



Universidad Mariano Gálvez de Guatemala

Facultad de Ingeniería en Sistemas

Programación I

Ing. César Alejandro Juárez López

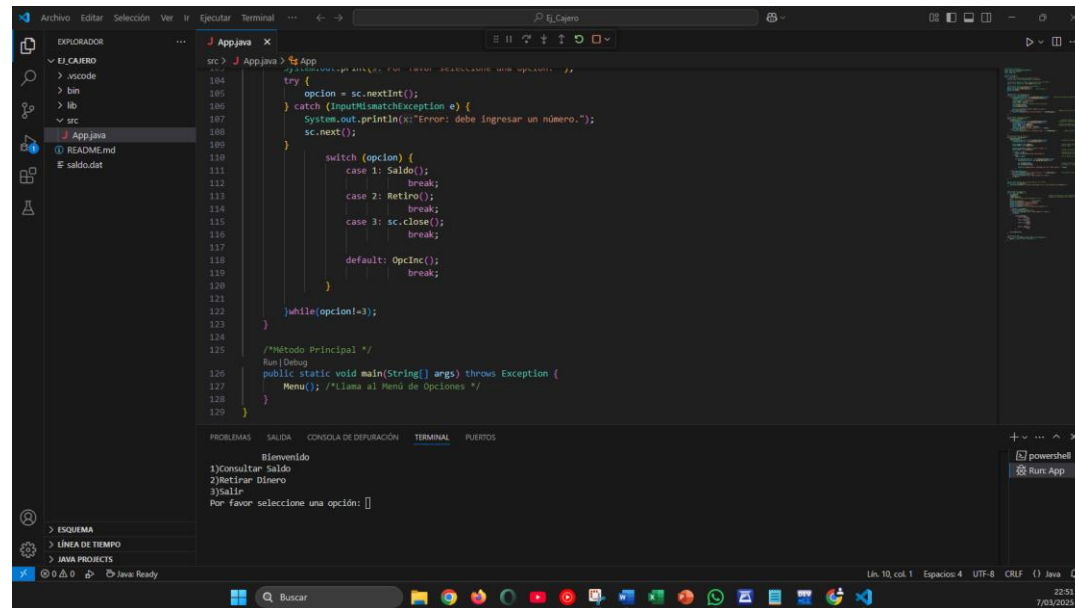
Tarea Semana 5 Archivos Binarios

Luis Armando Reyes Argueta

Carné 0900-24-5602

Guatemala, 08 de marzo de 2025

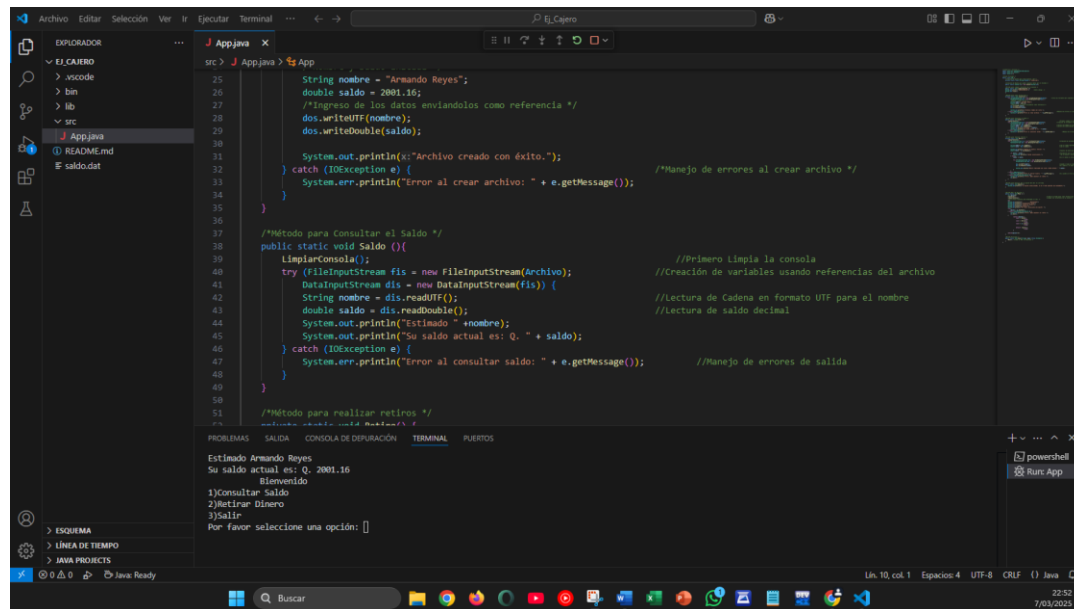
1) Menú Principal



```
src> J App.java > App
//Método para crear un archivo con opciones
104
105 try {
106     opcion = sc.nextInt();
107 } catch (InputMismatchException e) {
108     System.out.println("Error: debe ingresar un número.");
109     sc.next();
110 }
111 switch (opcion) {
112     case 1: saldo();
113         break;
114     case 2: Retiro();
115         break;
116     case 3: sc.close();
117         break;
118     default: OpInc();
119         break;
120 }
121 }while(opcion!=3);
122 }
123
124 //Método Principal */
125 public static void main(String[] args) throws Exception {
126     Menu(); /*Llama al Menú de Opciones */
127 }
128
129 }

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Bienvenido
1)Consultar Saldo
2)Retirar Dinero
3)Salir
Por favor seleccione una opción: []
```

