



Universidad Mariano Gálvez de Guatemala
Sede de Villa Nueva, Guatemala

Ingeniería en Sistemas de la Información y Ciencia Computacional

Tema:
Laboratorio #6

Curso: Programación I
Docente: Ing. Carlos Alejandro Arias

Estudiante: Pablo Sebastián Quan Montenegro
Carné: 5090-23-2625

08/03/2024

Introducción

El siguiente código en C++ presenta un programa que ofrece diversas funcionalidades, un menú interactivo con múltiples opciones. Se incluyen distintas áreas como una biblioteca virtual, registro de estudiantes, una calculadora simple, gestión bancaria, y un registro de películas en la taquilla. El programa utiliza clases para organizar la información relacionada con libros, estudiantes, películas y operaciones bancarias.

El menú principal del programa permite al usuario elegir entre estas opciones, cada una representada por una función específica. Las funciones incluyen la organización de libros en una biblioteca, la visualización de información sobre estudiantes de una universidad, operaciones básicas de una calculadora, funciones de un cajero automático y la presentación de películas en la taquilla.

El código también incorpora mecanismos para repetir acciones, como volver al menú principal o realizar operaciones adicionales específicamente en la calculadora, proporcionando una experiencia interactiva para el usuario.

Código Comentado

```
1  #include <iostream>
2  #include <string>
3  #include <locale>
4
5  void repeticion();
6
7  void administradorBiblioteca();
8  void informacionEstudiantes();
9  void calculadora();
10 void menuCajero();
11 void peliculasTaquilla();
12
13 using namespace std;
14
15 char respuesta;
16 bool repetir;
```

Parte inicial del código donde se agregaron las bibliotecas correspondientes:

- iostream: se utiliza para cout y cin, entre otras funciones relacionadas con las cadenas de texto.
- string: se utiliza para poder hacer uso de funciones de getline(cin, (variable)).
- locale: es una biblioteca que permite el uso de las tildes y demás caracteres que no se encuentran disponibles por default.

Luego se encuentran las funciones que se utilizan:

- repeticion(): se encarga de permitir volver a repetir alguna otra función del menú.
- administradorBiblioteca(): muestra los libros disponibles en una biblioteca.
- informacionEstudiantes(): muestra información acerca de estudiantes universitarios.
- calculadora(): como su nombre lo indica, es una calculadora siempre que incluye suma, resta, multiplicación y división.
- menuCajero(): es donde muestra las distintas opciones que ofrece un cajero y permite realizarlas.
- peliculasTaquilla(): muestras las películas que se encuentra en taquilla.

Luego se tiene a 2 tipos de variables globales, ya que serán útiles con la función de repeticion().

Class Biblioteca

```
17
18     class Biblioteca {
19         private:
20             string m_autor;
21             string m_titulo;
22             string m_anio;
23             string m_isbn;
24         public:
25             Biblioteca(string titulo, string autor, string anio, string isbn)
26                 : m_titulo(titulo), m_autor(autor), m_anio(anio), m_isbn(isbn) {}
27
28             void mostrarNuevoLibro() {
29                 cout << "Titulo del libro: " << m_titulo << endl;
30                 cout << "Autor del libro: " << m_autor << endl;
31                 cout << "Anio de publicacion: " << m_anio << endl;
32                 cout << "ISBN del libro: " << m_isbn;
33             }
34     };
```

La clase tiene como atributos:

- autor (quién escribió el libro)
- título (nombre de la obra)
- año (representado como anio por el tema de los caracteres especiales, fecha que se estrenó)
- isbn (código único que lo identifica)

Después se creó su constructor para inicializar los atributos durante la instancia de creación de un objeto. Por último, un método donde enseña la información sobre el libro.

Función void administradorBiblioteca();

```
237
238     void administradorBiblioteca() {
239         Biblioteca libro1("El corazón de la piedra", "José María García López ", "2013", "978-84-939750-7-4");
240         Biblioteca libro2("Salmos de vísperas", "Esteban Hernández Castelló", "2003", "84 932914 8 X");
241         Biblioteca libro3("La música en las catedrales españolas del Siglo de Oro", "Robert Stevenson", "1993", "84-206-8562-3");
242         Biblioteca libro4("Tomás Luis de Victoria: A guide to research", "Eugene Casjen Cramer", "1998", "0 8153 2096 5");
243
244         cout << "\tBienvenido al mostrador de libros de la biblioteca\n\n";
245         cout << "Libros disponibles: \n\n";
246
247         libro1.mostrarNuevoLibro();
248         cout << "\n-----\n";
249         libro2.mostrarNuevoLibro();
250         cout << "\n-----\n";
251         libro3.mostrarNuevoLibro();
252         cout << "\n-----\n";
253         libro4.mostrarNuevoLibro();
254         cout << "\n-----\n";
255         system("pause");
256     }
```

En esta función se crearon los 4 objetos (o libros) de la clase Biblioteca, se da la bienvenida y luego se muestran en pantalla los libros disponibles a través del método de la misma clase.

Class Estudiante

```
58  class Estudiante {
59      private:
60          string m_nombre;
61          string m_apellido;
62          int m_edad;
63          string m_carrera;
64          string m_salon;
65          float m_promedio;
66      public:
67          Estudiante(string nombre, string apellido, int edad, string carrera, string salon, float promedio)
68              : m_nombre(nombre), m_apellido(apellido), m_edad(edad), m_carrera(carrera), m_salon(salon), m_promedio(promedio) {}
69
70          void mostrarInformacionEstudiante() {
71              cout << "Nombres: " << m_nombre << endl;
72              cout << "Apellidos: " << m_apellido << endl;
73              cout << "Edad: " << m_edad << endl;
74              cout << "Carrera: " << m_carrera << endl;
75              cout << "Salon: " << m_salon << endl;
76              cout << "Promedio: " << m_promedio;
77          }
78      };
79 
```

La clase tiene como atributos:

- nombre (únicamente nombres de los estudiantes)
- apellido (únicamente apellidos de los estudiantes)
- edad (los años que tienen)
- carrera (código único que lo identifica)
- salón (edificio y sala donde se encuentra la clase)
- promedio (promedio entre todas las materias que cursan)

Después se creó su constructor para inicializar los atributos durante la instancia de creación de un objeto. Por último, un método donde enseña la información sobre los estudiantes.

