PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 2
10ma práctica (tipo b)
(Segundo Semestre 2024)

Indicaciones Generales:
Duración: **110 minutos.**

- No se permite el uso de apuntes de clase, fotocopias ni material impreso.
- Deberá modular correctamente el proyecto en archivos independientes. Las soluciones deberán desarrollarse bajo un estricto diseño descendente. Cada función no debe sobrepasar las 20 líneas de código aproximadamente. Sólo se permitirá una clase por archivo. La clase Principal solo podrá contener el método main y el código contenido en él solo podrá estar conformado por tareas implementadas como métodos públicos de otras clases. En el archivo Principal.java deberá incluir un comentario en el que coloque claramente su nombre y código, de no hacerlo se le descontarán 0.5 puntos en la nota final.
- El código comentado no se calificará. De igual manera no se calificará el código de una función si esta función no es llamada en ninguna parte del proyecto o su llamado está comentado.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no muestren resultados o que estos no sean coherentes en base al 60%.
- Se tomará en cuenta en la calificación el uso de comentarios relevantes.
- Se les recuerda que, de acuerdo con el reglamento disciplinario de nuestra institución, constituye una falta grave copiar del trabajo realizado por otra persona o cometer plagio.
- No se harán excepciones ante cualquier trasgresión de las indicaciones dadas en la prueba.

Puntaje total: 20 puntos

- La unidad de trabajo será t:\ (Si lo coloca en otra unidad, no se calificará su laboratorio y se le asignará como nota cero). En la unidad de trabajo t:\ colocará el(los) proyecto(s) solicitado(s).
- Cree allí una carpeta con el nombre Lab10_2024_2_CO_PA_PN donde: CO indica: Código del alumno, PA indica: Primer Apellido del alumno y PN indica: primer nombre. De no colocar este requerimiento se le descontarán 3 puntos de la nota final.

Cuestionario

- La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en el capítulo 8 del curso: Introducción al lenguaje de programación Java. En este laboratorio se trabajará con clases abstractas y concretas, métodos abstractos, sobre-escritos y concretos, herencia, polimorfismo, encapsulamiento, lectura de archivos e impresión en pantalla.
- Al finalizar la práctica, comprima la carpeta dada en las indicaciones iniciales empleando el programa Zip que viene por defecto en el Windows, no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares.

Para el diseño de su solución, considere que:

- **Se dispondrá de un único archivo que contiene todos los datos necesarios para la realización del laboratorio.**
- **Se dispondrá de una solución base que contiene la estructura y clases iniciales implementadas de la solución final.**
- **Únicamente puede emplear las clases Scanner y File para leer los datos del archivo de entrada.**
- **Los arreglos dinámicos serán manejados únicamente con ArrayList.**
- **La impresión de los resultados se realizará únicamente en pantalla.**

Descripción del caso

Una de las cafeterías del campus universitario busca optimizar la experiencia de sus estudiantes, con un enfoque especial en los alumnos de intercambio. Para lograrlo, está implementando un programa piloto en Java que permitirá generar menús personalizados según el idioma y la moneda del país de origen de cada estudiante, ofreciendo así un servicio más inclusivo y accesible.

En esta marcha blanca participan tanto alumnos regulares como de intercambio. El programa procesará un archivo que contiene información clave: los países de origen de los estudiantes, incluyendo datos sobre moneda, idioma y tipo de cambio; detalles de los alumnos inscritos; y la lista de productos de la cafetería, con nombres y descripciones de bebidas traducidos a los idiomas correspondientes.

El programa deberá generar dos tipos de menús:
1. Un menú predeterminado completamente en español y con precios expresados en soles peruanos.
2. Menús personalizados para cada estudiante, donde los nombres, descripciones y precios de los productos estarán en el idioma y moneda de su país de origen.

Este enfoque no solo facilitará la adaptación de los estudiantes internacionales, sino que también enriquecerá la experiencia universitaria de todos los participantes.