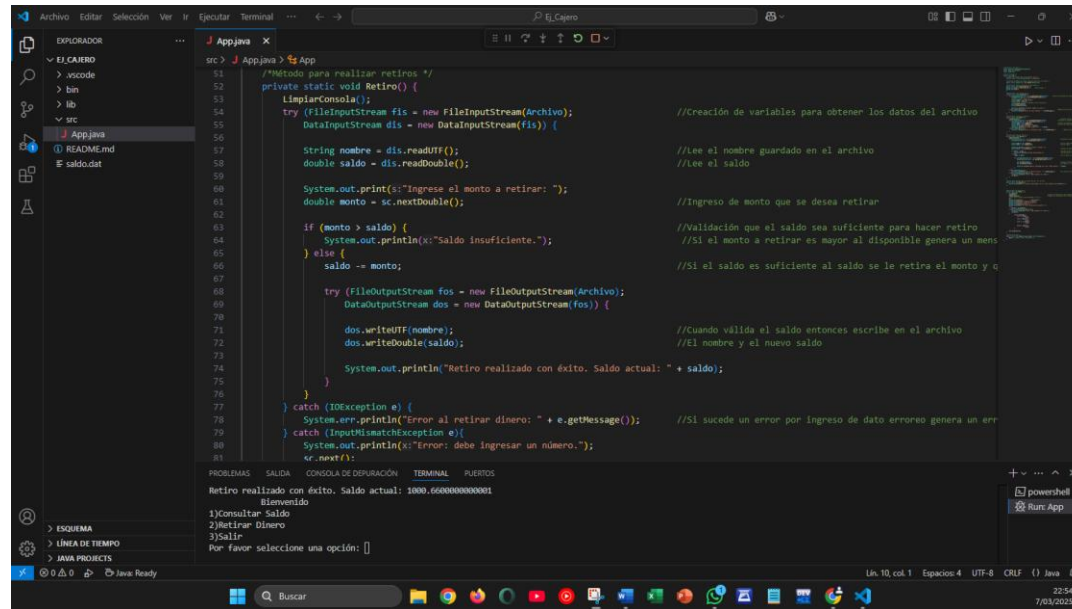
1) Consultar Saldo



```
src> J App.java > App
25 String nombre = "Armando Reyes";
26 double saldo = 2001.16;
27 //Ingreso de los datos enviandolos como referencia */
28 dos.writeUTF(nombre);
29 dos.writeDouble(saldo);
30
31 System.out.println("Archivo creado con éxito.");
32 } catch (IOException e) {
33     System.err.println("Error al crear archivo: " + e.getMessage());
34 }
35 }
36
37 //Método para Consultar el Saldo */
38 public static void Saldo () {
39     limpiarConsola();
40     try (FileInputStream fis = new FileInputStream(Archivo);
41         DataInputStream dis = new DataInputStream(fis)) {
42         String nombre = dis.readUTF();
43         double saldo = dis.readDouble();
44         System.out.println("Estimado " + nombre);
45         System.out.println("Su saldo actual es: Q. " + saldo);
46     } catch (IOException e) {
47         System.err.println("Error al consultar saldo: " + e.getMessage());
48     }
49 }
50
51 //Método para realizar retiros */
52 public static void Retiro() {
53     limpiarConsola();
54 }
55 }

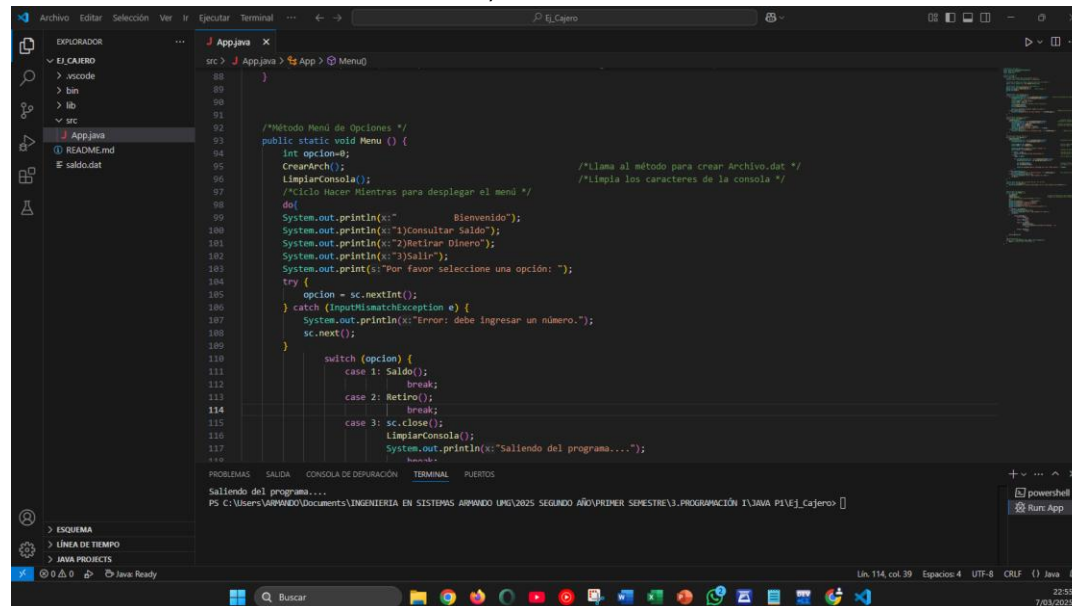
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Estimado Armando Reyes
Su saldo actual es: Q. 2001.16
Bienvenido
1)Consultar Saldo
2)Retirar Dinero
3)Salir
Por favor seleccione una opción: []
```