Función void informacionEstudiantes();

```
258  void informacionEstudiantes() {
259      Estudiante estudiante1("Pablo Sebastián", "Quan Montenegro", 19, "Ingenieria en Sistemas", "B-101", 90.6);
260      Estudiante estudiante2("Jorge Mario", "Folgar Martínez", 18, "Arquitectura", "B-105", 90.5);
261      Estudiante estudiante3("Luis Eduardo", "Guevara Juárez", 18, "Arte", "A-202", 90.4);
262      Estudiante estudiante4("Esteban Mauricio", "Contreras Molina", 20, "Derecho", "C-301", 90);
263
264      cout << "\tBienvenido al registro de estudiantes de la UMG\n\n";
265      cout << "Estudiantes disponibles: \n\n";
266
267      estudiante1.mostrarInformacionEstudiante();
268      cout << "\n-----\n";
269      estudiante2.mostrarInformacionEstudiante();
270      cout << "\n-----\n";
271      estudiante3.mostrarInformacionEstudiante();
272      cout << "\n-----\n";
273      estudiante4.mostrarInformacionEstudiante();
274      cout << "\n-----\n";
275      system("pause");
276  }
277 
```

En esta función se crearon los 4 objetos (o estudiantes) de la clase Estudiante, se da la bienvenida y luego se muestran en pantalla los alumnos disponibles a través del método de la misma clase.

Class CalculadoraSimple

```
79
80     class CalculadoraSimple {
81         private:
82             int m_opcion;
83             float m_num1, m_num2;
84         public:
85             CalculadoraSimple(int opcion, float num1, float num2)
86                 : m_opcion(opcion), m_num1(num1), m_num2(num2) {}
87
88             void mostrarResultado() {
89                 cout << "El resultado de la ";
90                 switch (m_opcion) {
91                     case 1:
92                         cout << "suma es: " << m_num1 + m_num2 << endl;
93                         break;
94                     case 2:
95                         cout << "resta es: " << m_num1 - m_num2 << endl;
96                         break;
97                     case 3:
98                         cout << "multiplicacion es: " << m_num1 * m_num2 << endl;
99                         break;
100                    case 4:
101                        cout << "division es: " << m_num1 / m_num2 << endl;
102                        break;
103                }
104            };
105        };
106    
```

La clase tiene como atributos:

- opcion (sirve para conocer el tipo de operación que se realizará)
- num1 (primer número a operar)
- num2 (segundo número a operar)

Tanto num1 como num2 dependen de la posición en el caso de una resta o división.

Después se creó su constructor para inicializar los atributos durante la instancia de creación de un objeto, en este caso es una operación. Según el número de opción que se ingresa, es la operación que se va a realizar con num1 y num2. Y el único método disponible es la impresión del resultado.

Función void calculadora();

```
277
278     void calculadora() {
279         bool repetirCalculo;
280         char respuestaCalculo;
281         do {
282             int operacion, numero1, numero2;
283             do {
284                 system("cls");
285                 cout << "\tCalculadora Basica\n\n";
286                 cout << "1. Suma\n2. Resta\n3. Multiplicacion\n4. Division\n\n";
287                 cout << "Elige la operacion que deseas realizar: ";
288                 cin >> operacion;
289             } while (operacion < 1 or operacion > 4);
290
291             system("cls");
292             cout << "Elige el numero 1: ";
293             cin >> numero1;
294             cout << "Elige el numero 2: ";
295             cin >> numero2;
296
297             CalculadoraSimple operar(operacion, numero1, numero2);
298             operar.mostrarResultado();
299             cout << endl;
300 }
```

Esta es la primera parte de la calculadora, donde hay 2 do whiles, el primero es donde si se ingresa una operación que no sea de las disponibles (1,2,3,4), se repetirá la pregunta.

Luego de haber ingresado una opción válida, se le pide al usuario el ingresar el valor que le dará a num1 y num2.

Después de haber obtenido esos valores, se crea un objeto de la clase CalculadoraSimple pasando como atributos los valores por referencia que decidió el usuario y la opción.

```
300
301     do {
302         cout << "Desea realizar otra operacion de la calculadora? S/N: ";
303         cin >> respuestaCalculo;
304     } while (respuestaCalculo != 'n' and respuestaCalculo != 'N' and respuestaCalculo != 'S' and respuestaCalculo != 's');
305
306     if (respuestaCalculo == 's' or respuestaCalculo == 'S') {
307         repetirCalculo = true;
308     }
309     else if (respuestaCalculo == 'n' or respuestaCalculo == 'N') {
310         repetirCalculo = false;
311     }
312
313 } while (repetirCalculo == true);
```

En esta segunda parte es solo otro do while donde si no ingresa “n”, “N”, o “S”, “s”, volverá a repetirse la pregunta. Y el while que se observa al final es parte del primer do while donde si se elige la opción que da como resultado el repetirse == true, entonces la calculadora volverá a ejecutarse.

Class GestionBanco

```
106
107     class GestionBanco {
108         private:
109             string m_titular;
110             float m_saldo;
```

La clase GestionBanco tiene como atributos:

- m_titular (propietario)
- m_saldo (cantidad de dinero en la cuenta)

```
111     public:
112         GestionBanco(string titular, float saldo)
113             : m_titular(titular), m_saldo(saldo) {}
```

Se procede a realizar su constructor.

```
114
115     void retirarDinero() {
116         int opcionDinero;
117
118         do {
119             system("cls");
120             cout << "\tRetiro de dinero\n\n";
121             cout << "Opciones a retirar: \n\n";
122
123             cout << "1. Q50\n";
124             cout << "2. Q100\n";
125             cout << "3. Q500\n";
126             cout << "4. Q1000\n";
127             cout << "5. Ingresar cantidad\n\n";
128
129             cout << "Digite la opcion: ";
130             cin >> opcionDinero;
131
132         } while (opcionDinero < 1 or opcionDinero > 5);
```

El primer método comprende en ofrecer de ofrecerle al usuario la opción de retiro de dinero. Así como en un cajero automático, se ofrecen cantidades predeterminadas para retirar dinero y una opción extra donde se le permite al usuario ingresar una cantidad.

