Práctico 2

Puede **descargar Dev C++** de Embarcadero como IDE de desarrollo.

SIEMPRE:

- **Crear el código fuente y guardarlo en un archivo del tipo apellido_nombre_tp00ejer000.cpp**
  ** Los ceros representan la numeración que les irá dando acorde al número de Práctico y ejercicio.*
- **Comentar el código fuente a fin de dar una mayor legibilidad.**
- **La salida por pantalla debe contener toda la información necesaria para el usuario, a fin de que sea entendible el programa.**
- **Debe figurar en la parte superior de la pantalla el apellido y nombre del estudiante.**

Introducción a C++

Esta sección no está pensada para implementar objetos, sino para que usted gane cierta soltura y confianza con la sintáxis de C++, implementando códigos que contengan condicionales, bucles y arreglos.

1. Crear un programita en C++ que imprima en pantalla "Hola Pepe, tanto tiempo!". Utilizar una constante para guardar el texto.

2. Escribir un programita que muestre en pantalla los límites inferior y superior de los valores que pueden tomar los números enteros. *(INT_MIN, INT_MAX)*

3. Crear un programita que calcule el area de un círculo, para un radio fijo dentro del código. Defina la constante PI.

4. Crear un programa que pida al usuario dos números y devuelva la suma de ambos, y su multiplicación.

5. Crear un programita que calcule el volumen de un cono. Recordar que la fórmula de volúmen de un cono es $V = \frac{1}{3}\pi r^2 h$ donde el radio y la altura deben ser ingresados por el usuario, y se debe definir la constante pi en el código.

6. Crear un programita que determine la nota en el sistema educativo norteamericano, en base al porcentaje sacado en un examen por un alumno. Sabiendo que
   entre 0-59 la nota será F.
   Entre 60-69 la nota será D.
   Entre 70-79 será C.
   Entre 80-89 será B.
   Y finalmente, entre 90-100 la mejor nota, será A.

7. Crear un programita que presente al usuario un menú de bebidas posibles de comprar en el kiosco: jugo de naranja, licuado de banana, licuado verde, jugo apio y manzana, leche de canela. El usuario deberá decir la que prefiere, con un número. El programa deberá mostrar cuál es la bebida que eligió. *(1 → 'Usted se compró un jugo de naranja')*

8. Crear un programita que permita al usuario ingresar una palabra y la imprima 20 veces en pantalla, de dos maneras. La primera vez, todas seguidas *(HolaHolaHola...Hola)*. La segunda vez, una palabra por línea.

9. Pedir una palabra al usuario. Mostrar una letra por línea.

10. Guardar 10 números en un array. Pedir una posición de elemento al usuario y mostrar ese elemento en pantalla.

11. Crear un programita que guarde los gastos totales de los últimos 6 meses, en un array y calcule la suma total de dinero gastada.

12. Crear un programita que tenga definido un array con las 5 mejores marcas de auto. Pedirle al usuario ingresar la marca de su auto, y determinar si su auto está entre las mejores marcas o no. Darle una felicitación o un mensaje de consuelo.

13. Crear un programita que permita al usuario seguir ingresando un número, hasta que el mismo valga 9.

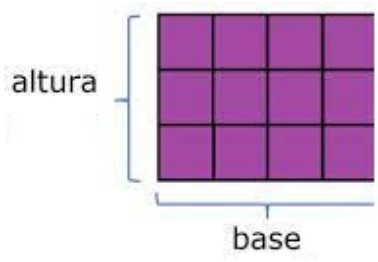
14. Crear un programa que permita encontrar los números perfectos que hay en el rango $[1, 500]$. *Un número perfecto es un número entero positivo que es igual a la suma de sus divisores propios positivos, sin incluirse a si mismo.*
Por ejemplo: el 28.
Los divisores de 28 *(que no sea 28)* son 1, 2, 4, 7, 14
La suma de $1 + 2 + 4 + 7 + 14 = 28$ entonces 28 se considera un número perfecto.

15. Un pangrama o frase holoalfabética es un texto que usa todas las letras posibles del alfabeto de un idioma. *(Ejemplo "Jovencillo emponzoñado de whisky: ¡qué figurota exhibe!")*
Desarrolle un programita que permita chequear si una frase ingresada por el usuario es un pangrama o no. *(* Recuerde que toda frase que conste de varias palabras, tiene espacios que las separa. Corrobore que el usuario no ingrese simplemente todas las letras de alfabeto de corrido y listo.)*

Programación Orientada a Objetos - C++

Los siguientes ejercicios, del práctico anterior o no *implementados en Python*, deberán ser implementados nuevamente, pero esta vez en el lenguaje C++.