2) Retirar Efectivo



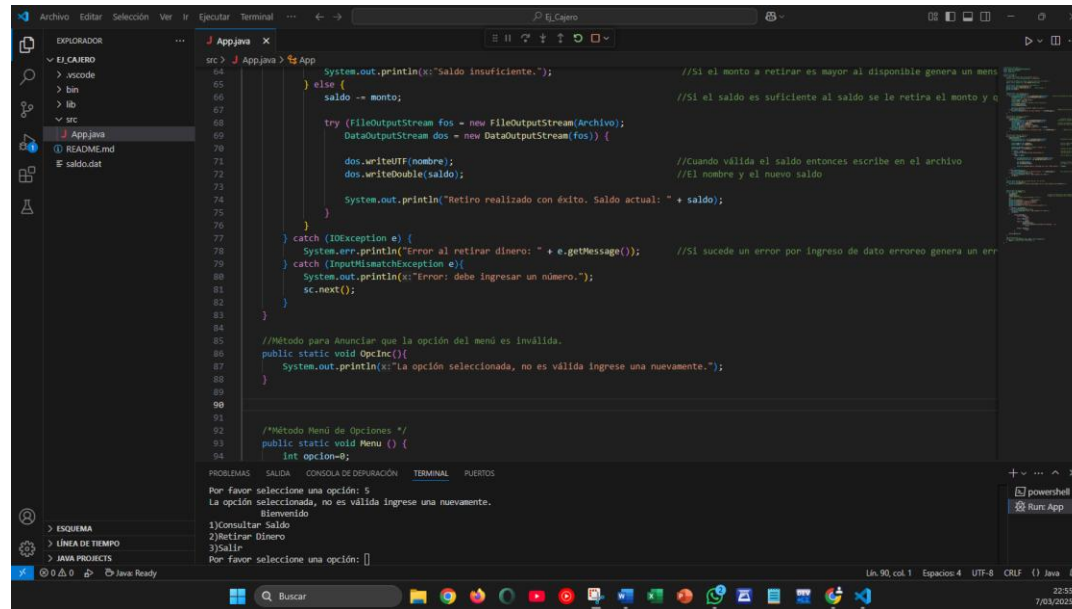
```
src > J App.java > App
51 //Método para realizar retiros */
52 private static void Retiro() {
53     LimpiarConsole();
54     try (FileInputStream fis = new FileInputStream(archivo);
55         DataInputStream dis = new DataInputStream(fis)) {
56         //Creación de variables para obtener los datos del archivo
57         String nombre = dis.readUTF();
58         double saldo = dis.readDouble();
59         //Lee el nombre guardado en el archivo
60         //Lee el saldo
61         System.out.print(s:"Ingrese el monto a retirar: ");
62         double monto = sc.nextDouble();
63         //Ingreso de monto que se desea retirar
64         if (monto > saldo) {
65             System.out.println(s:"Saldo insuficiente.");
66             //Validación que el saldo sea suficiente para hacer retiro
67             //Si el monto a retirar es mayor al disponible genera un mens
68         } else {
69             saldo -= monto;
70             //Si el saldo es suficiente al saldo se le retira el monto y q
71             try (FileOutputStream fos = new FileOutputStream(archivo);
72                 DataOutputStream dos = new DataOutputStream(fos)) {
73                 dos.writeUTF(nombre);
74                 dos.writeDouble(saldo);
75                 //Cuando válida el saldo entonces escribe en el archivo
76                 //El nombre y el nuevo saldo
77                 System.out.println("Retiro realizado con éxito. Saldo actual: " + saldo);
78             }
79         } catch (IOException e) {
80             System.err.println("Error al retirar dinero: " + e.getMessage());
81             //Si sucede un error por ingreso de dato erroneo genera un err
82         } catch (InputMismatchException e) {
83             System.out.println(s:"Error: debe ingresar un número.");
84             sc.next();
85         }
86     }
87 }
88
89 PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Retiro realizado con éxito. Saldo actual: 1000.60000000000001
Bienvenido
1) Consultar Saldo
2) Retirar Dinero
3) Salir
Por favor seleccione una opción: []
```