Además, ese bloque se encuentra encerrado dentro de un do while donde si no se elige una opción del 1 al 5, se volverá a repetir.

```
133  
134     if ((opcionDinero == 1 and m_saldo < 50) or (opcionDinero == 2 and m_saldo < 100) or (opcionDinero == 3 and m_saldo < 500) or (opcionDinero == 4 and m_saldo < 1000)) {  
135         system("cls");  
136         cout << "Fondos insuficientes disponibles en la cuenta del usuario\n\n";  
137         return;
```

Después tenemos una condicional donde si el saldo de la cuenta es menor a alguna de las opciones predeterminadas elegidas, entonces se enseña un mensaje de fondos insuficientes y regresa al usuario al menú del cajero automático.

```
139  
140     switch (opcionDinero) {  
141         case 1:  
142             m_saldo -= 50;  
143             break;  
144         case 2:  
145             m_saldo -= 100;  
146             break;  
147         case 3:  
148             m_saldo -= 500;  
149             break;  
150         case 4:  
151             m_saldo -= 1000;  
152             break;  
153         case 5:  
154             system("cls");  
155             int cantRetirar;  
156             cout << "Ingrese la cantidad a retirar: ";  
157             cin >> cantRetirar;  
158             if (cantRetirar > m_saldo) {  
159                 system("cls");  
160                 cout << "Fondos insuficientes disponibles en la cuenta del usuario\n\n";  
161                 return;  
162             }  
163             m_saldo -= cantRetirar;  
164             break;  
165         }  
166     }  
167     cout << "\n\tSe ha retirado de manera correcta\n\n";  
168 }
```

La última parte comprende de un switch, donde según la opción seleccionada, se restará a los fondos de la cuenta del usuario para retirar el dinero. En el case 5, donde el usuario elige la cantidad, en dado caso la cantidad que seleccione es mayor al de los fondos de la cuenta, entonces no se dará la transacción, caso contrario se restará al saldo.

Por último, se muestra un mensaje que todo ha salido correctamente.

```
169  
170     void depositarDinero() {  
171         float deposito;  
172         system("cls");  
173         cout << "\tDepósito de dinero\n\n";  
174         cout << "Ingrese la cantidad a depositar en su cuenta: ";  
175         cin >> deposito;  
176         m_saldo += deposito;  
177         cout << "\n\tEl depósito se ha realizado de manera correcta\n\n";  
178     }
```

Este método tiene como propósito permitir al usuario depositar una cantidad.

```
179
180     void consultarEstadoCuenta() const{
181         system("cls");
182         cout << "\tResumen estado de cuenta\n\n";
183         cout << "Nombre del usuario: " << m_titular << endl;
184         cout << "Fondos de la cuenta: Q" << m_saldo << endl;
185     }
186 };
187
```

El último método es solo presentar los datos del usuario, nombre del titular y el saldo disponible en la cuenta. Se le agregó un const ya que en ningún momento se cambia algún dato, su finalidad solo es mostrar.

Función void MenuCajero();

```
315
316     void menuCajero() {
317         int opcionUsuario;
318         bool decisionRepetir;
319         int opcionRepetir;
320
321         GestionBanco usuario1("Pablo Sebastian Quan Montenegro", 500);
322
323         do {
324             do {
325                 system("cls");
326                 cout << "\tBienvenido al menu del cajero\n\n";
327                 cout << "1. Consultar estado de cuenta\n";
328                 cout << "2. Retirar dinero\n";
329                 cout << "3. Depositar dinero\n";
330                 cout << "4. Salir\n\n";
331                 cout << "Digite la opcion a elegir: ";
332                 cin >> opcionUsuario;
333             } while (opcionUsuario < 1 or opcionUsuario > 5);
```

Este void su finalidad es ser el menú de un cajero. En la primera parte se le solicita al usuario ingrese alguna de las opciones disponibles, en dado caso elige otra que no existe, se volverá a realizar la pregunta.

Además, también se creó al objeto usuario1, con un nombre del titular y un saldo inicial en su cuenta.

```
334
335         switch (opcionUsuario) {
336             case 1:
337                 usuario1.consultarEstadoCuenta();
338                 break;
339             case 2:
340                 usuario1.retirarDinero();
341                 break;
342             case 3:
343                 usuario1.depositarDinero();
344                 break;
345             case 4:
346                 return;
347         }
```

Luego tenemos a un switch donde según la opción seleccionada, se ejecutará el método de la clase.

```
348
349     int repeTempo = 0;
350     cout << endl;
351     do {
352         if (repeTempo > 0)
353             system("cls");
354
355         repeTempo += 1;
356
357         cout << "Desea realizar otra accion?\n1. Si\n2. No\nDigite la respuesta: ";
358         cin >> opcionRepetir;
359     } while (opcionRepetir != 1 and opcionRepetir != 2);
360     if (opcionRepetir == 1)
361         decisionRepetir = true;
362     else if (opcionRepetir == 2)
363         decisionRepetir = false;
364
365 } while (decisionRepetir == true);
366
367 }
```

Por último la parte del código donde decidirá según la opción de usuario (s,n) de volver a ejecutar el menú del cajero.

Class Peliculas

```
36 class Peliculas {
37     private:
38         string m_director;
39         string m_titulo;
40         string m_anio;
41         string m_genero;
42         string m_hora;
43         string m_clasificacion;
44     public:
45         Peliculas(string titulo, string director, string anio, string genero, string clasificacion, string hora)
46             : m_titulo(titulo), m_director(director), m_anio(anio), m_genero(genero), m_clasificacion(clasificacion), m_hora(hora) {}
47
48         void peliculasDisponibles() {
49             cout << "Nombre de la pelicula: " << m_titulo << endl;
50             cout << "Genero: " << m_genero << endl;
51             cout << "Clasificacion " << m_clasificacion << endl;
52             cout << "Director: " << m_director << endl;
53             cout << "Fecha de estreno: " << m_anio << endl;
54             cout << "Hora de la funcion: " << m_hora;
55         }
56     };
}
```

Esta clase tiene como atributos:

- m_director (nombre del director)
- m_titulo (nombre de la película)
- m_anio (fecha de lanzamiento)
- m_genero (genero de la película)
- m_hora (horas disponibles en el cine)
- m_clasificacion (edades aptas para la película)

Después se encuentra su constructor.

Por último, existe un método donde enseña todos los datos anteriormente mencionados.

