'a.exe', 'GrafoPrueba.txt', 'libreria.h', 'ListaActores.csv', 'Practica1.pdf', 'Proyecto.c', 'Proyecto.rar', and 'Proyecto2.c'. The main editor window displays the source code for 'Proyecto2.c', which includes a function 'mostrarResultado' and a main function. The terminal window at the bottom shows the program's output, which includes prompts for user input and the resulting calculations and status messages for each actor.

```
Proyecto2.c - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C Proyecto.c C Proyecto2.c X libreria.h
C Proyecto2.c > mostrarResultado(Actor [MAX_ACTORES], int, int)
1 #include <stdio.h>
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Ingrese el rango de edad m[1]: 1
Ingrese el rango de edad m[2]: 100
Ingrese la cantidad total de premios Goya: 5
*** Resultado ***
Casting:
Actor 1: H
No tiene premio Goya.
Coste total del casting: -1
PS C:\Users\david\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Especializado\Algoritmos\Practicas\Tema 2\Proyecto> .\a.exe
Ingrese el número de actores en el casting: 11
Ingrese el rango de edad m[1]: 1
Ingrese el rango de edad m[2]: 100
Ingrese la cantidad total de premios Goya: N
*** Resultado ***
Casting:
Actor 1: N
No tiene premio Goya.
Actor 2:
No tiene premio Goya.
Actor 3: 00h?: ;
No tiene premio Goya.
Actor 4: + 8
No tiene premio Goya.
Actor 5: P 0
No tiene premio Goya.
Actor 6: r +
No tiene premio Goya.
Actor 7: < n
No tiene premio Goya.
Actor 8: + I
No tiene premio Goya.
Actor 9: 00h?: ;
No tiene premio Goya.
Actor 10: 10 10h?o
No tiene premio Goya.
Actor 11: *Ida 10
No tiene premio Goya.
Coste total del casting: -1
PS C:\Users\david\Documents\Cursos\Grado Ingenieria Informatica\UCAM\Master_Especializado\Algoritmos\Practicas\Tema 2\Proyecto>
```

Ilustración 13: Ejemplo 2 Prueba Parte 3

3. Aclaraciones y comentarios.

He dedido realizarla de esta manera, dado que los recursos eran escasos. Cuando le pides el número si no esta no encuentra la solución y si esta intenta calcular la solución hasta encontrarla entonces la muestra.