DOCUMENTACIÓN INTERNA DEMOSTRATIVA DEL OBJETO CONTRACTUAL DEL BOOTCAMP DE NUEVAS TECNOLOGÍAS CON EL OBJETO O ACTIVIDADES MISIONALES DE LA FUNDACIÓN MARINA ORTH

2025

NIT: 900142115-2
Calle 10 Sur N 50FF – 28. Oficina 305- Edificio Primer Plano, Medellín, Colombia.
(57- 4) 322 37 39 - (320) 692 3139
www.fundaorth.org - Contactus@fundaorth.org

Resumen metodológico

El trabajo dentro de los clubes se fundamenta en **metodologías activas**, particularmente el **Aprendizaje Basado en Proyectos (ABP)** y el **Aprendizaje Basado en Problemas (PBL)**. Estas metodologías permiten que los estudiantes sean protagonistas de su propio proceso formativo, enfrentándose a retos reales y construyendo soluciones mediante la experimentación, la creatividad y la colaboración.

En un inicio, se contempla el desafío de integrar a estudiantes nuevos. Para ello, se propone un modelo de nivelación similar al enfoque de *Escuela Nueva*, donde los participantes con mayor experiencia profundizan en nuevos conocimientos y, a la vez, acompañan a quienes se están incorporando, generando un aprendizaje cooperativo.

El trabajo en equipo es un eje central: los estudiantes asumen **roles definidos** dentro de cada grupo (diseñador, ingeniero, organizador, moderador), los cuales deben rotarse periódicamente para garantizar el desarrollo de diversas competencias.

En cada sesión se busca fortalecer tanto las **habilidades técnicas** (pensamiento crítico, resolución de problemas, pensamiento lógico y creativo) como las **habilidades socioemocionales** (liderazgo, comunicación, empatía y gestión del trabajo en equipo), entendidas como elementos clave para el aprendizaje integral y el desarrollo de proyectos con impacto social y comunitario.

Base de nuestra metodología

Desde nuestra metodología de formación en robótica y nuevas tecnologías, cada clase se estructura a partir de **dos fundamentos**:

1. **Guía conceptual**: documento técnico-pedagógico que contiene los fundamentos conceptuales, objetivos de aprendizaje, justificación teórica y técnica, y los materiales necesarios para el desarrollo de la temática en particular. Este insumo garantiza que los profesionales en robótica cuenten con una base sólida y homogénea para la ejecución de las sesiones.
2. **momentos de clase**: **cinco momentos claves**, con actividades específicas que guían el aprendizaje de los estudiantes de manera activa y experiencial.

Momentos de la clase metodología FMO

1. **Saludo y rompehielos**
    a. Inicio de la sesión con una dinámica lúdico-pedagógica breve.
    b. Busca generar un ambiente positivo, despertar motivación y predisponer a los estudiantes para la actividad.
2. **Exploración de conocimientos previos**

a. Espacio para verificar las tareas o actividades asignadas previamente.
b. Permite identificar qué saben los estudiantes sobre la temática, aclarar dudas y activar saberes iniciales como punto de partida.

3. **Introducción y explicación de la temática**
   a. Presentación estructurada de los contenidos conceptuales y técnicos de la sesión.
   b. Incluye tanto la explicación teórica como la preparación para la experiencia práctica (conceptos, procedimientos y objetivos).
4. **Aplicación y experimentación**
   a. Momento central de la clase donde los estudiantes participan activamente mediante la construcción de circuitos, programación, prototipado o desarrollo de aplicaciones.
   b. Se prioriza el aprendizaje experiencial y la producción de un resultado tangible (prototipo o proyecto en avance).
5. **Cierre y evaluación**
   a. Reflexión colectiva sobre lo aprendido y evaluación del proceso (formativa y práctica).
   b. Consolidación de ideas clave, socialización de logros y definición de tareas o pendientes para la siguiente sesión.