Función void peliculasTaquilla();

```
368
369     void peliculasTaquilla() {
370         Peliculas pelicula1("Kung Fu Panda 4", "Jack Black", "2024", "Animacion", "Todo publico", "16:00");
371         Peliculas pelicula2("Atrapados en lo profundo", "Claudio Fäh", "2024", "Accion/Aventura", "B14", "17:00");
372         Peliculas pelicula3("Imaginario: Juguete diabólico", "Jeff Wadlow", "2024", "Terror", "B15", "19:00");
373         Peliculas pelicula4("Dune: Part Two", "Denis Villeneuve", "2024", "Ciencia Ficcion", "B12", "21:00");
374
375         cout << "\tBienvenido a las peliculas en taquilla de Cinepolis\n\n";
376         cout << "Peliculas disponibles: \n\n";
377
378         pelicula1.peliculasDisponibles();
379         cout << "\n-----\n";
380         pelicula2.peliculasDisponibles();
381         cout << "\n-----\n";
382         pelicula3.peliculasDisponibles();
383         cout << "\n-----\n";
384         pelicula4.peliculasDisponibles();
385         cout << "\n-----\n";
386         system("pause");
387     }
388 }
```

Primero se crean los objetos correspondientes, enfocados a las películas,

Luego se imprime en pantalla las distintas opciones disponibles en la taquilla de Cinepolis.

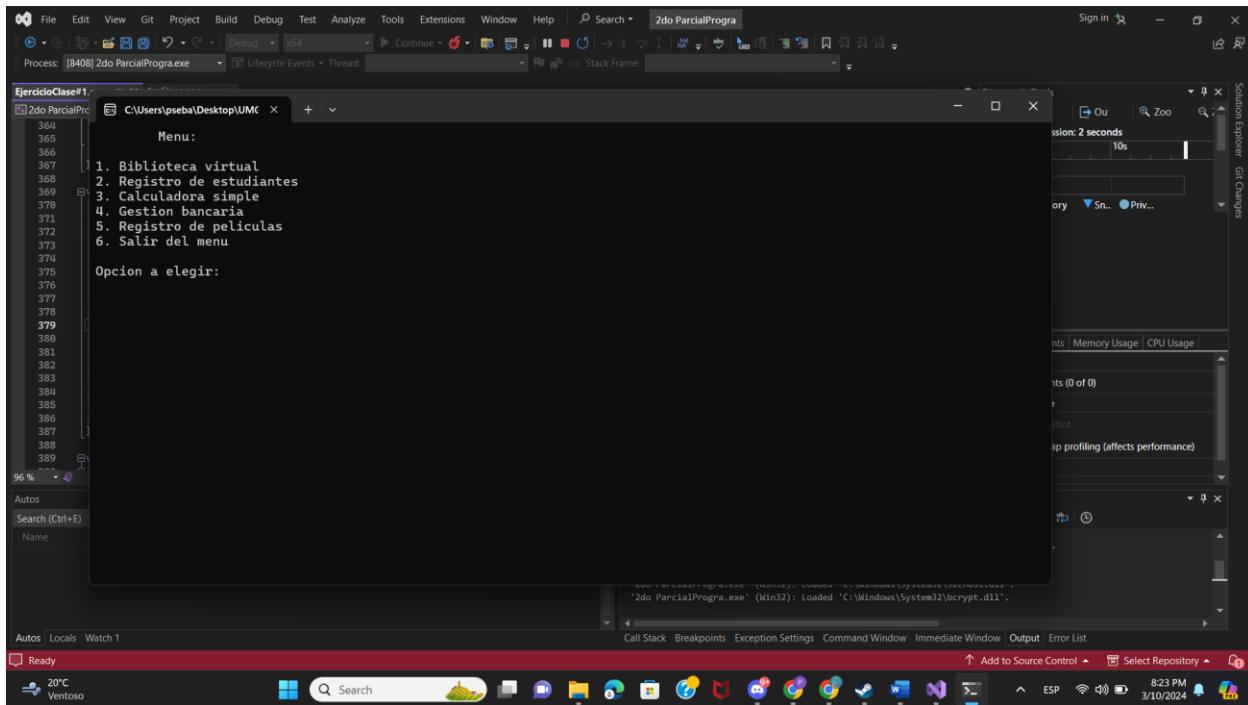
Función void repeticion();

```
388
389     void repeticion() {
390         do {
391             system("cls");
392             cout << "Desea regresar al menu? S/N: ";
393             cin >> respuesta;
394         } while (respuesta != 'n' and respuesta != 'N' and respuesta != 'S' and respuesta != 's');
395
396         if (respuesta == 's' or respuesta == 'S') {
397             repetir = true;
398         }
399         else if (respuesta == 'n' or respuesta == 'N') {
400             repetir = false;
401         }
402     }
403 }
```

Esta función tiene como propósito verificar si el usuario desea seguir dentro del menú y elegir otra opción de las que están disponibles, o sencillamente decide que no y se termine la ejecución del programa. Además, esta función se aplica cada vez que se termina de ejecutar cualquiera de las opciones disponibles del menú.

