



ESTRUCTURAS DE DATOS

DESARROLLAR METODOS FALTANTES

Programación I ITDSC3-003-SDN

NOMBRE: *Sabrina Amparo Valdez |2025-1320*

FACILITADOR: *Starling Alfredo Germosén Reynoso*

SECCIÓN: *#1 lunes*

CARRERA: *Desarrollo de Software*



complete los métodos de la clase "contactes" o desarrolle su propia solución que cumpla con el requerimiento 1 del proyecto "Agenda Personal".

```

4 //sabrina amparo valdez 20251326
5 0 referencias
6 class Program
7 {
8     0 referencias
9     static void Main(string[] args)
10    {
11        Console.WriteLine("Mi Agenda Perrón");
12        Console.WriteLine("Bienvenido a tu lista de contactes");
13
14        Dictionary<int, string> names = new Dictionary<int, string>();
15        Dictionary<int, string> phones = new Dictionary<int, string>();
16        Dictionary<int, string> emails = new Dictionary<int, string>();
17        List<string> addresses = new List<string>();
18        List<int> ids = new List<int>();
19
20        bool running = true;
21        while (running)
22        {
23            Console.WriteLine("\n--- MENÚ PRINCIPAL ---");
24            Console.Write("1. Agregar Contacto ");
25            Console.Write("2. Ver Contactos ");
26            Console.Write("3. Buscar Contactos ");
27            Console.Write("4. Modificar Contacto ");
28            Console.Write("5. Eliminar Contacto ");
29            Console.WriteLine("6. Salir");
30            Console.Write("Elige una opción: ");
31
32            try
33            {
34                int choice = Convert.ToInt32(Console.ReadLine());
35
36                switch (choice)
37                {
38                    case 1:
39                        AddContact(ref names, ref phones, ref emails, ref addresses, ref ids);
40                        break;
41                    case 2:
42                        ViewContacts(ref names, ref phones, ref emails, ref addresses, ref ids);
43                        break;
44                    case 3:
45                        SearchContact(ref names, ref phones, ref emails, ref addresses, ref ids);
46                        break;
47                    case 4:
48                        EditContact(ref names, ref phones, ref emails, ref addresses, ref ids);
49                        break;
50                    case 5:
51                        DeleteContact(ref names, ref phones, ref emails, ref addresses, ref ids);
52                        break;
53                    case 6:
54                        running = false;
55                        Console.WriteLine("¡Hasta luego!");
56                        break;
57                    default:
58                        Console.WriteLine("Opción no válida");
59                        break;
60                }
61            }
62        }
63    }
64 }
```

```

59 }
60 catch (FormatException)
61 {
62     Console.WriteLine("Error: Por favor ingresa un número válido.");
63 }
64 }
65 }
66
67 1 referencia
68 static void AddContact(ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses, ref List<int> ids)
69 {
70     Console.WriteLine("\n--- AGREGAR CONTACTO ---");
71     Console.WriteLine("Vamos a agregar ese contacte que te trae loco.");
72
73     try
74     {
75         int id = ids.Count > 0 ? ids.Max() + 1 : 1;
76         ids.Add(id);
77
78         Console.Write("Digite el Nombre: ");
79         var name = Console.ReadLine();
80         if (string.IsNullOrEmpty(name))
81         {
82             Console.WriteLine("Error: El nombre no puede estar vacío.");
83             ids.Remove(id);
84             return;
85         }
86         names.Add(id, name);
87
88         Console.Write("Digite el Teléfono: ");
89         var telefono = Console.ReadLine();
90         phones.Add(id, telefono);
91
92         Console.Write("Digite el Email: ");
93         var email = Console.ReadLine();
94         emails.Add(id, email);
95
96         Console.Write("Digite la dirección: ");
97         var address = Console.ReadLine();
98         addresses.Add(address);
99
100        Console.WriteLine("¡Contacto agregado exitosamente!");
101    }
102    catch (Exception ex)
103    {
104        Console.WriteLine($"Error al agregar contacto: {ex.Message}");
105    }
106 }
107
108 4 referencias
109 static void ViewContacts(ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses, ref List<int> ids)
110 {
111     Console.WriteLine("\n--- LISTA DE CONTACTOS ---");
112 }
```



```

108 Console.WriteLine("\n--- LISTA DE CONTACTOS ---");
109
110
111 if (ids.Count == 0)
112 {
113     Console.WriteLine("No hay contactos en la agenda.");
114     return;
115 }
116
117 Console.WriteLine("Id\tNombre\t\tTeléfono\t\tEmail\t\t\tDirección");
118 Console.WriteLine("-----");
119
120 foreach (var id in ids)
121 {
122     if (names.ContainsKey(id) && phones.ContainsKey(id) && emails.ContainsKey(id))
123     {
124         string direccion = id <= addresses.Count ? addresses[id - 1] : "No especificada";
125         Console.WriteLine($"{id}\t{names[id]}\t\t{phones[id]}\t\t{emails[id]}\t\t{direccion}");
126     }
127 }
128
129
130 1 referencia
131 static void EditContact(ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses, ref List<int> ids)
132 {
133     Console.WriteLine("\n--- MODIFICAR CONTACTO ---");
134
135     if (ids.Count == 0)
136     {
137         Console.WriteLine("No hay contactos para modificar.");
138         return;
139     }
140
141     ViewContacts(ref names, ref phones, ref emails, ref addresses, ref ids);
142
143     try
144     {
145         Console.Write("Digite el Id de Contacto Para Editar: ");
146         int idSeleccionado = Convert.ToInt32(Console.ReadLine());
147
148         if (!ids.Contains(idSeleccionado))
149         {
150             Console.WriteLine("Error: El ID especificado no existe.");
151             return;
152         }
153
154         var nombreSeleccionado = names[idSeleccionado];
155         var telefonoSeleccionado = phones[idSeleccionado];
156         var emailSeleccionado = emails[idSeleccionado];
157         string direccionSeleccionada = addresses[idSeleccionado - 1];
158
159         Console.Write($"El nombre actual es: {nombreSeleccionado}, Digite el Nuevo Nombre: ");
160         var nuevoNombre = Console.ReadLine();
161         if (!string.IsNullOrEmpty(nuevoNombre))
162         {
163             names[idSeleccionado] = nuevoNombre;
164         }
165     }
166 }