El programa deberá ser capaz de:

1. **Cargar y procesar la información:** Leer desde un archivo los datos necesarios sobre los países de origen, los alumnos participantes y los productos ofrecidos por la cafetería.
2. **Listar datos iniciales:** Mostrar en pantalla la lista de países y estudiantes inscritos, incluyendo información relevante como idioma, moneda, y tipo de cambio.
3. **Generar menús predeterminados:** Crear un menú estándar completamente en español, con precios expresados en soles peruanos.
4. **Generar menús personalizados:** Crear un menú único para cada alumno participante, adaptando los nombres, descripciones y precios de los productos al idioma y moneda de su país de origen.

Este sistema permitirá a la cafetería gestionar su menú de manera más eficiente, optimizando los tiempos de preparación y minimizando posibles malentendidos en la comunicación con los estudiantes, especialmente aquellos de intercambio. Esto contribuirá a una experiencia más fluida y satisfactoria tanto para los clientes como para el personal.

El archivo de texto **datos.txt** está organizado en tres secciones claramente definidas, separadas por la palabra clave **'FIN'**. La primera sección incluye información detallada sobre los países de origen de los estudiantes participantes, la segunda sección describe los datos específicos de los estudiantes inscritos, y la tercera presenta los detalles de los productos disponibles en la cafetería. Este formato estructurado facilita el procesamiento y análisis de la información por parte del programa.

Anatomía del archivo de datos:

La sección "Países de Origen" contiene los siguientes campos: **tipo** (N para nacionales, I para internacionales), nombre del país, moneda utilizada, tipo de cambio relativo al sol peruano, y el idioma principal del país.

Tipo	Nombre	Moneda	Tipo de cambio	Idioma
N	Perú	PEN	1.00	ES
I	Alemania	EUR	0.24	DE
…	…	…	…	…

La sección "Estudiantes" incluye los siguientes campos: tipo (R para estudiantes regulares, I para estudiantes de intercambio), código único del estudiante, nombre, especialidad, facultad a la que pertenece, país de origen, idiomas que habla, si corresponde al tipo de estudiante, y, en el caso de estudiantes de intercambio, la duración del intercambio en semestres.

Tipo	Código	Nombre	Especialidad	Facultad	País	Idiomas	Duración
R	20120476	Lozada_Yino_Martha	Matemáticas	FCI			
I	4450	Johann_Weber			Austria	Alemán Inglés Castellano Francés Italiano	1
…	…	…	…	…	…	…	…

La sección de productos contiene los siguientes campos: tipo de producto (**B** para bebida, **H** para helado), código, traducciones del nombre en varios idiomas, precio, traducciones de la descripción en diferentes idiomas, stock disponible, tipo específico de bebida (si aplica) (**C** para café, **H** para chocolate, **T** para té e **I** para infusión), cantidad de toppings, y la lista de toppings asociados (en caso de que aplique según el tipo de producto). Esta estructura permite organizar la información de manera clara y detallada para facilitar su uso dentro del sistema.

Tipo	Código	Traducciones del nombre	Precio	Traducciones de la descripción	Stock	T. Beb	C. Top	Toppings
B	B001	ES Espresso_Doble EN Double_Espresso DE Espresso_Doppio FIL Espreso_Dobleng FR Espresso_Doppio JA ダブルエスプレッソ PT Espresso_Duplo UK Double_Espresso	2.50	ES Un_espresso_intenso_con_una_mezcla_de_... EN An_intense_espresso_with_a_blend_of_... DE Ein_intensiver_Espresso_mit_einer_... FIL Isang_matapang_na_espresso_na_may_... FR Un_espresso_intense_avec_un_mélange_de_... JA アラビカとロブスタのブレンドで強い味わいの… PT Um_espresso_intenso_com_uma_mistura_de_... UK An_intense_espresso_with_a_blend_...	120	C		
H	H001	ES Helado_de_Vainilla EN Vanilla_Ice_Cream DE Vanilleeis FIL Sorbetes_ng_Banilya FR Glace_à_la_Vanille JA バニラアイスクリーム PT Sorvete_de_Baunilha UK Vanilla_Ice_Cream	2.50	ES Cremoso_helado_de_vainilla_elaborado_con_... EN Creamy_vanilla_ice_cream_made_with_... DE Cremiges_Vanilleeis_hergestellt_aus_... FIL Malambot_na_sorbetes_ng_banilya_na_... FR Glace_à_la_vanille_crémeuse_faite_avec_des_... JA 高品質な材料で作られたクリーミー… PT Sorvete_de_baunilha_cremoso_feito_com_... UK Creamy_vanilla_ice_cream_made_with_...	120		3	Jarabe_de_Chocolate Confites Trozos_de_Waffle
…	…	…	…	…	…	…	…	…

Cada producto está registrado en una única fila dentro del archivo de datos, con los diferentes campos separados por espacios. El formato de cada fila incluye detalles específicos como el tipo de producto, el código y las traducciones del nombre. Por ejemplo:

B B001 ES [traducción] EN [traducción]...