16. Dado el Ejercicio del Trabajo Práctico N° 01 - Punto 01 que dice:



Trabajo con Rectangulos.

a) *Crear una clase llamada 'Rectangulo'.*

b) *La clase debe contener los atributos: base y altura.*

c) *Crear los métodos como: calculo_area y calculo_perimetro.*

d) *Se debe crear el objeto correspondiente, y se le debe pedir al operador que ingrese los datos necesarios para que el programa calcule el ´area' y el 'perımetro'.*

17. Dado el Ejercicio del Trabajo Práctico N° 01 - 03 que dice:

Trabajo con Liquidación de Sueldos

a) *Crear una clase llamada 'Empleado'.*

b) *La clase debe contener los atributos: nombre, horas_trabajadas y tarifa_hora.*

c) *Crear el método: calculo_salario.*

d) *Se debe crear el objeto correspondiente, y se le debe pedir al operador que ingrese los datos necesarios para que el programa calcule el sueldo que le corresponde cobrar en función de las horas trabajadas y el valor de la hora trabajada.*

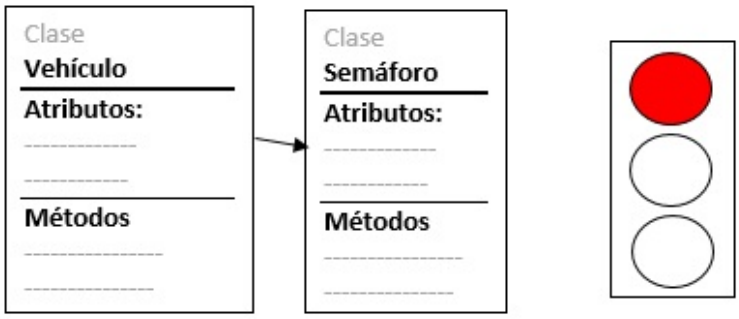
Profundizando OOP en Python

En esta sección, los ejercicios serán implementados nuevamente en Python.

18. Programa que controla el Uso de un Semáforo.

Realizar un programa que tenga por objetivo determinar el estado del semáforo *(rojo, amarillo o verde)* y determine si un automóvil puede avanzar.

a) Crear dos clases, una llamada "Semaforo" y otra llamada "Vehiculo".



b) La clase Semaforo debe tener los siguientes atributos color *(verde, amarillo y rojo)*, encendido *(on, off)* y los métodos cambio_de_color *(pasar al estado: rojo, amarillo o verde)* y cambio_encendido *(pasar del estado ON al OFF)*.

c) La clase "Vehiculo" debe tener los siguientes atributos: marca *(Ej. Ford, Chevrolet, etc.), modelo (F100, Captus, etc.)* y movimiento *(avanzar, detenerse)*.

d) Por cada vez que se ejecuta el programa, la clase semáforo arranca con un color en forma aleatoria *(utilizar la función Random)*. Armar un programa que el usuario ingrese los valores por el teclado sobre la marca y modelo del vehiculo. La clase vehículo debe preguntarle a la clase del semáforo por el estado del color. Si es amarillo o verde el vehículo puede avanzar y se debe mostrar un mensaje, si el semáforo es color rojo el vehículo se debe detener. Comentar el código y salidas por pantalla.

19. Programa de Admisión a la Universidad.

Crear un programa que tenga por objetivo la admisión a la universidad de los estudiantes, para ello el estudiante, debe ser mayor de edad, argentino o nacionalizado, y haber aprobado el nivel secundario. Se deben crear dos clases:

a) Una clase llamada Universidad con su constructor sin atributos propios. Con los métodos: agregar_estudiante() y mostrar_estudiantes()

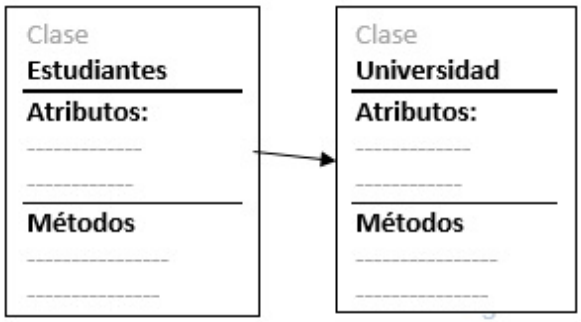


Figura 1: Imagen de la representación rel. clases Estudiantes - Universidad.

b) Crear una clase llamada Estudiante con los atributos: nombre, apellido, edad, dni, nacionalidad, aprobado_secundaria.
Y los métodos: es_mayor_de_edad(), es_argentino_o_nacionalizado(), ha_aprobado_secundaria(), es_admisible(). Los datos deben ser ingresados por el usuario.