Resumen Malla curricular Robótica

1. Electrónica

1.1 Circuitos básicos en serie y paralelo

- **Justificación:** Comprender cómo se distribuye la corriente y el voltaje en configuraciones elementales es esencial para todo sistema electrónico. Esta base permite a los estudiantes diagnosticar fallas, calcular consumos y dimensionar componentes en proyectos de mayor complejidad.
- **Materiales:** Protoboard, resistencias ¼ W, cables Dupont, pila de 9V, multímetro digital.

1.2 Circuitos mixtos

- **Justificación:** Introduce el análisis de redes eléctricas combinadas, fortaleciendo la capacidad de aplicar la Ley de Ohm y las leyes de Kirchhoff. Esto desarrolla habilidades de modelado y abstracción aplicables a cualquier diseño electrónico real.

- **Materiales:** Protoboard, resistencias surtidas, fuentes DC regulables, multímetro.

1.3 Uso de resistencias y medición de corriente/voltaje

- **Justificación:** Permite a los estudiantes interiorizar la relación entre magnitudes eléctricas y el papel de las resistencias en el control de energía. Este aprendizaje es la base de todo diseño de seguridad y eficiencia en circuitos.
- **Materiales:** Multímetro digital, resistencias, fuente DC de laboratorio, protoboard.

1.4 Potenciómetros y resistencias variables

- **Justificación:** Introduce la noción de control analógico y modulación de señales, indispensable para aplicaciones como regulación de brillo, volumen o velocidad.
- **Materiales:** Potenciómetros lineales y logarítmicos, LEDs, protoboard, fuente DC.

1.5 Aplicación práctica con prototipos (ej. LEDs y motores)

- **Justificación:** Integra los conceptos de energía, resistencia y movimiento en un prototipo lúdico. Facilita la transición del análisis teórico a la creación tangible, estimulando creatividad y resolución de problemas.
- **Materiales:** Motores DC, transistores NPN, LEDs, resistencias, batería 9V, protoboard.

1.6 Transistores y etapas de conmutación

- **Justificación:** Introducir el transistor como interruptor controlado sienta la base para comprender electrónica digital, drivers de potencia y microcontroladores.
- **Materiales:** Transistores NPN (ej. 2N2222), resistencias, relés, LEDs, fuente DC.

1.7 Uso de sensores análogos y digitales

- **Justificación:** Sensar el entorno (luz, temperatura, movimiento, humedad) es esencial en IoT y robótica. Se enseña la diferencia entre señales digitales y análogas y cómo traducirlas a datos útiles.
- **Materiales:** DHT22, LDR, PIR, sensor ultrasónico HC-SR04, módulo de humedad del suelo, resistencias de pull-up.

1.8 Manejo de actuadores básicos

- **Justificación:** Permite comprender cómo se traduce la señal eléctrica en acción física. Es la base de la robótica aplicada y del diseño de sistemas de automatización.
- **Materiales:** Relés optoacoplados 5V, buzzers, servos SG90, LEDs, motores DC pequeños.

1.9 Prototipado de sistemas electrónicos integrados

- **Justificación:** Consolida la línea de electrónica en proyectos que combinan sensores y actuadores, aplicando criterios de diseño, seguridad y eficiencia. Desarrolla habilidades de integración de sistemas.
- **Materiales:** Protoboard, microcontrolador (ESP32 o Arduino), sensores varios, actuadores, fuente de 5V regulada (LM2596S).

Ejemplo Guía Conceptual Robótica

ROBÓTICA – Sesión 2 y 3 (circuitos eléctricos en serie y paralelos)

Pregunta orientadora

¿Qué cuidados debemos tener a la hora de manipular circuitos y dispositivos electrónicos?
¿Cuáles son los dispositivos electrónicos básicos y cómo podemos crear nuestro primer circuito electrónico?

Objetivo

- Conocer los principales cuidados que debemos tener a la hora de manipular circuitos y dispositivos electrónicos
- Identificar los dispositivos electrónicos básicos, funciones y manejo y sus características.

Resumen

Habiendo conocido los diferentes tipos de robots y las principales magnitudes en un circuito eléctrico, pasamos ahora a identificar y manipular algunos dispositivos electrónicos, haciendo un uso adecuado de los mismos en pro de adquirir buenas prácticas y crear nuestro primer circuito.