Este esquema de traducciones se repite para todos los idiomas admitidos. Los precios también están incluidos y siguen el mismo formato, como en:

2.50 ES [traducción] EN [traducción]...

Después de las traducciones y precios, se registran los datos restantes, que incluyen el stock, el tipo de bebida, la cantidad de toppings y los toppings correspondientes. Esta estructura garantiza una representación clara y organizada de la información de cada producto.

Con esta información se solicita elaborar un proyecto en Netbeans cuya clase Principal estará compuesta por el siguiente código:

```
import java.io.FileNotFoundException;

public class Principal {
    public static void main(String[] args) throws FileNotFoundException {
        Cafeteria cafeteria = new Cafeteria("datos.txt");
        cafeteria.cargarPaises();
        cafeteria.imprimirPaises();
        cafeteria.cargarParticipantes();
        cafeteria.imprimirParticipantes();

        cafeteria.cargarMenu();
        cafeteria.imprimirMenuPredeterminado();
        cafeteria.imprimirMenusPersonalizados();
    }
}
```

Definición de clases:

Nombre	Abstracta	Clase Base	Atributos	Métodos
Registro	Si	N/A	N/A	cargar(archivo: Scanner): void -> **abstracto** imprimir(): void -> **abstracto**
Pais	No	Registro	nombre: String moneda: String tipoCambio: double idioma: String	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe getters y setters
Alumno	Si	Registro	código: int nombre: String	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe getPaisOrigen(): String -> abstracto calcularPrecio(producto: Producto, double tipoCambio): double -> abstracto getters y setters
AlumnoRegular	No	Alumno	especialidad: String facultad: String	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe getPaisOrigen(): String -> sobrescribe calcularPrecio(producto: Producto, double tipoCambio): double -> sobrescribe getters y setters
AlumnoIntercambio	No	Alumno	paisOrigen: String idiomas: List<String> numeroDeSemestres: int	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe getPaisOrigen(): String -> sobrescribe calcularPrecio(producto: Producto, double tipoCambio): double -> sobrescribe getters y setters
Traduccion	No	N/A	idioma: String texto: String	getters y setters
Producto	No	Registro	codigo: String nombres: List<Traduccion> descripciones: List<Traduccion> precio: double stock: int	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe obtenerNombre(String idioma): String obtenerDescripcion(String idioma): String imprimir(alumno: Alumno, pais: Pais): void getters y setters
Bebida	No	Producto	tipoBebida: char	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe imprimir(alumno: Alumno, pais: Pais): void -> sobrescribe getters y setters
Helado	No	Producto	toppings: List<String>	cargar(archivo: Scanner): void -> sobrescribe imprimir(): void -> sobrescribe imprimir(alumno: Alumno, pais: Pais): void -> sobrescribe getters y setters

Menu	No	N/A	productos: List<Productos>	cargarProductos(archivo: Scanner): void imprimirMenu(): void imprimirMenuPersonalizado(alumno: Alumno, pais: Pais): void
Cafeteria	No	N/A	paises: List<Pais> participantes: List<Alumno> menu: Menu archivo: Scanner	cargarPaises(): void cargarParticipantes(): void cargarMenu(): void imprimirPaises(): void imprimirParticipantes(): void imprimirMenuPredeterminado(): void imprimirMenusPersonalizados(): void
Principal	No	N/A	N/A	main(String[] args)