3) Salir



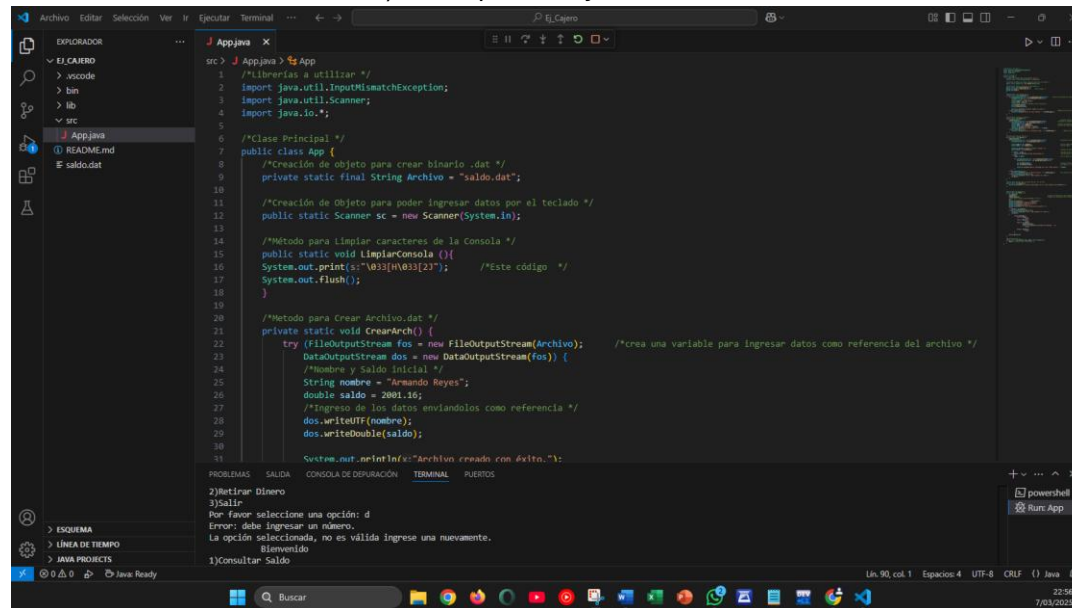
```
src > J App.java > App > Menu()
88 }
89
90 //Método Menu de Opciones */
91 public static void Menu () {
92     int opcion=0;
93     CrearArch();
94     LimpiarConsole();
95     //Llama al método para crear Archivo.dat */
96     //Limpia los caracteres de la consola */
97     //Ciclo Hacer Mientras para desplegar el menú */
98     do {
99         System.out.println(s:" Bienvenido");
100         System.out.println(s:"1) Consultar Saldo");
101         System.out.println(s:"2) Retirar Dinero");
102         System.out.println(s:"3) Salir");
103         System.out.print(s:"Por favor seleccione una opción: ");
104         try {
105             opcion = sc.nextInt();
106         } catch (InputMismatchException e) {
107             System.out.println(s:"Error: debe ingresar un número.");
108             sc.next();
109         }
110         switch (opcion) {
111             case 1: Saldo();
112             case 2: Retiro();
113             case 3: sc.close();
114             LimpiarConsole();
115             System.out.println(s:"Saliedo del programa....");
116         }
117     } while (opcion != 3);
118 }
119
120 PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
Saliedo del programa....
PS C:\Users\YANWANG\Documents\INGENIERIA EN SISTEMAS AVANZADO UNO\2025 SEGUNDO AÑO\PRIMER SEMESTRE\3.PROGRAMACIÓN I\3JAVA P1\Ej_Cajero>
```

