	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): René Adrián Dávila Pérez

Asignatura: Programación Orientada a Objetos

Grupo: 1

No de Práctica(s): PROYECTO 1

Integrante(s): 322184022

322238682

321570789

322069093

322249723

No. de lista o brigada: 5

Semestre: 2026-1

Fecha de entrega: 26/09/2025

Observaciones:

CALIFICACIÓN: _____

Índice

1. Introducción

En este proyecto se trabajó con el lenguaje de programación Java, fueron aplicados los conceptos de clases, objetos, atributos y métodos. Todo esto, reforzando el conocimiento y manejo del lenguaje Java. El proyecto consistió en el desarrollo de un algoritmo que simula un carrito de compras, dicho de otra manera se creó un programa que almacena datos, artículos y precios en este caso. Para esto se empleó ArrayList para crear listas dinámicas, a través de las cuales se almacenaron los artículos y sus respectivos precios. También se manejó la eliminación de los artículos cuando sea necesario. Se trabajó la organización y gestión de datos, representados como productos en este caso. Este proyecto permite ver cómo se aplican los conceptos a un problema cercano a la vida real y cotidiano, como son los carritos de compras o bien podrian ser los carritos virtuales como los hay en Amazon, Mercado Libre, entre otros.

1.1. Objetivos

- Desarrollo de un programa de gestión de artpiculos y precios
- Comprender y aplicar los conceptos como clases, objetos, atributos, métodos y constructores.
- Desarrolla habilidades de gestión de datos simulando un problema real y cotidiano como los carritos de compras.

2. Marco Teórico

- **Clases**: son modelos o abstracciones de la realidad que representan un elemento de un conjunto de *objetos* [1]
- **Objeto:** es la estructura o entidad abstracta que se crea a partir de la definición de una *clase*, de tal manera, que ésta permite organizar y relacionar los *datos* de un programa o problema (*atributos*) con sus respectivos *métodos*. [3]
    - **Atributos**: Conjunto de rasgos o características que posee un objeto [3]
    - **Métodos**: Conjunto de procedimientos o funciones para la manipulación o interacción con los datos de un objeto.[3]
        - **Constructor:** Método que tiene el mismo nombre que la clase y cuyo propósito es inicializar los atributos de un nuevo objeto. Se ejecuta automáticamente cuando se crea un objeto o instancia de la clase.[6]
        - **Método estático:** métodos que no requieren un objeto para ser invocados, se pueden ejecutar a través de la clase. [6]
        - **Método de instancia:** Estos métodos operan en instancias de una clase y pueden acceder a los campos y otros métodos de esa instancia. Son invocados a través de un objeto creado a partir de la clase. [4]

- **Estructuras de control** [7]
  - **for**: Toma como punto de partida un índice de control, al final de cada bucle este índice se actualiza según la regla establecida en la estrucutra del bucle. Finalmente, el bucle comprueba si el índice satisface la condición booleana.
  - **for-each**: Se utiliza para iterar a lo largo de una lista de objetos o variables primitivas.[2]
  - **if-else**: Se evalua una expresión booleana, si se cumple la condición se ejecuta la instrucción principal, por otro lado si la condición es falsa se ejecuta la segunda instrucción.
  - **switch**: Se utiliza para tomas decisiones sobre la misma variable, es decir, se implementan menús de opciones.
- **ArrayList** [5]
  Es una clase que implementa la interfaz List y proporciona métodos para manipular el tamaño de la matriz que se utiliza internamente, permitiendo almacenar elementos de forma dinámica.

  Sintaxis:
  ArrayList<String>nombreArrayList= new ArrayList<String>();
  (puede ser de cualquier otro elemento u objeto).

  Métodos implementados
  - **add**: Anexa el elemento especificado al final de la lista.
  - **remove**: Quita el elemento en la posición especificada en la lista.
  - **get**: Devuelve el elemento en la posición especificada en la lista.