Tabla 1 - Definición de clases

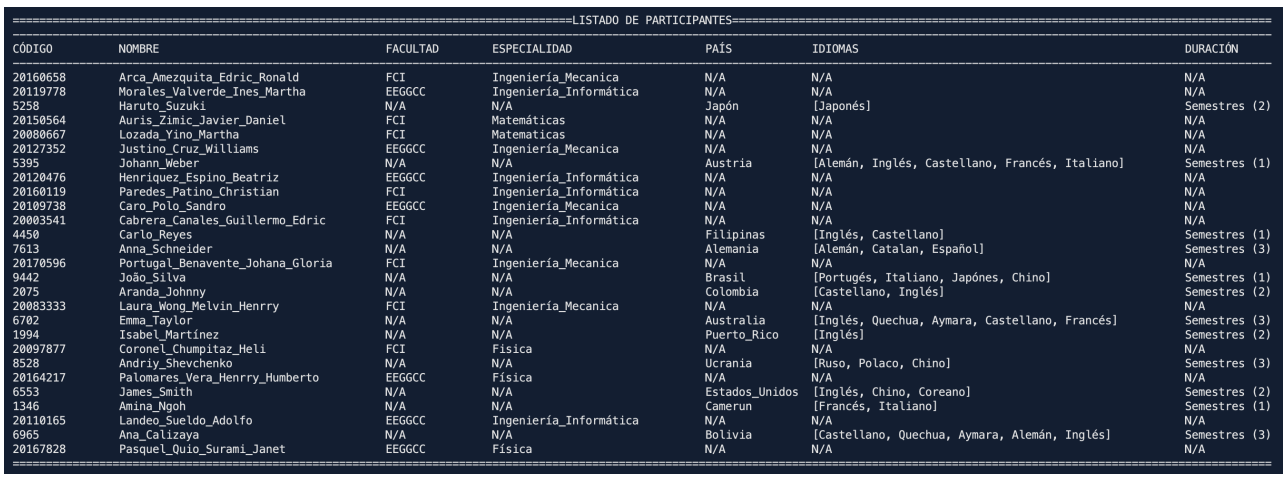
Nota:

- Se ha entregado una solución base que incluye las clases Pais, Traduccion, Registro, Cafeteria y Principal.
- Aunque se permite la creación de métodos o clases adicionales si se considera necesario, este laboratorio ha sido diseñado y probado utilizando exclusivamente las definiciones proporcionadas anteriormente. Por lo tanto, se recomienda limitarse a estas definiciones para garantizar la compatibilidad y evitar posibles problemas durante la implementación y evaluación.

Parte 1 (8 puntos)

Implementación de los métodos **cargarParticipantes** e **imprimirParticipantes** de la clase **Cafetería.**

- **cargarParticipantes (4 puntos)**: El método **cargarParticipantes**, el cual se encargará de leer la sección de estudiantes que participan en el piloto en el archivo **datos.txt**. Tenga que cuenta que las operaciones sobre los objetos Alumno se deben realizar únicamente a través de un objeto de la clase base Alumno e invocando el método polimórfico en estos objetos para la carga de datos.
- **imprimirParticipantes (4 puntos)**: El método **imprimirParticipantes,** el cual se encargará de los participantes en el piloto. Tenga en cuenta que las operaciones sobre los objetos Alumno se deben realizar únicamente a través de un objeto de la clase base Alumno e invocando el método polimórfico en estos objetos para la impresión de datos. A continuación, se muestra una propuesta de impresión en la Figura 2. Es obligatorio que los dato sean impresos de forma clara y organizada.