4) Opción Incorrecta



```
64      System.out.println(x:"Saldo insuficiente."); //Si el monto a retirar es mayor al disponible genera un mensaje
65  } else {
66      saldo --> monto; //Si el saldo es suficiente al saldo se le retira el monto y se actualiza
67  }
68  try (FileOutputStream fos = new FileOutputStream(Archivo);
69      DataOutputStream dos = new DataOutputStream(fos)) {
70      dos.writeUTF(nombre); //Cuando válida el saldo entonces escribe en el archivo
71      dos.writeDouble(saldo); //El nombre y el nuevo saldo
72  }
73  System.out.println("Retiro realizado con éxito. Saldo actual: " + saldo);
74  }
75  } catch (IOException e) {
76      System.err.println("Error al retirar dinero: " + e.getMessage()); //Si sucede un error por ingreso de dato erroneo genera un error
77  } catch (InputMismatchException e){
78      System.out.println(x:"Error: debe ingresar un número.");
79      sc.next();
80  }
81  }
82  }
83  }
84  //Método para Anunciar que la opción del menú es inválida.
85  public static void OpInco(){
86      System.out.println(x:"La opción seleccionada, no es válida ingrese una nuevamente.");
87  }
88  }
89  }
90  //Método Menú de Opciones */
91  public static void Menu () {
92      int opcion=0;
93  }
94  }
95  }
96  }
97  }
98  }
99  }
100  }
101  }
102  }
103  }
104  }
105  }
106  }
107  }
108  }
109  }
110  }
111  }
112  }
113  }
114  }
115  }
116  }
117  }
118  }
119  }
120  }
121  }
122  }
123  }
124  }
125  }
126  }
127  }
128  }
129  }
130  }
131  }
132  }
133  }
134  }
135  }
136  }
137  }
138  }
139  }
140  }
141  }
142  }
143  }
144  }
145  }
146  }
147  }
148  }
149  }
150  }
151  }
152  }
153  }
154  }
155  }
156  }
157  }
158  }
159  }
160  }
161  }
162  }
163  }
164  }
165  }
166  }
167  }
168  }
169  }
170  }
171  }
172  }
173  }
174  }
175  }
176  }
177  }
178  }
179  }
180  }
181  }
182  }
183  }
184  }
185  }
186  }
187  }
188  }
189  }
190  }
191  }
192  }
193  }
194  }
195  }
196  }
197  }
198  }
199  }
200  }
201  }
202  }
203  }
204  }
205  }
206  }
207  }
208  }
209  }
210  }
211  }
212  }
213  }
214  }
215  }
216  }
217  }
218  }
219  }
220  }
221  }
222  }
223  }
224  }
225  }
226  }
227  }
228  }
229  }
230  }
231  }
232  }
233  }
234  }
235  }
236  }
237  }
238  }
239  }
240  }
241  }
242  }
243  }
244  }
245  }
246  }
247  }
248  }
249  }
250  }
251  }
252  }
253  }
254  }
255  }
256  }
257  }
258  }
259  }
260  }
261  }
262  }
263  }
264  }
265  }
266  }
267  }
268  }
269  }
270  }
271  }
272  }
273  }
274  }
275  }
276  }
277  }
278  }
279  }
280  }
281  }
282  }
283  }
284  }
285  }
286  }
287  }
288  }
289  }
290  }
291  }
292  }
293  }
294  }
295  }
296  }
297  }
298  }
299  }
300  }
301  }
302  }
303  }
304  }
305  }
306  }
307  }
308  }
309  }
310  }
311  }
312  }
313  }
314  }
315  }
316  }
317  }
318  }
319  }
320  }
321  }
322  }
323  }
324  }
325  }
326  }
327  }
328  }
329  }
330  }
331  }
332  }
333  }
334  }
335  }
336  }
337  }
338  }
339  }
340  }
341  }
342  }
343  }
344  }
345  }
346  }
347  }
348  }
349  }
350  }
351  }
352  }
353  }
354  }
355  }
356  }
357  }
358  }
359  }
360  }
361  }
362  }
363  }
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  }
390  }
391  }
392  }
393  }
394  }
395  }
396  }
397  }
398  }
399  }
400  }
401  }
402  }
403  }
404  }
405  }
406  }
407  }
408  }
409  }
410  }
411  }
412  }
413  }
414  }
415  }
416  }
417  }
418  }
419  }
420  }
421  }
422  }
423  }
424  }
425  }
426  }
427  }
428  }
429  }
430  }
431  }
432  }
433  }
434  }
435  }
436  }
437  }
438  }
439  }
440  }
441  }
442  }
443  }
444  }
445  }
446  }
447  }
448  }
449  }
450  }
451  }
452  }
453  }
454  }
455  }
456  }
457  }
458  }
459  }
460  }
461  }
462  }
463  }
464  }
465  }
466  }
467  }