20. Programa: Padre, Madre e hijo *(relación genética)*. Crear un programa que tenga por objetivo determinar el color de ojos y color de piel que un hijo heredará de sus padres *(genética)*. El usuario debe ingresar los datos por teclado y mostrar los resultados por pantalla. Comentar el código.

a) Se deberán crear tres clases llamadas: padre, madre e hijo. La clase padre y madre tienen los atributos color_piel: *(claro, oscuro)* y color_ojo. El atributo color_ojo debe cargar en forma aletoria los valores: verde, azul o marrón *(color de ojos)*.

b) Hacer un método que muestre el color de la piel y de los ojos del padre y de la madre.

c) Hacer un programa en donde la clase hijo interactúe con la clase madre y padre, y de la unión de la clase madre y padre la clase hijo tendrá el siguiente color de ojos, dado por:

- verde con verde será verde, azul con azul será azul.
- verde con marron será marron,
- azul con marron será marron.

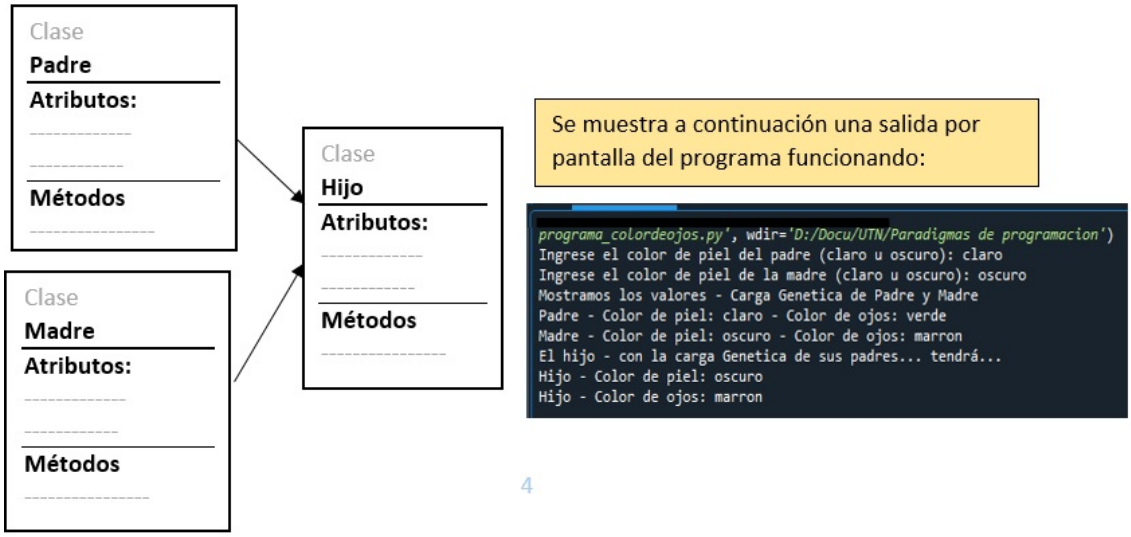


Figura 2: Relación genética, familia.

21. Del Vehículo, al Coche en concreto. Crear un programa que partiendo de una clase padre llamada vehículo la clase hija llamada 'coche' herede todas las propiedades y métodos de la clase padre. Para la clase padre se piden definir los siguientes atributos: marca *(ejemplo Ford)*, modelo *(Ejemplo Ranger)*, year *(year = año, Ej. 2010)* y el método mostrar_informacion_vehiculo.
Para la clase hija, se deben definir los siguientes atributos: marca, modelo, year, cant_puertas. Crear el método llamado: mostrar_info_coche.

Figura 3: Relación entre las clases Vehiculo y Coche.

22. Implementar la clase Vector, que representará el tipo de dato vector matemático y tendrá los siguientes atributos, sus coordenadas x, y, z y los siguienes métodos modulo, sumar *(a un segundo vector que se le de como parámetro)*, restar, versor, multiplicar_escalar *(que tomará un escalar y devolverá un vector resultando de multiplicar dicho escalar en cada elemento del vector original) y producto_escalar con otro vector.*

23. Preguntas de Revisión:
    Nota: Brindamos a modo de guía de estudio, los principales temas que abarca el desarrollo de la temática del módulo. El estudiante puede consultar bibliografía brindada por la cátedra.

    a) ¿Qué es la programación orientada a objetos y cuál es su objetivo principal?
    b) ¿Qué es una clase y cómo se relaciona con un objeto en la programación orientada a objetos?
    c) ¿Qué es un Objeto? ¿Qué características posee?
    d) ¿Qué es el proceso de instanciación? ¿Para qué sirve?
    e) ¿Qué son los métodos y cómo se utilizan en la programación orientada a objetos?
    f) ¿Cuáles son los cuatro pilares principales de la programación orientada a objetos?
    g) ¿Qué es la abstracción?
    h) ¿Qué es la herencia en la programación orientada a objetos y cómo se implementa?
    i) ¿Qué es la encapsulación en la programación orientada a objetos y por qué es importante?
    j) ¿Qué es el polimorfismo en la programación orientada a objetos y cómo se puede utilizar?
    k) ¿Qué es un constructor y cuál es su función en la programación orientada a objetos?

FIN TRABAJO PRÁCTICO N° 02