LISTADO DE PARTICIPANTES

CÓDIGO	NOMBRE	FACULTAD	ESPECIALIDAD	PAÍS	IDIOMAS	DURACIÓN
20160658	Arca_Amezquita_Edric_Ronald	FCI	Ingeniería_Mecanica	N/A	N/A	N/A
20119778	Morales_Valverde_Ines_Martha	EEGGCC	Ingeniería_Informática	N/A	N/A	N/A
5258	Haruto_Suzuki	N/A	N/A	Japón	[Japonés]	Semestres (2)
20150564	Auris_Zimic_Javier_Daniel	FCI	Matemáticas	N/A	N/A	N/A
20080667	Lozada_Yino_Martha	FCI	Matematicas	N/A	N/A	N/A
20127352	Justino_Cruz_Williams	EEGGCC	Ingeniería_Mecanica	N/A	N/A	N/A
5395	Johann_Weber	N/A	N/A	Austria	[Alemán, Inglés, Castellano, Francés, Italiano]	Semestres (1)
20120476	Henriquez_Espino_Beatriz	EEGGCC	Ingeniería_Informática	N/A	N/A	N/A
20160119	Paredes_Patino_Christian	FCI	Ingeniería_Informática	N/A	N/A	N/A
20109738	Caro_Polo_Sandro	EEGGCC	Ingeniería_Mecanica	N/A	N/A	N/A
20003541	Cabrera_Canales_Guillermo_Edric	FCI	Ingeniería_Informática	N/A	N/A	N/A
4450	Carlo_Reyes	N/A	N/A	Filipinas	[Inglés, Castellano]	Semestres (1)
7613	Anna_Schneider	N/A	N/A	Alemania	[Alemán, Catalan, Español]	Semestres (3)
20170596	Portugal_Benavente_Johana_Gloria	FCI	Ingeniería_Mecanica	N/A	N/A	N/A
9442	João_Silva	N/A	N/A	Brasil	[Portugés, Italiano, Japónes, Chino]	Semestres (1)
2075	Aranda_Johnny	N/A	N/A	Colombia	[Castellano, Inglés]	Semestres (2)
20083333	Laura_Wong_Melvin_Henrry	FCI	Ingeniería_Mecanica	N/A	N/A	N/A
6702	Emma_Taylor	N/A	N/A	Australia	[Inglés, Quechua, Aymara, Castellano, Francés]	Semestres (3)
1994	Isabel_Martínez	N/A	N/A	Puerto_Rico	[Inglés]	Semestres (2)
20097877	Coronel_Chumpitaz_Heli	FCI	Fisica	N/A	N/A	N/A
8528	Andriy_Shevchenko	N/A	N/A	Ucrania	[Ruso, Polaco, Chino]	Semestres (3)
20164217	Palomares_Vera_Henrry_Humberto	EEGGCC	Física	N/A	N/A	N/A
6553	James_Smith	N/A	N/A	Estados_Unidos	[Inglés, Chino, Coreano]	Semestres (2)
1346	Amina_Ngoh	N/A	N/A	Camerun	[Francés, Italiano]	Semestres (1)
20110165	Landeo_Sueldo_Adolfo	EEGGCC	Ingeniería_Informática	N/A	N/A	N/A
6965	Ana_Calizaya	N/A	N/A	Bolivia	[Castellano, Quechua, Aymara, Alemán, Inglés]	Semestres (3)
20167828	Pasquel_Quio_Surami_Janet	EEGGCC	Física	N/A	N/A	N/A

Figura 1 - Impresión de los participantes

Se evaluará el correcto uso del encapsulamiento, la herencia y el polimorfismo, así como la reutilización de código en la clase base.

Parte 2 (12 puntos)

Es fundamental resaltar que esta etapa del laboratorio depende directamente de la correcta implementación de las dos primeras partes. Por lo tanto, asegúrese de que esas fases estén completadas antes de proceder.

A continuación, implemente los siguientes métodos en la clase **Cafeteria**:

1. **cargaMenu (4 puntos)**: Este método se encargará de cargar la información de los productos desde la tercera y última sección del archivo datos.txt.
   Para ello, invoca al método **cargarProductos** de la clase **Menu**, el cual recibe un objeto **Scanner** como parámetro. Este método garantiza que los objetos de tipo **Producto** sean instanciados correctamente utilizando la clase derivada adecuada, lo que asegura la correcta aplicación del polimorfismo.

La carga de datos se realizará mediante un método polimórfico específico para la carga de los datos de productos, asegurando que cada tipo de producto se procese de manera apropiada y conforme a su estructura y características particulares. Tenga en cuenta que los nombres y descripciones de los productos son cargados como una lista de traducciones List<Traduccion>.