468  }
469  }
470  }
471  }
472  }
473  }
474  }
475  }
476  }
477  }
478  }
479  }
480  }
481  }
482  }
483  }
484  }
485  }
486  }
487  }
488  }
489  }
490  }
491  }
492  }
493  }
494  }
495  }
496  }
497  }
498  }
499  }
500  }
501  }
502  }
503  }
504  }
505  }
506  }
507  }
508  }
509  }
510  }
511  }
512  }
513  }
514  }
515  }
516  }
517  }
518  }
519  }
520  }
521  }
522  }
523  }
524  }
525  }
526  }
527  }
528  }
529  }
530  }
531  }
532  }
533  }
534  }
535  }
536  }
537  }
538  }
539  }
540  }
541  }
542  }
543  }
544  }
545  }
546  }
547  }
548  }
549  }
550  }
551  }
552  }
553  }
554  }
555  }
556  }
557  }
558  }
559  }
560  }
561  }
562  }
563  }
564  }
565  }
566  }
567  }
568  }
569  }
570  }
571  }
572  }
573  }
574  }
575  }
576  }
577  }
578  }
579  }
580  }
581  }
582  }
583  }
584  }
585  }
586  }
587  }
588  }
589  }
590  }
591  }
592  }
593  }
594  }
595  }
596  }
597  }
598  }
599  }
600  }
601  }
602  }
603  }
604  }
605  }
606  }
607  }
608  }
609  }
610  }
611  }
612  }
613  }
614  }
615  }
616  }
617  }
618  }
619  }
620  }
621  }
622  }
623  }
624  }
625  }
626  }
627  }
628  }
629  }
630  }
631  }
632  }
633  }
634  }
635  }
636  }
637  }
638  }
639  }
640  }
641  }
642  }
643  }
644  }
645  }
646  }
647  }
648  }
649  }
650  }
651  }
652  }
653  }
654  }
655  }
656  }
657  }
658  }
659  }
660  }
661  }
662  }
663  }
664  }
665  }
666  }
667  }
668  }
669  }
670  }
671  }
672  }
673  }
674  }
675  }
676  }
677  }
678  }
679  }
680  }
681  }
682  }
683  }
684  }
685  }
686  }
687  }
688  }
689  }
690  }
691  }
692  }
693  }
694  }
695  }
696  }
697  }
698  }
699  }
700  }
701  }
702  }
703  }
704  }
705  }
706  }
707  }
708  }
709  }
710  }
711  }
712  }
713  }
714  }
715  }
716  }
717  }
718  }
719  }
720  }
721  }
722  }
723  }
724  }
725  }
726  }
727  }
728  }
729  }
730  }
731  }
732  }
733  }
734  }
735  }
736  }
737  }
738  }
739  }
740  }
741  }
742  }
743  }
744  }
745  }
746  }
747  }
748  }
749  }
750  }
751  }
752  }
753  }
754  }
755  }
756  }
757  }
758  }
759  }
760  }
761  }
762  }
763  }
764  }
765  }
766  }
767  }
768  }
769  }
770  }
771  }
772  }
773  }
774  }
775  }
776  }
777  }
778  }
779  }
780  }
781  }
782  }
783  }
784  }
785  }
786  }
787  }
788  }
789  }
790  }
791  }
792  }
793  }
794  }
795  }
796  }
797  }
798  }
799  }
800  }
801  }
802  }
803  }
804  }
805  }
806  }
807  }
808  }
809  }
810  }
811  }
812  }
813  }
814  }
815  }
816  }
817  }
818  }
819  }
820  }
821  }
822  }
823  }
824  }
825  }
826  }
827  }
828  }
829  }
830  }
831  }
832  }
833  }
834  }
835  }
836  }
837  }
838  }
839  }
840  }
841  }
842  }
843  }
844  }
845  }
846  }
847  }
848  }
849  }
850  }
851  }
852  }
853  }
854  }
855  }
856  }
857  }
858  }
859  }
860  }
861  }
862  }
863  }
864  }
865  }
866  }
867  }
868  }
869  }
870  }
871  }
872  }
873  }
874  }
875  }
876  }
877  }
878  }
879  }
880  }
881  }
882  }
883  }
884  }
885  }
886  }
887  }
888  }
889  }
890  }
891  }
892  }
893  }
894  }
895  }
896  }
897  }
898  }
899  }
900  }
901  }
902  }
903  }
904  }
905  }
906  }
907  }
908  }
909  }
910  }
911  }
912  }
913  }
914  }
915  }
916  }
917  }
918  }
919  }
920  }
921  }
922  }
923  }
924  }
925  }
926  }
927  }
928  }
929  }
930  }
931  }
932  }
933  }
934  }
935  }
936  }
937  }
938  }
939  }
940  }
941  }
942  }
943  }
944  }
945  }
946  }
947  }
948  }
949  }
950  }
951  }
952  }
953  }
954  }
955  }
956  }
957  }
958  }
959  }
960  }
961  }
962  }
963  }
964  }
965  }
966  }
967  }
968  }
969  }
970  }
971  }
972  }
973  }
974  }
975  }
976  }
977  }
978  }
979  }
980  }
981  }
982  }
983  }
984  }
985  }
986  }
987  }
988  }
989  }
990  }
991  }
992  }
993  }
994  }
995  }
996  }
997  }
998  }
999  }
1000  }
```