```

```

165 Console.Write($"El Teléfono actual es: {telefonoSeleccionado}, Digite el Nuevo Teléfono: ");
166 var nuevoTelefono = Console.ReadLine();
167 phones[idSeleccionado] = nuevoTelefono;
168
169 Console.Write($"El Email actual es: {emailSeleccionado}, Digite el Nuevo Email: ");
170 var nuevoEmail = Console.ReadLine();
171 emails[idSeleccionado] = nuevoEmail;
172
173 Console.Write($"La dirección actual es: {direccionSeleccionada}, Digite la nueva dirección: ");
174 var nuevaDireccion = Console.ReadLine();
175 addresses[idSeleccionado - 1] = nuevaDireccion;
176
177 Console.WriteLine("Contacto modificado exitosamente!");
178 }
179 catch (FormatException)
180 {
181     Console.WriteLine("Error: Por favor ingresa un ID válido.");
182 }
183 catch (Exception ex)
184 {
185     Console.WriteLine($"Error al modificar contacto: {ex.Message}");
186 }
187
188
189 1 referencia
190 static void DeleteContact(ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses, ref List<int> ids)
191 {
192     Console.WriteLine("\n--- ELIMINAR CONTACTO ---");
193
194     if (ids.Count == 0)
195     {
196         Console.WriteLine("No hay contactos para eliminar.");
197         return;
198     }
199
200     ViewContacts(ref names, ref phones, ref emails, ref addresses, ref ids);
201
202     try
203     {
204         Console.Write("Digite el Id de Contacto Para Eliminar: ");
205         int idSeleccionado = Convert.ToInt32(Console.ReadLine());
206
207         if (!ids.Contains(idSeleccionado))
208         {
209             Console.WriteLine("Error: El ID especificado no existe.");
210             return;
211         }
212
213         Console.WriteLine($"¿Está seguro que desea eliminar el contacto '{names[idSeleccionado]}'?");
214         Console.Write("1. Sí, 2. No: ");
215         int opcion = Convert.ToInt32(Console.ReadLine());
216     }
217 }