2. **imprimirMenuPredeterminado (3 puntos)**: Este método imprimirá un menú estándar, presentando los nombres y descripciones de los productos en idioma español, junto con sus precios en nuevos soles.
   Para lograr esto, se invoca al método **imprimirMenu** de la clase **Menu**, el cual recorre la lista de productos disponibles en el menú. Durante esta iteración, se utiliza un método polimórfico específico para imprimir la información de cada producto de manera adecuada según su tipo. Una propuesta de impresión se muestra a continuación en la Figura 3.

```
============================ Menú del día ====================================
B001: Espresso_Doble
Un_espresso_intenso_con_una_mezcla_de_Café_Arábica_y_Robusta_para_un_sabor_equilibrado_y_fuerte
Precio: S/ 2.50
Disponile: Sí
Tipo: Café
------------------------------------------------------------------------------
B002: Café_Latte
Suave_y_cremoso_elaborado_con_una_mezcla_de_cafés_y_leche_vaporizada_para_una_textura_delicada
Precio: S/ 3.20
Disponile: Sí
Tipo: Café
------------------------------------------------------------------------------
```

Figura 2 - Menú Predeterminado

El reporte del menú predeterminado debe incluir el código, nombre, descripción en español, precio en nuevos soles, y disponibilidad (stock mayor a cero). Para bebidas, debe mostrar el tipo (C: Café, H: Chocolate, T: Té, I: Infusión), y para helados, debe detallar los toppings disponibles.

Recomendación: Al imprimir el nombre y la descripción de un producto, se sugiere emplear los métodos **obtenerNombre** y **obtenerDescripción**, especificando el idioma predeterminado, por ejemplo: **obtenerNombre("ES")** o **obtenerDescripción("ES")**, donde "ES" corresponde al código de idioma español. Esto asegura que la información sea consistente con el lenguaje seleccionado para el menú estándar. Estos métodos recorren la lista de traducciones y retornan la traducción de acuerdo con el idioma.

3. **imprimirMenusPersonalizados (5 puntos)**: Este método generará un menú personalizado para cada estudiante según su país de origen, adaptando los nombres y descripciones al idioma correspondiente y mostrando los precios convertidos a la moneda local.

   El método imprimirMenusPersonalizados recorre la lista de alumnos del piloto e invoca el método imprimirMenuPersonalizado de la clase Menu. Este método procesa la lista de productos y llama al método imprimir (alumno, país) de cada producto. Dicho método traduce el nombre y la descripción al idioma del país de origen del estudiante, convierte el precio de nuevos soles a la moneda local usando el tipo de cambio, y reemplaza el prefijo “S/” con el nombre de la moneda.

   El cálculo del precio ajustado se implementa en los métodos calcularPrecio de las clases AlumnoRegular y AlumnoIntercambio. Este método utiliza el producto y el tipo de cambio, aplicando un descuento del 3 % para estudiantes regulares y del 5 % para estudiantes internacionales.

   La Figura 4 muestra una propuesta de menú personalizado.

```
================ Menú en JA(Japón) para, 5258:Haruto_Suzuki ========================
B001: ダブルエスプレッソ
アラビカとロブスタのブレンドで強い味わいのエスプレッソ
Precio: JPY 86.97
Disponile: Sí
Tipo: Café
------------------------------------------------------------------------------------------------------------
B002: カフェラテ
スムーズでクリーミーなコーヒーと蒸したミルクのブレンドで繊細なテクスチャー
Precio: JPY 111.32
Disponile: Sí
Tipo: Café
------------------------------------------------------------------------------------------------------------
```

Figura 3 - Menú Personalizado

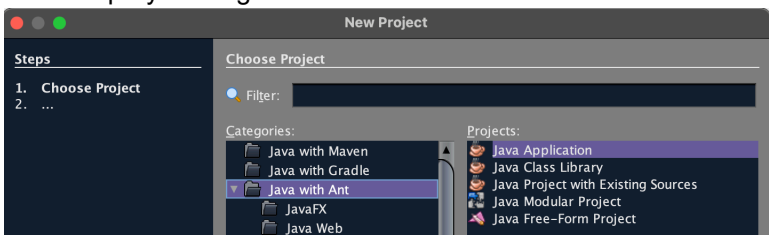
Recomendación:

- Al implementar el método getPaisOrigen en la clase AlumnoRegular, este debe retornar “Perú”, así el código se mantiene estándar para todos los tipos de alumnos.
- Implementar un método buscarPais en la clase Cafeteria el cual recibe el país de origen del alumno, recorre la lista de países y retorna el objeto País.

