**FACULTAD DE CIENCIA Y TECNOLOGÍA**

Icono

Descripción generada automáticamente**SAN MIGUEL**

Imagen que contiene Logotipo

Descripción generada automáticamente

CÁTEDRA:

**Programación computacional lll**

CATEDRÁTICO:

**WILLIAN ALEXI MONTES GIRON**

ALUMNOS:

**LILIAN AMARALY PERLA ARIAS - SMSS180923.**

**FLOR GUADALUPE VILLATORO VASQUEZ - SMSS125223.**

### **Código 1: Registro de Asistencia de Estudiantes**

#### **Descripción General:**

Este código implementa un sistema para registrar la asistencia de estudiantes, permitiendo ingresar datos como el nombre del estudiante, el estado de asistencia, y en caso de permiso, la razón del mismo. La información se almacena en un registro y se muestra al usuario.

#### **Clases y Métodos:**

**Clase** AsistenciaEstudiantes**:**

* 1. **Atributos:**
     1. nombre: Nombre del estudiante.
     2. estado\_asistencia: Estado de asistencia ('asistido', 'permiso', 'inasistencia').
     3. fecha: Fecha de la asistencia.
     4. razon\_permiso: Razón del permiso (si aplica).
  2. **Métodos:**
     1. asignar\_fecha(fecha): Asigna la fecha de la asistencia.
     2. asignar\_razon\_permiso(razon): Asigna la razón del permiso si el estado de asistencia es 'permiso'.
     3. mostrar\_info\_asistencia(): Devuelve una cadena con la información completa de la asistencia.

**Funciones:**

* 1. solicitar\_datos\_asistencia(): Solicita al usuario los datos de asistencia y retorna un objeto AsistenciaEstudiantes.
  2. solicitud\_while(): Ejecuta un bucle para registrar y mostrar la asistencia de varios estudiantes, y al finalizar muestra todos los registros.

#### **Explicación:**

* **POO:** La clase AsistenciaEstudiantes permite encapsular la información y operaciones relacionadas con la asistencia. Los métodos permiten gestionar el estado y detalles adicionales de la asistencia.
* **Bucle While:** Se utiliza para permitir el registro continuo hasta que el usuario decida detenerse.

### **Código 2: Reserva de Hotel**

#### **Descripción General:**

Este código gestiona el proceso de reserva en un hotel, incluyendo la selección de servicios adicionales y la generación de facturas. Permite registrar reservas y calcular el total de los gastos, incluyendo servicios extra.

#### **Clases y Métodos:**

**Clase** Reserva**:**

* + **Atributos:**
    - nombre\_cliente: Nombre del cliente.
    - tipo\_habitacion: Tipo de habitación reservada.
    - numero\_noches: Número de noches de la reserva.
    - servicios\_extra: Lista de servicios adicionales y sus costos.
    - total\_gastos: Total de gastos, incluyendo habitación y servicios extra.
  + **Métodos:**
    - agregar\_servicio\_extra(servicio, costo): Agrega un servicio extra y su costo al total.
    - calcular\_total(costo\_habitacion): Calcula el costo total basado en el costo de la habitación y el número de noches.
    - mostrar\_factura(): Devuelve una cadena con la factura detallada.

**Funciones:**

* + solicitar\_datos\_reserva(): Solicita al usuario los datos de la reserva y retorna un objeto Reserva.
  + seleccionar\_servicios\_extra(reserva): Permite al usuario agregar servicios extra a la reserva.
  + repetir\_reservacion(): Ejecuta un bucle para registrar reservas, calcular el total y mostrar facturas. Muestra todas las reservas al final.

#### **Explicación:**

* **POO:** La clase Reserva organiza la información de cada reserva y calcula los costos. Permite agregar servicios adicionales y gestionar el total.
* **Bucle While:** Permite manejar múltiples reservas y ofrece una opción para registrar otra reserva o finalizar.

### **Código 3: Gestión de Planilla de Empleados**

#### **Descripción General:**

Este código maneja el cálculo de pagos para empleados, diferenciando entre empleados con salario fijo y por horas. Incluye un bono adicional para empleados con más de 5 años de servicio.

#### **Clases y Métodos:**

**Clase** Empleado**:**

* + **Atributos:**
    - nombre: Nombre del empleado.
    - años\_trabajados: Años de servicio del empleado.
  + **Métodos:**
    - calcular\_bono\_antigüedad(): Calcula un bono basado en los años trabajados (100 unidades para más de 5 años).
    - calcular\_pago(): Método abstracto para ser implementado por clases derivadas.

**Clase** EmpleadoPlazaFija **(hereda de** Empleado**):**

* + **Atributos:**
    - salario\_base: Salario base del empleado.
    - comisiones: Comisiones adicionales.
  + **Métodos:**
    - calcular\_pago(): Calcula el pago total sumando el salario base, comisiones y bono.

**Clase** EmpleadoPorHoras **(hereda de** Empleado**):**

* + **Atributos:**
    - horas\_trabajadas: Número de horas trabajadas.
    - tarifa\_por\_hora: Tarifa por hora trabajada.
  + **Métodos:**
    - calcular\_pago(): Calcula el pago total basado en las horas trabajadas, tarifa y bono.

**Funciones:**

* + solicitar\_datos\_empleado(): Solicita los datos del empleado y crea el objeto correspondiente.
  + calcular\_planilla(): Ejecuta un bucle para registrar empleados y calcular sus pagos, mostrando los resultados.

#### **Explicación:**

* **POO:** Las clases Empleado, EmpleadoPlazaFija, y EmpleadoPorHoras permiten manejar diferentes tipos de empleados y calcular sus pagos de manera modular.
* **Bucle While:** Facilita la entrada continua de datos de empleados y muestra los pagos calculados.

### **Código 4: Gestión de Animales en un Zoológico**

#### **Descripción General:**

Este código gestiona el registro de animales en un zoológico, incluyendo la asignación de tratamientos y la generación de reportes.

#### **Clases y Métodos:**

**Clase** Animal**:**

* + **Atributos:**
    - nombre: Nombre del animal.
    - tipo: Tipo de animal.
    - edad: Edad del animal.
    - area: Área donde se encuentra el animal en el zoológico.
    - tratamiento: Tratamiento asignado (dosis y frecuencia).
  + **Métodos:**
    - asignar\_tratamiento(dosis, frecuencia): Asigna un tratamiento al animal.
    - \_\_str\_\_(): Representación en cadena del animal.

**Clase** Zoologico**:**

* + **Atributos:**
    - animales: Lista de objetos Animal.
  + **Métodos:**
    - agregar\_animal(animal): Agrega un animal a la lista.
    - mostrar\_reporte(): Muestra un reporte de todos los animales.
    - mostrar\_tratamientos(): Muestra los animales que están en tratamiento.

**Funciones:**

* + bucle\_veterinaria(): Ejecuta un bucle para registrar nuevos animales, mostrar reportes, y gestionar tratamientos. Finaliza cuando se selecciona la opción de salir.

#### **Explicación:**

* **POO:** Las clases Animal y Zoologico permiten gestionar de manera eficiente la información de los animales y sus tratamientos.
* **Modularidad:** La separación de responsabilidades facilita la gestión y actualización de la información.