Ejecución

Menú principal



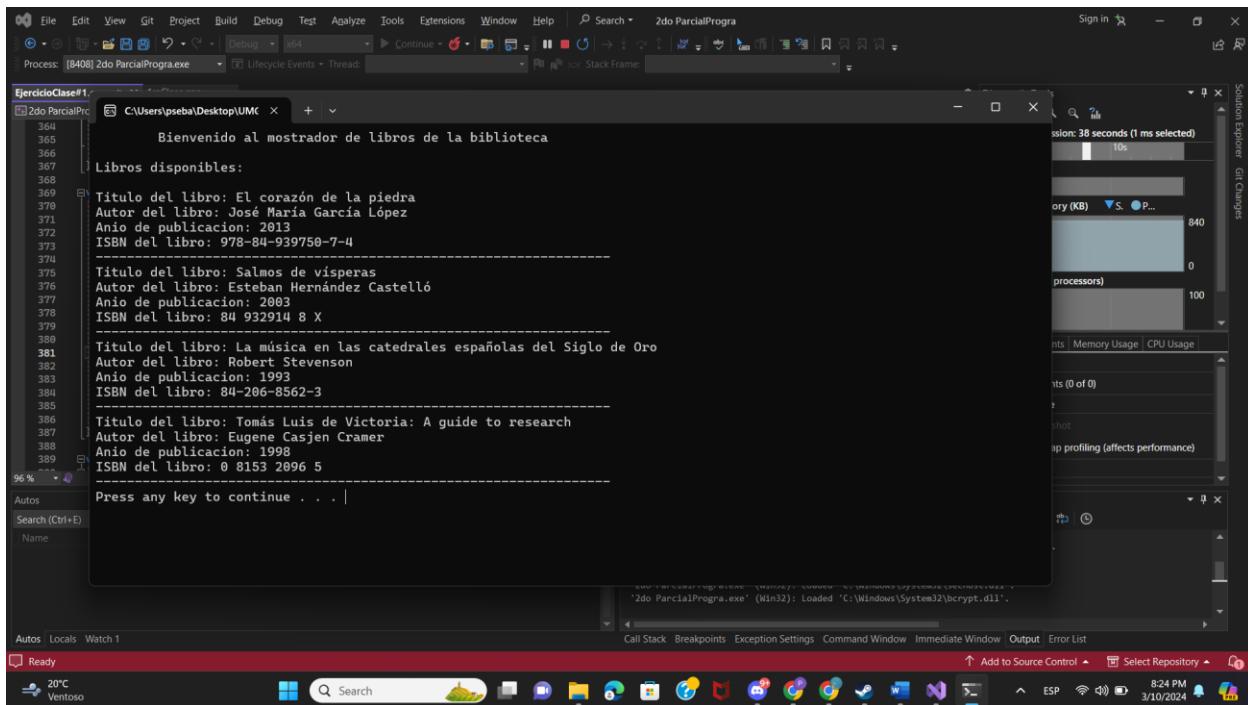
The screenshot shows the Visual Studio IDE interface during the execution of the program. The code editor displays the following menu options:

```
Menu:
1. Biblioteca virtual
2. Registro de estudiantes
3. Calculadora simple
4. Gestión bancaria
5. Registro de películas
6. Salir del menú

Opción a elegir:
```

The status bar at the bottom indicates the process ID is 8408 and the file is "2do ParcialProgra.exe". The taskbar shows various open applications including Microsoft Edge, File Explorer, and the Visual Studio icon.

Biblioteca virtual



The screenshot shows the Visual Studio IDE interface during the execution of the program. The code editor displays the following output from the "Biblioteca virtual" menu option:

```
Bienvenido al mostrador de libros de la biblioteca
Libros disponibles:
Título del libro: El corazón de la piedra
Autor del libro: José María García López
Año de publicación: 2013
ISBN del libro: 978-84-939750-7-4
-----
Título del libro: Salmos de vísperas
Autor del libro: Esteban Hernández Castelló
Año de publicación: 2003
ISBN del libro: 84 932914 8 X
-----
Título del libro: La música en las catedrales españolas del Siglo de Oro
Autor del libro: Robert Stevenson
Año de publicación: 1993
ISBN del libro: 84-206-8562-3
-----
Título del libro: Tomás Luis de Victoria: A guide to research
Autor del libro: Eugene Casjen Cramer
Año de publicación: 1998
ISBN del libro: 0 8153 2096 5

Press any key to continue . . . |
```

The status bar at the bottom indicates the process ID is 8408 and the file is "2do ParcialProgra.exe". The taskbar shows various open applications including Microsoft Edge, File Explorer, and the Visual Studio icon.

Registro de estudiantes

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The status bar at the bottom indicates the process is 0x0408 2do ParcialProgra.exe, the architecture is x64, and the current file is EjercicioClase1.cs.

The main window displays the following text from the application's output:

```
Bienvenido al registro de estudiantes de la UMG
Estudiantes disponibles:
    Nombres: Pablo Sebastián
    Apellidos: Quan Montenegro
    Edad: 19
    Carrera: Ingeniería en Sistemas
    Salón: B-181
    Promedio: 90.6
    -----
    Nombres: Jorge Mario
    Apellidos: Folgar Martínez
    Edad: 18
    Carrera: Arquitectura
    Salón: B-185
    Promedio: 90.5
    -----
    Nombres: Luis Eduardo
    Apellidos: Guevara Juárez
    Edad: 18
    Carrera: Arte
    Salón: A-282
    Promedio: 90.4
    -----
    Nombres: Esteban Mauricio
    Apellidos: Contreras Molina
    Edad: 20
    Carrera: Derecho
    Salón: C-381
    Promedio: 90
Press any key to continue . . .
```

The bottom status bar shows the system is Ready, the temperature is 20°C, and the date is 3/10/2024.

Menú de la calculadora

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The status bar at the bottom indicates the process is 0x0408 2do ParcialProgra.exe, the architecture is x64, and the current file is EjercicioClase1.cs.

The main window displays the following text from the application's output:

```
Calculadora Basica
1. Suma
2. Resta
3. Multiplicacion
4. Division
Elige la operacion que deseas realizar: |
```

The bottom status bar shows the system is Ready, the temperature is 20°C, and the date is 3/10/2024.

En el caso de la suma

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The process being debugged is "2do ParcialProgra.exe". The code editor window displays the following text:

```
364 Elige el numero 1: 56
365 Elige el numero 2: 20
366 El resultado de la suma es: 76
367
368 Desea realizar otra operacion de la calculadora? S/N: |
```

The status bar at the bottom shows "Ready", the date "3/10/2024", and the time "8:27 PM".

En el caso de la resta

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The process being debugged is "2do ParcialProgra.exe". The code editor window displays the following text:

```
364 Elige el numero 1: 56
365 Elige el numero 2: 10
366 El resultado de la resta es: 46
367
368 Desea realizar otra operacion de la calculadora? S/N: |
```

The status bar at the bottom shows "Ready", the date "3/10/2024", and the time "8:28 PM".

En el caso de la multiplicación

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The toolbar below has icons for file operations like Open, Save, and Print. The main window displays a code editor with a file named 'EjercicioClase1.cs'. The code contains the following logic:

```
364 Elige el numero 1: 48
365 Elige el numero 2: 2
366 El resultado de la multiplicacion es: 88
367
368 Desea realizar otra operacion de la calculadora? S/N: q
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
```

The status bar at the bottom shows the path 'C:\Users\pseba\Desktop\UMC' and the message 'The thread 0x584 has exited with code 0 (0x0)'.