```




```

215         if (opcion == 1)
216         {
217             names.Remove(idSeleccionado);
218             phones.Remove(idSeleccionado);
219             emails.Remove(idSeleccionado);
220             addresses.RemoveAt(idSeleccionado - 1);
221             ids.Remove(idSeleccionado);
222
223             Console.WriteLine("¡Contacto eliminado exitosamente!");
224         }
225         else
226         {
227             Console.WriteLine("Eliminación cancelada.");
228         }
229     }
230 }
231
232 catch (FormatException)
233 {
234     Console.WriteLine("Error: Por favor ingresa un número válido.");
235 }
236
237 catch (Exception ex)
238 {
239     Console.WriteLine($"Error al eliminar contacto: {ex.Message}");
240 }
241
242 1 referencia
243 static void SearchContact(ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses, ref List<int> ids)
244 {
245     Console.WriteLine("\n--- BUSCAR CONTACTO ---");
246
247     if (ids.Count == 0)
248     {
249         Console.WriteLine("No hay contactos para buscar.");
250         return;
251     }
252
253     Console.WriteLine("1. Buscar por ID");
254     Console.WriteLine("2. Buscar por Nombre");
255     Console.Write("Elige una opción de búsqueda: ");
256
257     try
258     {
259         int opcionBusqueda = Convert.ToInt32(Console.ReadLine());
260
261         if (opcionBusqueda == 1)
262         {
263             ViewContacts(ref names, ref phones, ref emails, ref addresses, ref ids);
264             Console.Write("Digite el Id de Contacto Para Mostrar: ");
265             int idSeleccionado = Convert.ToInt32(Console.ReadLine());
266
267             if (ids.Contains(idSeleccionado))
268             {
269                 MostrarDetalleContacto(idSeleccionado, ref names, ref phones, ref emails, ref addresses);
270             }
271             else
272             {
273                 Console.WriteLine("Error: El ID especificado no existe.");
274             }
275         }
276         else if (opcionBusqueda == 2)
277         {
278             Console.Write("Digite el nombre o parte del nombre a buscar: ");
279             string nombreBuscado = Console.ReadLine().ToLower();
280
281             bool encontrado = false;
282             Console.WriteLine("\n--- RESULTADOS DE BÚSQUEDA ---");
283
284             foreach (var id in ids)
285             {
286                 if (names.ContainsKey(id) && names[id].ToLower().Contains(nombreBuscado))
287                 {
288                     MostrarDetalleContacto(id, ref names, ref phones, ref emails, ref addresses);
289                     encontrado = true;
290                 }
291             }
292
293             if (!encontrado)
294             {
295                 Console.WriteLine("No se encontraron contactos con ese nombre.");
296             }
297         }
298         else
299         {
300             Console.WriteLine("Opción de búsqueda no válida.");
301         }
302     }
303     catch (FormatException)
304     {
305         Console.WriteLine("Error: Por favor ingresa un número válido.");
306     }
307 }
308
309 2 referencias
310 static void MostrarDetalleContacto(int id, ref Dictionary<int, string> names, ref Dictionary<int, string> phones, ref Dictionary<int, string> emails, ref List<string> addresses)
311 {
312     Console.WriteLine($"--- DETALLES DEL CONTACTO ID: {id} ---");
313     Console.WriteLine($"Nombre: {names[id]}");
314     Console.WriteLine($"Teléfono: {phones[id]}");
315     Console.WriteLine($"Email: {emails[id]}");
316     Console.WriteLine($"Dirección: {addresses[id - 1]}");
317     Console.WriteLine("-----");
318 }

```

En este caso, agregué mejoras al código para hacerlo más robusto y fácil de usar, incluyendo manejo de errores con try-catch para evitar que el programa se cierre pues inesperadamente, también validaciones que verifican si existen contactos antes de realizar operaciones, una función de búsqueda mejorada que permite encontrar contactos tanto por ID como por nombre, y mensajes más claros que en mi experiencia pues me han ayudado durante todo el proceso.