3. Desarrollo

- **Clase Articulo:**
  - **Atributos:**
    - **String articulo:** Nombre del artículo
    - **double precio:** Precio del artículo en un tipo flotante
  - **Métodos:**
    - **public void printArticulo() :** imprime el artículo de la forma: *nombreInstancia.printArticulo()*, que no recibe argumentos, ya que es un método de instancia, que quiere decir que toma los valores de la instancia (objeto del que se llama)[4]
- **Clase Carrito:**
  - **Atributos:**
    - **ArrayList <Articulo>lista = new ArrayList<>();** ArrayList donde se almacenan objetos de tipo **Articulo**, sobre ella se trabajará en los *métodos*
  - **Métodos:**
    - **public void agregarArticulo(Articulo art){...} :** recibe como argumento un objeto de tipo **Articulo**, dentro de la función se llama al método **.add** del ArrayList lista y agrega el **Articulo** que se recibió en los argumentos.
    - **public void eliminarArticulo(int indArticulo){...}:** recibe como argumento *int indArticulo, que es el* ***número de artículo*** *que se desea eliminar*, dentro del método, primero copiamos en un objeto temporal de tipo **Articulo**, unicamente con la finalidad de imprimirlo, después, con el método **.remove** del ArrayList **lista** elimina el **Articulo** mediante su *índice*, que es **int indArticulo-1**, se le debe restar 1 ya que cuando se imprime la lista la enumeración de elementos empieza en 1, pero los índices de **lista** empiezan en 0.
    - **public void printLista(){...}:** dentro de la función, primero con el método **.isEmpty** del ArrayList **lista** verifica primero si está vacía, si lo está, envía el mensaje "No hay artículos", si no está vacía, con una variable de control int numArt=1 (que inicalizamos en 1 ya que si no está vacía, al menos hay un elemento), entra a un **for-each** con un **Articulo** temporal que recorre cada **Articulo** dentro de la **lista**, este método sirve para imprimir cada artículo con el formato: *Artículo numArt :* y llama al método **printArticulo()** del *artículo temporal*, finalmente, incrementa numArt para la siguiente impresión.

- **Clase Proyecto:** Clase que contiene el método **main**
  - Importación de java.util.Scanner necesaria para seleccionar una opción del **Menú**
  - Declaración de nuevo Scanner llamado *sc*
  - Creación de un nuevo **Carrito** llamado **carr**
  - Bienvenida al programa
  - Declaración de la variable de tipo entero **opcion**
  - Dentro de un **do-while**:
    **Uso del do-while para que haga AL MENOS 1 vez la ejecución, y al final verificar si es el caso donde sale de este control*
    - Se imprime el **Menú de opciones**, que son:
      1. Ver lista de compras
      2. Agregar artículo
      3. ELiminar artículo
      4. Salir
    - Mediante *sc* se escanea la opción que elija el usuario y se almacena en la variable **opcion**
    - Haciendo uso de **switch-case**, donde la variable o expresión a evaluar es **opcion**:
      El caso dentro del do-while para que se rompa la condición y salga, es que el valor la variable **opcion** sea igual a 4, mientras sea diferente de 4, seguirá imprimiendo el menú y pidiendo la opción.
      - **case 1:** (Ver lista de compras)
        Como lo que se quiere es imprimir la lista completa, sólo se llama al método **printLista()** de **carr**
      - **case 2:** (Agregar artículo)
        Mediante dos salidas y lecturaS de entrada con *sc* se piden los datos del atículo a agregar:
        Nombre del artíuculo (String) y Precio del artículo (double), los cuales se guardan en dos diferentes variables del tipo correspondiente
        Luego, creamos un nuevo **Articulo** llamado **art**, donde en su **constructor** se ingresan como argumentos las variables donde se guardaron el nombre y precio guardados del Scanner.
        Por último, con el **método** **agregarArticulo(art)** de **carr**, se pone a **art** como argumento para que lo agregue al ArrayList de **carr**.
      - **case 3:** (Eliminar artículo)
        En este caso, se llama al método **printLista()** de **carr** para que el usuario pueda ver la lista con el número del producto (que se guardan conforme se fueron ingresando), luego, pide el número de

artículo que se desea eliminar y lo guarda en una variable que llamamos **elim**, posteriormente, se llama al método **eliminarArticulo(elim)**, donde se le pone como argumento justamente la variable del número de artículo a eliminar