En el caso de la división

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The toolbar below has icons for file operations like Open, Save, and Print. The main window displays a code editor with a file named 'EjercicioClase1.cs'. The code contains the following logic:

```
364 Elige el numero 1: 50
365 Elige el numero 2: 4
366 El resultado de la division es: 12.5
367
368 Desea realizar otra operacion de la calculadora? S/N: |
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
```

The status bar at the bottom shows the path 'C:\Users\pseba\Desktop\UMC' and the message 'The thread 0x584 has exited with code 0 (0x0)'.

Menú del cajero

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays a code editor for a file named 'EjercicioClase1.cs'. The code is as follows:

```
364 Bienvenido al menu del cajero
365
366 1. Consultar estado de cuenta
367 2. Retirar dinero
368 3. Depositar dinero
369 4. Salir
370
371 Digite la opcion a elegir: |
```

The code is in C# and presents a menu to the user. The current line of code is line 371, which prompts the user to enter their choice. The status bar at the bottom of the IDE shows the message 'The thread 0x584 has exited with code 0 (0x0)'.

Consulta al estado de cuenta

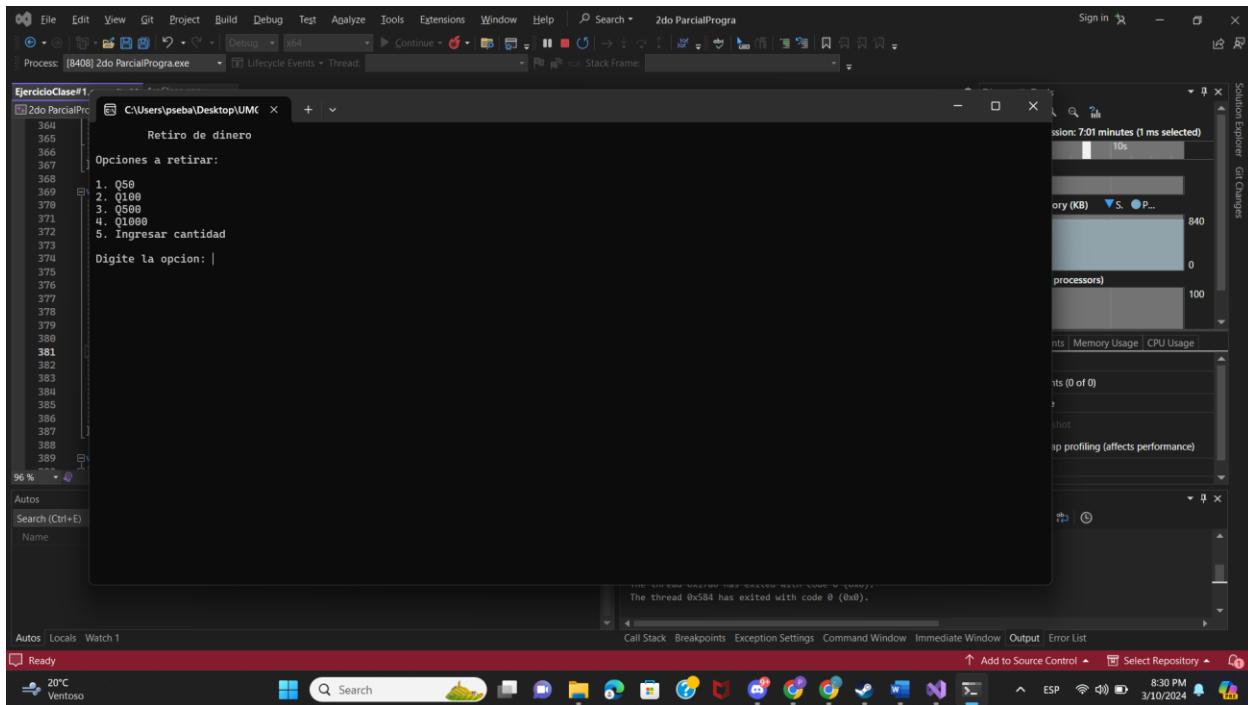
The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays a code editor for a file named 'EjercicioClase1.cs'. The code is as follows:

```
364 Resumen estado de cuenta
365
366 Nombre del usuario: Pablo Sebastian Quan Montenegro
367 Fondos de la cuenta: Q500
368
369 Desea realizar otra accion?
370 1. Si
371 2. No
372 Digite la respuesta: |
```

The code is in C# and displays a summary of the user's account status. It then asks if the user wants to perform another action, offering two options: 'Si' (Yes) or 'No'. The current line of code is line 372, which prompts the user for their response. The status bar at the bottom of the IDE shows the message 'The thread 0x584 has exited with code 0 (0x0)'.