5) Excepciones y Fallos



```
1  //Librerías a utilizar */
2  import java.util.InputMismatchException;
3  import java.util.Scanner;
4  import java.io.*;
5
6  /*Clase Principal */
7  public class App {
8      /*Creación de objeto para crear binario .dat */
9      private static final String Archivo = "saldo.dat";
10
11      /*Creación de Objeto para poder ingresar datos por el teclado */
12      public static Scanner sc = new Scanner(System.in);
13
14      /*Método para Limpiar caracteres de la Consola */
15      public static void LimpiarConsola () {
16          System.out.print("\033[H\033[2J"); //Este código
17          System.out.flush();
18      }
19
20      /*Metodo para Crear Archivo.dat */
21      private static void CrearArch() {
22          try (FileOutputStream fos = new FileOutputStream(Archivo);
23              DataOutputStream dos = new DataOutputStream(fos)) { //crea una variable para ingresar datos como referencia del archivo
24              /*Nombre y Saldo inicial */
25              String nombre = "Armando Reyes";
26              double saldo = 2001.56;
27              /*Ingreso de los datos enviandolos como referencia */
28              dos.writeUTF(nombre);
29              dos.writeDouble(saldo);
30          }
31          System.out.println("Archivo creado con éxito.");
32      }
33
34      /*Método para Consultar Saldo */
35      public static void ConsultarSaldo() {
36          try (FileInputStream fis = new FileInputStream(Archivo);
37              DataInputStream dis = new DataInputStream(fis)) {
38              String nombre = dis.readUTF();
39              double saldo = dis.readDouble();
40              System.out.println("Nombre: " + nombre + " Saldo: " + saldo);
41          }
42      }
43
44      /*Método para Retirar Dinero */
45      public static void RetirarDinero() {
46          try {
47              double monto = sc.nextDouble();
48              double saldo = 2001.56;
49              saldo -= monto;
50              System.out.println("Saldo actual: " + saldo);
51          } catch (InputMismatchException e) {
52              System.out.println("Error: debe ingresar un número.");
53          }
54      }
55
56      /*Método para Salir */
57      public static void Salir() {
58          System.out.println("¡Adios!");
59      }
60
61      /*Método para Anunciar que la opción del menú es inválida. */
62      public static void OpInco(){
63          System.out.println(x:"La opción seleccionada, no es válida ingrese una nuevamente.");
64      }
65
66      /*Método Menú de Opciones */
67      public static void Menu () {
68          int opcion=0;
69          while (opcion != 0) {
70              System.out.println("Bienvenido");
71              System.out.println("1) Consultar Saldo");
72              System.out.println("2) Retirar Dinero");
73              System.out.println("3) Salir");
74              System.out.println("Por favor seleccione una opción: ");
75              opcion = sc.nextInt();
76              switch (opcion) {
77                  case 1: ConsultarSaldo(); break;
78                  case 2: RetirarDinero(); break;
79                  case 3: Salir(); break;
80                  default: OpInco();
81              }
82          }
83      }
84  }
85  }
86  }
87  }
88  }
89  }
90  }
91  }
92  }
93  }
94  }
95  }
96  }
97  }
98  }
99  }
100  }
101  }
102  }
103  }
104  }
105  }
106  }
107  }
108  }
109  }
110  }
111  }
112  }
113  }
114  }
115  }
116  }
117  }
118  }
119  }
120  }
121  }
122  }
123  }
124  }
125  }
126  }
127  }
128  }
129  }
130  }
131  }
132  }
133  }
134  }
135  }
136  }
137  }
138  }
139  }
140  }
141  }
142  }
143  }
144  }
145  }
146  }
147  }
148  }
149  }
150  }
151  }
152  }
153  }
154  }
155  }
156  }
157  }
158  }
159  }
160  }
161  }
162  }
163  }
164  }
165  }
166  }
167  }
168  }
169  }
170  }
171  }
172  }
173  }
174  }
175  }
176  }
177  }
178  }
179  }
180  }
181  }
182  }
183  }
184  }
185  }
186  }
187  }
188  }
189  }
190  }
191  }
192  }
193  }
194  }
195  }
196  }
197  }
198  }
199  }
200  }
201  }
202  }
203  }
204  }
205  }
206  }
207  }
208  }
209  }
210  }
211  }
212  }
213  }
214  }
215  }
216  }
217  }
218  }
219  }
220  }
221  }
222  }
223  }
224  }
225  }
226  }
227  }
228  }
229  }
230  }
231  }
232  }
233  }
234  }
235  }
236  }
237  }
238  }
239  }
240  }
241  }
242  }
243  }
244  }
245  }
246  }
247  }
248  }
249  }
250  }
251  }
252  }
253  }
254  }
255  }
256  }
257  }
258  }
259  }
260  }
261  }
262  }
263  }
264  }
265  }
266  }
267  }
268  }
269  }
270  }
271  }
272  }
273  }
274  }
275  }
276  }
277  }
278  }
279  }
280  }
281  }
282  }
283  }
284  }
285  }
286  }
287  }
288  }
289  }
290  }
291  }
292  }