- **case 4:** (Salir)
  Se muestra un mensaje: "**Saliendo...**"
  Logra salir ya que en el **do-while** cuando siga el flujo, se incumplirá la condición de *while* y terminará el programa.
- **default:** (Ninuna de las opciones del menú)
  En este caso, muestra un mensaje de error: "**Opcion no válida**", y sale del switch pero permite volver a iniciar el control **do-while** para ingresar otra opción

4. Resultados

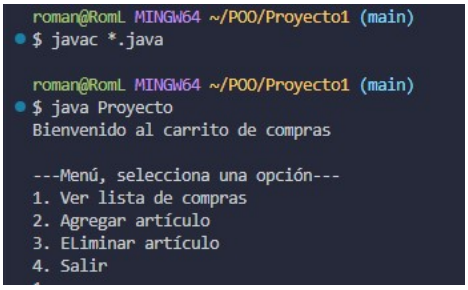


Figura 1: Compilación y ejecucion

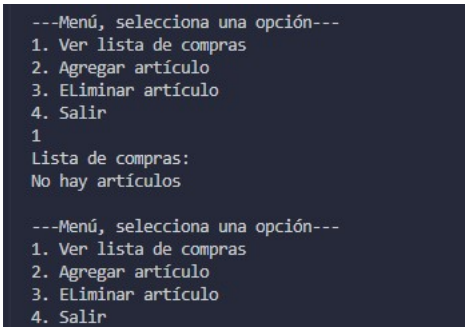
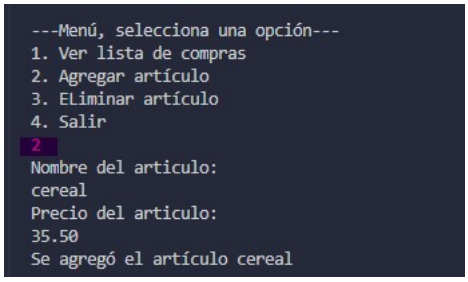
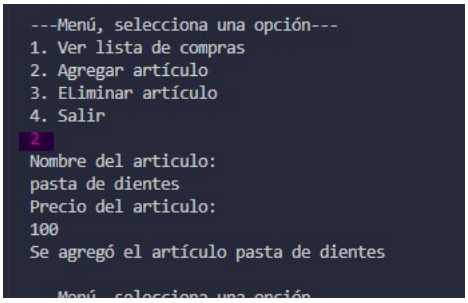


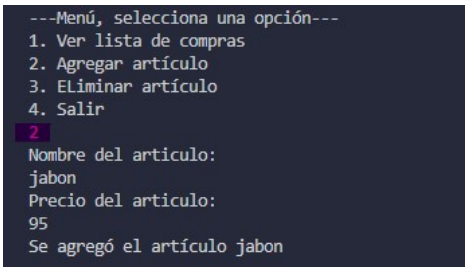
Figura 2: Imprimir la lista, que está vacía al inicio del programa



(a)



(b)



(c)