Cuando se elige retirar dinero

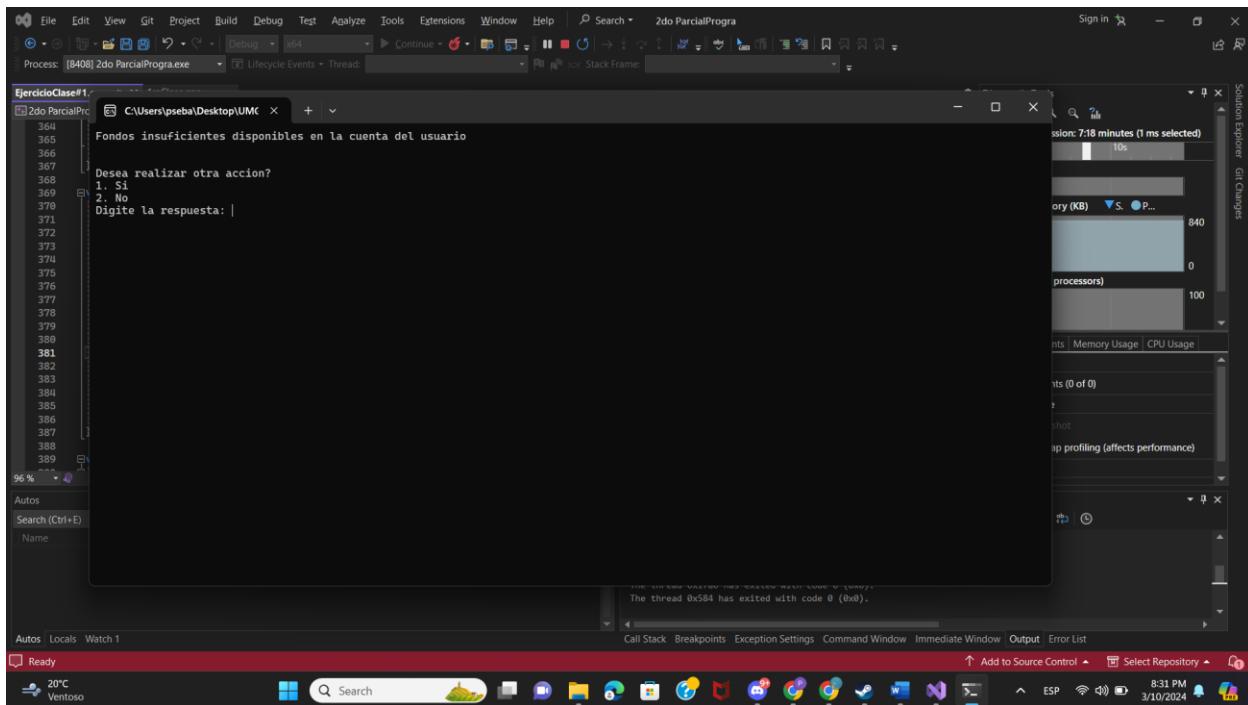
Menú del retiro de dinero



```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search > 2do ParcialProgra
Process: [8408] 2do ParcialProgra.exe > Lifecycle Events > Thread: Stack Frame: Sign in
EjercicioClase1.cs  C:\Users\pseba\Desktop\UMC > + >
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
Retiro de dinero
Opciones a retirar:
1. Q50
2. Q100
3. Q500
4. Q1000
5. Ingresar cantidad
Digite la opcion: |
```

The status bar at the bottom of the IDE shows: "The thread 0x584 has exited with code 0 (0x0)."

Cuando se ingresa una cantidad mayor al saldo de la cuenta disponible



```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search > 2do ParcialProgra
Process: [8408] 2do ParcialProgra.exe > Lifecycle Events > Thread: Stack Frame: Sign in
EjercicioClase1.cs  C:\Users\pseba\Desktop\UMC > + >
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
Fondos insuficientes disponibles en la cuenta del usuario
Desea realizar otra accion?
1. Si
2. No
Digite la respuesta: |
```

The status bar at the bottom of the IDE shows: "The thread 0x584 has exited with code 0 (0x0)."

Cuando la cantidad no excede al saldo de la cuenta (seleccionando la opción de Q50)

```
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
```

Retiro de dinero

Opciones a retirar:

1. Q50
2. Q100
3. Q500
4. Q1000
5. Ingresar cantidad

Digite la opcion: 1

Se ha retirado de manera correcta

Desea realizar otra accion?

1. Si
2. No

Digite la respuesta: |

The thread 0x584 has exited with code 0 (0x0).

Al retirar X cantidad de dinero, esta se va restando en la cuenta, aquí se comprueba: (anteriormente se tenia Q500, ahora Q450 ya que se eligió retirar Q50)

```
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
```

Resumen estado de cuenta

Nombre del usuario: Pablo Sebastian Quan Montenegro

Fondos de la cuenta: Q450

Desea realizar otra accion?

1. Si
2. No

Digite la respuesta: |

The thread 0x584 has exited with code 0 (0x0).

Depositando dinero

The screenshot shows a terminal window in Visual Studio Code with the following output:

```
Deposito de dinero
Ingrese la cantidad a depositar en su cuenta: 5000
El deposito se ha realizado de manera correcta
Desea realizar otra accion?
1. Si
2. No
Digite la respuesta: |
```

The terminal window is titled "EjercicioClase#1" and has a file path "C:\Users\pseba\Desktop\UMC". The status bar at the bottom indicates "Process: [8408] 2do ParcialProgra.exe". The right side of the screen shows the Windows Task Manager and the system tray.

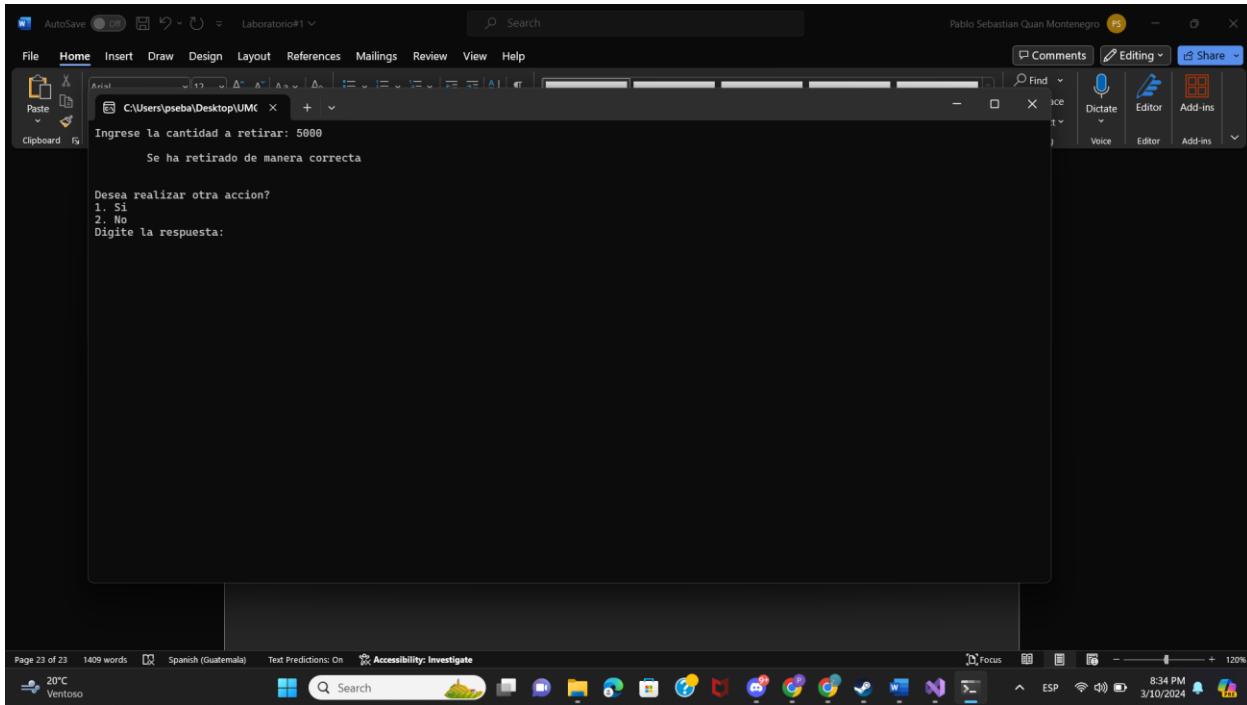
También se puede comprobar que la cantidad depositada (en este caso Q5000) se le suma al saldo de la cuenta