293  }
294  }
295  }
296  }
297  }
298  }
299  }
300  }
301  }
302  }
303  }
304  }
305  }
306  }
307  }
308  }
309  }
310  }
311  }
312  }
313  }
314  }
315  }
316  }
317  }
318  }
319  }
320  }
321  }
322  }
323  }
324  }
325  }
326  }
327  }
328  }
329  }
330  }
331  }
332  }
333  }
334  }
335  }
336  }
337  }
338  }
339  }
340  }
341  }
342  }
343  }
344  }
345  }
346  }
347  }
348  }
349  }
350  }
351  }
352  }
353  }
354  }
355  }
356  }
357  }
358  }
359  }
360  }
361  }
362  }
363  }
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  }
390  }
391  }
392  }
393  }
394  }
395  }
396  }
397  }
398  }
399  }
400  }
401  }
402  }
403  }
404  }
405  }
406  }
407  }
408  }
409  }
410  }
411  }
412  }
413  }
414  }
415  }
416  }
417  }
418  }
419  }
420  }
421  }
422  }
423  }
424  }
425  }
426  }
427  }
428  }
429  }
430  }
431  }
432  }
433  }
434  }
435  }
436  }
437  }
438  }
439  }
440  }
441  }
442  }
443  }
444  }
445  }
446  }
447  }
448  }
449  }
450  }
451  }
452  }
453  }
454  }
455  }
456  }
457  }
458  }
459  }
460  }
461  }
462  }
463  }
464  }
465  }
466  }
467  }
468  }
469  }
470  }
471  }
472  }
473  }
474  }
475  }
476  }
477  }
478  }
479  }
480  }
481  }
482  }
483  }
484  }
485  }
486  }
487  }
488  }
489  }
490  }
491  }
492  }
493  }
494  }
495  }
496  }
497  }
498  }
499  }
500  }
501  }
502  }
503  }
504  }
505  }
506  }
507  }
508  }
509  }
510  }
511  }
512  }
513  }
514  }
515  }
516  }
517  }
518  }
519  }
520  }
521  }
522  }
523  }
524  }
525  }
526  }
527  }
528  }
529  }
530  }
531  }
532  }
533  }
534  }
535  }
536  }
537  }
538  }
539  }
540  }
541  }
542  }
543  }
544  }
545  }
546  }
547  }
548  }
549  }
550  }
551  }
552  }
553  }
554  }
555  }
556  }
557  }
558  }
559  }
560  }
561  }
562  }
563  }
564  }
565  }
566  }
567  }
568  }
569  }
570  }
571  }
572  }
573  }
574  }
575  }
576  }
577  }
578  }
579  }
580  }
581  }
582  }
583  }
584  }
585  }
586  }
587  }
588  }
589  }
590  }
591  }
592  }
593  }
594  }
595  }
596  }
597  }
598  }
599  }
600  }
601  }
602  }
603  }
604  }
605  }
606  }
607  }
608  }
609  }
610  }
611  }
612  }
613  }
614  }
615  }
616  }
617  }
618  }
619  }
620  }
621  }
622  }
623  }
624  }
625  }
626  }
627  }
628  }
629  }
630  }
631  }
632  }
633  }
634  }
635  }
636  }
637  }
638  }
639  }
640  }
641  }
642  }
643  }
644  }
645  }
646  }
647  }
648  }
649  }
650  }
651  }
652  }
653  }
654  }
655  }
656  }
657  }
658  }
659  }
660  }
661  }
662  }
663  }
664  }
665  }
666  }
667  }
668  }
669  }
670  }
671  }
672  }
673  }
674  }
675  }
676  }
677  }
678  }
679  }
680  }
681  }
682  }
683  }
684  }
685  }
686  }
687  }
688  }
689  }
690  }
691  }
692  }
693  }
694  }
695  }
696  }
697  }
698  }
699  }
700  }
701  }
702  }
703  }
704  }
705  }
706  }
707  }
708  }
709  }
710  }
711  }
712  }
713  }
714  }
715  }
716  }
717  }
718  }
719  }
720  }
721  }
722  }
723  }
724  }
725  }
726  }
727  }
728  }
729  }
730  }
731  }
732  }
733  }
734  }
735  }
736  }
737  }
738  }
739  }
740  }
741  }
742  }
743  }
744  }
745  }
746  }
747  }
748  }
749  }
750  }
751  }
752  }
753  }
754  }
755  }
756  }
757  }
758  }
759  }
760  }
761  }
762  }
763  }
764  }
765  }
766  }
767  }
768  }
769  }
770  }
771  }
772  }
773  }
774  }
775  }
776  }
777  }
778  }
779  }
780  }
781  }
782  }
783  }
784  }
785  }
786  }
787  }
788  }
789  }
790  }
791  }
792  }
793  }
794  }
795  }
796  }
797  }
798  }
799  }
800  }
801  }
802  }
803  }
804  }
805  }
806  }
807  }
808  }
809  }
810  }
811  }
812  }
813  }
814  }
815  }
816  }
817  }
818  }
819  }
820  }
821  }
822  }
823  }
824  }
825  }
826  }
827  }
828  }
829  }
830  }
831  }
832  }
833  }
834  }
835  }
836  }
837  }
838  }
839  }
840  }
841  }
842  }
843  }
844  }
845  }
846  }
847  }
848  }
849  }
850  }
851  }
852  }
853  }
854  }
855  }
856  }
857  }
858  }
859  }
860  }
861  }
862  }
863  }
864  }
865  }
866  }
867  }
868  }
869  }
870  }
871  }
872  }
873  }
874  }
875  }
876  }
877  }
878  }
879  }
880  }
881  }
882  }
883  }
884  }
885  }
886  }
887  }
888  }
889  }
890  }
891  }
892  }
893  }
894  }
895  }
896  }
897  }
898  }
899  }
900  }
901  }
902  }
903  }
904  }
905  }
906  }
907  }
908  }
909  }
910  }
911  }
912  }
913  }
914  }
915  }
916  }
917  }
918  }
919  }
920  }
921  }
922  }
923  }
924  }
925  }
926  }
927  }
928  }
929  }
930  }
931  }
932  }
933  }
934  }
935  }
936  }
937  }
938  }
939  }
940  }
941  }
942  }
943  }
944  }
945  }
946  }
947  }
948  }
949  }
950  }
951  }
952  }
953  }
954  }
955  }
956  }
957  }
958  }
959  }
960  }
961  }
962  }
963  }
964  }
965  }
966  }
967  }
968  }
969  }
970  }
971  }
972  }
973  }
974  }
975  }
976  }
977  }
978  }
979  }
9
```