Figura 3: Agregar varios artículos

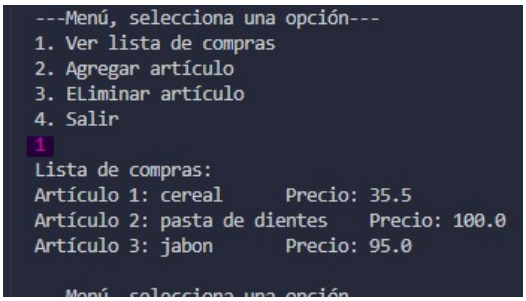


Figura 4: Imprimir lista después de agregar varios elementos

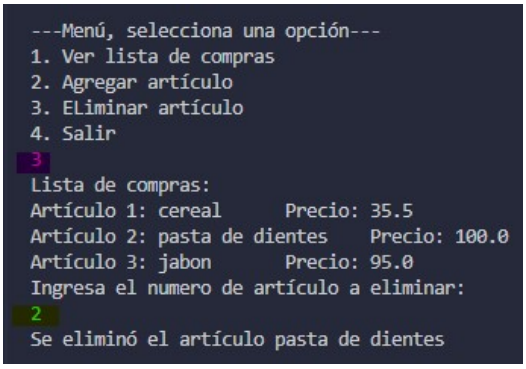


Figura 5: Eliminar artículo

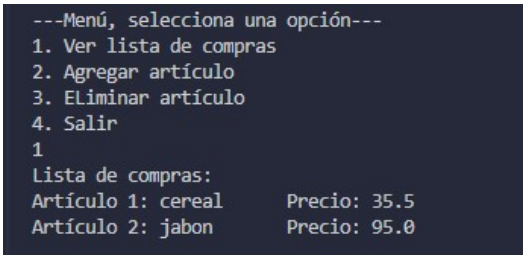


Figura 6: Imprimir lista después de eliminar un artículo

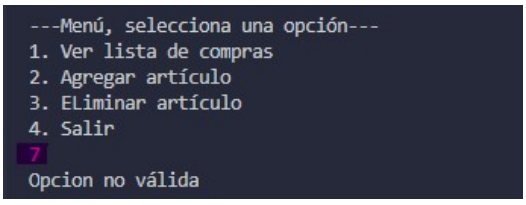


Figura 7: Ingresar una opción no válida

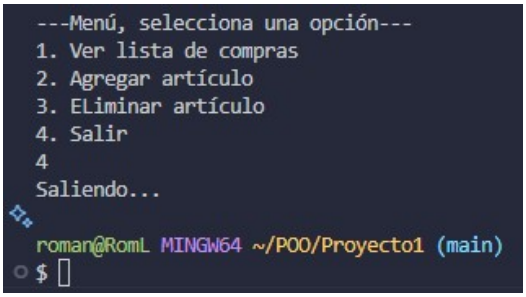


Figura 8: Opción de salir

5. Conclusiones

Esta practica demuestra exitosamente la aplicacion de los principios fundamentales en Java. A traves de la creacio de un simulador de carrito de compras, se logro materializar conceptos teoricos como clases (Articulo, Carrito), objetos, atributos y metodos en una solucion funcional y tangible. La utilizacion del .ArrayList"fue clave para gestionar de manea dinamica la coleecion de producots, permitiendo agregar y eliminar de forma eficiente, lo cual fuerza la comprension sobre el manejo de estructuras de datos Java. La implementaccion de un menu interactivo mediante estructuras de control como do-while y switch permitio desarrollar una logica de programa robusta y una experiencai de usuraio clara. En resumen, la practica no solo consolido el conocimiento tecnico del lenguaje Java y sus librerias, sino que tambien desarrollo habilidades para la resolucion de problemas al modelar una situacion cotidiana.

6. Referencias

[1] Bailón J. Avila, J. Clases y objetos. en análisis y diseño en poo. (https://portalacademico.cch.unam.mx/cibernetica1/analisis-y-diseno-en-poo/clases-y-objetos), 2022. Consultado el 23-09-2025.

[2] M. C. Berenguer. El bucle for en java (y for-each). (https://javautodidacta.es/bucle-for-java/), 2018. Consultado el 25-09-2025.

[3] Alianza BUNAM. Conceptos básicos de programación orientada a objetos. (https://alianza.bunam.unam.mx/cch/conceptos-basicos-de-programacion-orientada-a-objetos/), 2022. Consultado el 23-09-2025.

[4] Nascor. Introducción a los métodos en java: Tipos y funcionamiento. (https://cursosnascor.com/blog-detalle/introduccion-los-metodos-en-java-tipos-y-funcionamiento), 2023. Consultado el 23-09-2025.

[5] Oracle. Class arraylist. (https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html), 2025. Consultado el 25-09-2025.

[6] J. A. Solano. Clases y objetos. (https://repositorio-uapa.cuaed.unam.mx/repositorio/moodle/pluginfile.php/3068/mod$_r esource/content/1/UAPA-Clases-Objetos/index.html), 2020. Consultadoel 23-09-2025.

[7] Baltazar J. M. Ávila J. Estructuras condicionales. (https://portalacademico.cch.unam.mx/cibernetica2/estructuras-condicionales/introduccion), 2023. Consultado el 23-09-2025.