The screenshot shows a terminal window in Visual Studio Code with the following output:

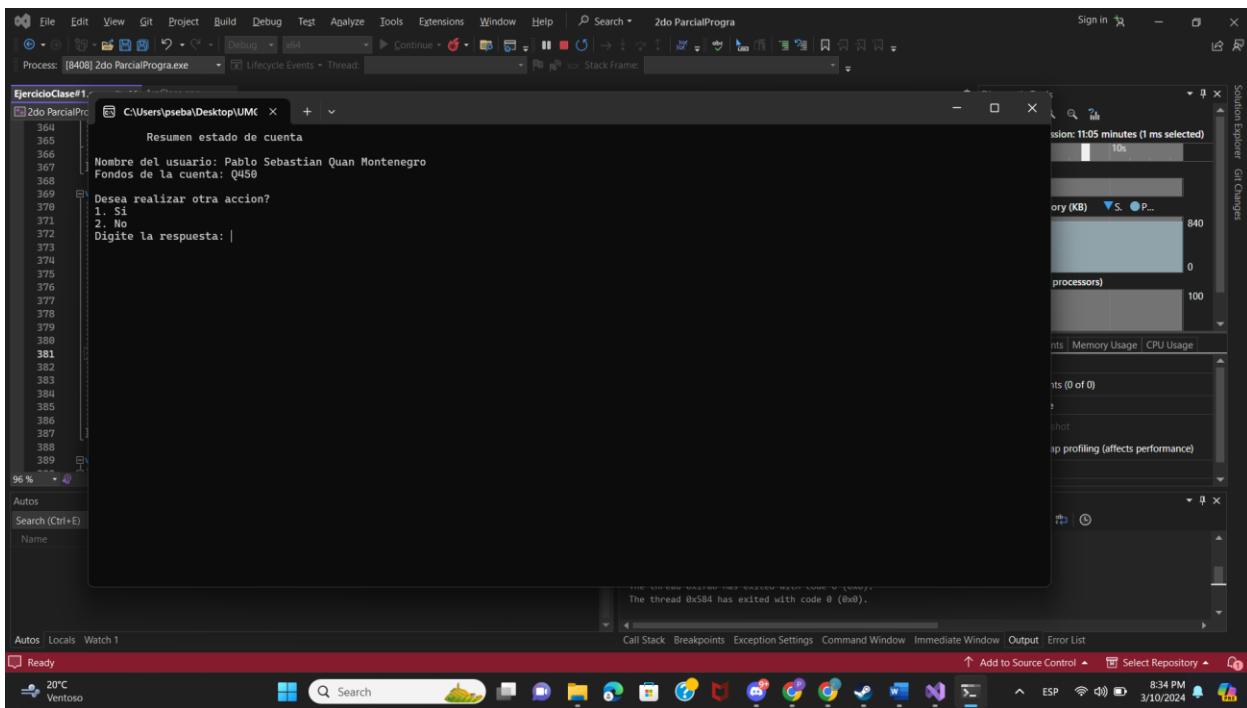
```
Resumen estado de cuenta
Nombre del usuario: Pablo Sebastian Quan Montenegro
Fondos de la cuenta: Q5458
Desea realizar otra accion?
1. Si
2. No
Digite la respuesta: 1|
```

The terminal window is titled "EjercicioClase#1" and has a file path "C:\Users\pseba\Desktop\UMC". The status bar at the bottom indicates "Process: [8408] 2do ParcialProgra.exe". The right side of the screen shows the Windows Task Manager and the system tray.

Cuando se ingresa una cantidad a retirar (Q5000)



Saldo de la cuenta regresa a su anterior estado



Taquilla en cinepolis

```
Bienvenido a las películas en taquilla de Cinepolis
Películas disponibles:
Nombre de la película: Kung Fu Panda 4
Genero: Animacion
Clasificación Todo publico
Director: Jack Black
Fecha de estreno: 2024
Hora de la función: 16:00

Nombre de la película: Atrapados en lo profundo
Genero: Accion/Aventura
Clasificación B14
Director: Claudio Fäh
Fecha de estreno: 2024
Hora de la función: 17:00

Nombre de la película: Imaginario: Juguete diabólico
Genero: Terror
Clasificación B15
Director: Jeff Wadlow
Fecha de estreno: 2024
Hora de la función: 19:00

Nombre de la película: Dune: Part Two
Genero: Ciencia Ficción
Clasificación B12
Director: Denis Villeneuve
Fecha de estreno: 2024
Hora de la función: 21:00

Press any key to continue . . .
```

Opción para regresar al menú principal

```
Desea regresar al menu? S/N: |
```

Conclusión

Se ha presentado un conjunto diverso de funcionalidades integradas en un programa en C++: una biblioteca virtual, un sistema de gestión bancaria, registros de estudiantes, una calculadora simple y una taquilla de películas. Cada módulo ha sido diseñado para proporcionar utilidades específicas y se ha enfocado en facilitar la interacción del usuario con diversas opciones.

La biblioteca virtual permite explorar una selección de libros, ofreciendo información. El registro de estudiantes presenta detalles de varios estudiantes, incluyendo sus nombres, edades, carreras y promedios. La calculadora brinda la posibilidad de realizar operaciones matemáticas básicas, y la taquilla de películas proporciona información sobre películas disponibles, incluyendo género, clasificación, director y horario.

El módulo de gestión bancaria permite a los usuarios consultar el estado de su cuenta, realizar retiros y depósitos. Este módulo destaca la importancia de una interfaz simple y funcional en un entorno bancario.

Además, se llegó a un mejor entendimiento sobre la funcionalidad de las clases. La abstracción fue un paso importante para poder realizar todos los programas solicitados.

Link Github

<https://github.com/pabloquan/Programaci-n-I-Laboratorio-6>