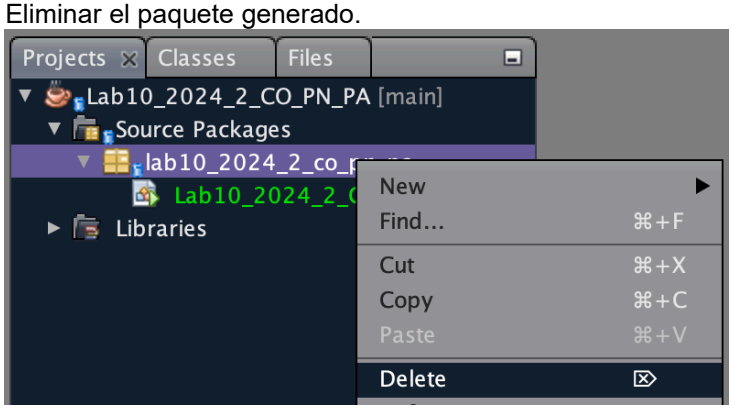
<u>Tenga en cuenta que debe implementar los métodos en las clases utilizadas.</u> Se evaluará el correcto uso del encapsulamiento, la herencia y el polimorfismo, así como la reutilización de código en la clase base.

Apéndice A: Copia de solución base.

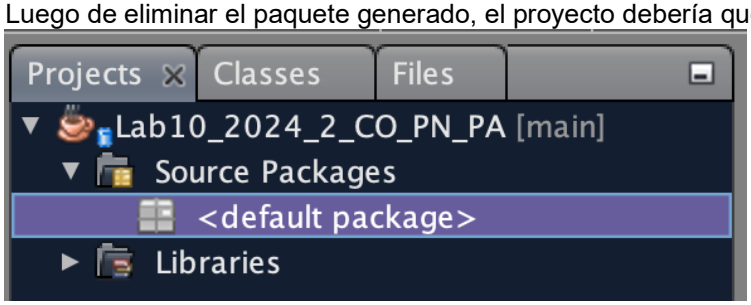
1. Crear el proyecto siguiendo las instrucciones de nombrado: T:\Lab10_2024_2_CO_PA_PN



2. Eliminar el paquete generado.

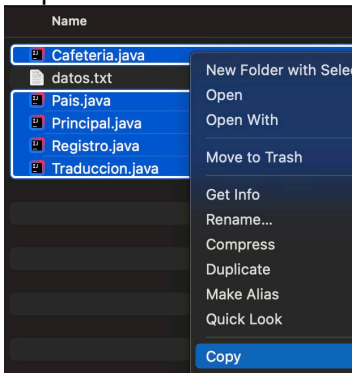


2.1. Luego de eliminar el paquete generado, el proyecto debería quedar así.

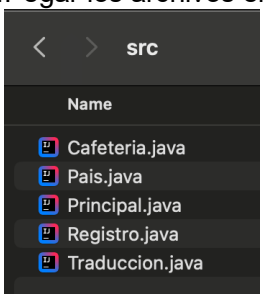


3. Copiar la solución base.

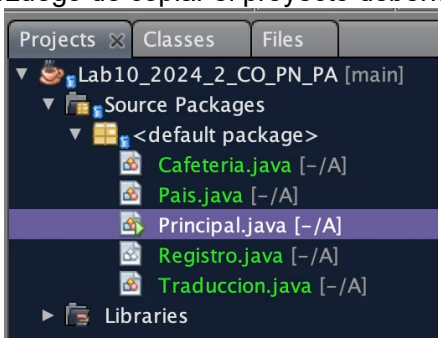
3.1. Copiar los archivos con extensión .java de la solución base que ha sido provista.



3.2. Pegar los archivos en el folder “src” de su proyecto. T:\Lab10_2024_2_CO_PA_PN\src



3.3. Luego de copiar el proyecto debería quedar de la siguiente manera.



3.4. Pegar el archivo datos.txt en la raíz de su proyecto T:\Lab10_2024_2_CO_PA_PN

Profesores del curso: Miguel Guanira, Andrés Melgar, Rony Cueva, Eric Huiza, Erasmo Gómez

Pando, 13 de diciembre de 2024