Al completar esta actividad, los estudiantes:

- Sabrán cuales son los cuidados que se deben tener a la hora de ensamblar circuitos electrónicos
- Identificarán los principales dispositivos utilizados en circuitos electrónicos
- Aprenderán a manipular la protoboard
- Crearán su primer circuito

SEGURIDAD Y MANEJO DE CIRCUITOS

A la hora de manipular y realizar montaje de circuitos electrónicos, es de gran importancia para nuestra seguridad y el buen manejo de los dispositivos tener presente los siguientes cuidados básicos:

- No consumir alimentos durante la clase o durante la manipulación de los circuitos
- El área de trabajo debe estar limpia y descongestionada
- Los circuitos electrónicos pueden producir descargas eléctricas, por lo tanto, no se puede trabajar con circuitos en funcionamiento (manipularlos con la fuente de voltaje conectados)
- No se deben bloquear las rutas de salida o pasillos y no se debe correr dentro del taller o área de trabajo
- Hacer uso adecuado de las herramientas de acuerdo con su uso específico (en caso de tener dudas acerca de su uso, siempre preguntar)
- Uno de los accidentes más recurrentes, son las quemaduras con pistolas de silicona y cautín.
    - El cautín debe reposar en un lugar seguro y en su base para soldar en el momento en que no se esté manipulando.
    - Encender la pistola de silicona solo al momento de su uso, y no dejarla encendida sin uso por tiempo prolongado
    - Al poner silicona, darle un tiempo prudente antes de manipular/tocar la pieza
    - Evitar tocar la punta del cautín y de la pistola de silicona

---

Actividad inicial:

Materiales: Tirillas impresas con las normas de seguridad

De acuerdo a la cantidad de estudiantes el docente formará equipos de trabajo y les entregará una o más tirillas con las normas de seguridad y les pedirá que realicen una actuación en la que sea evidente el incumplimiento de las normas que les correspondió. Los equipos tendrán un tiempo para preparar su actuación y la presentarán a los demás compañeros.
Luego de cada actuación se realiza un dialogo donde se trata de mencionar cuales son las normas que deseaba mostrar el equipo presentador.

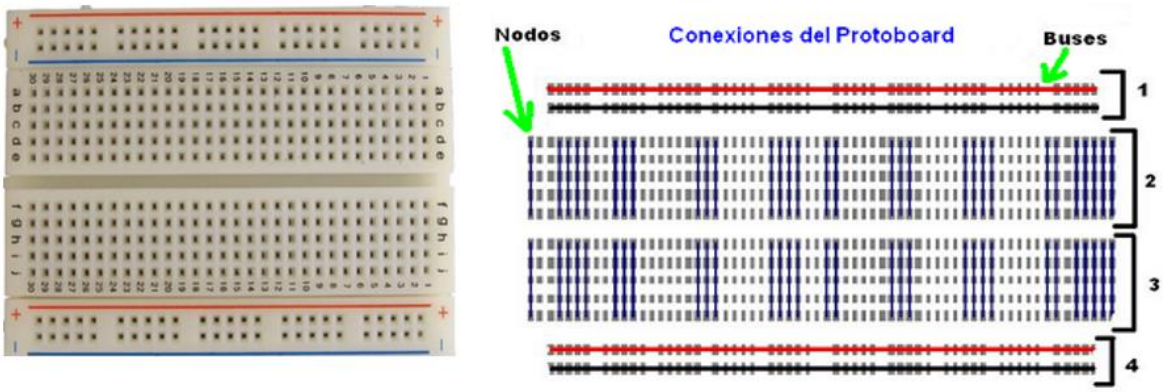
Nota: Si el aula/espacio lo permite, el docente pedirá a un par de estudiantes que, a la par que los compañeros preparan la presentación, ellos realicen una cartelera con las normas básicas (pueden ser varias tirillas de cartulina con una norma en cada una de ellas, o lo que se considere según el espacio)

Al finalizar se establecen acuerdos para el buen manejo y seguridad.

MANEJO DE PROTOBOARD

Antes de iniciar con la creación de circuitos es de suma importancia saber cómo utilizar la protoboard, la cual nos servirá para probar circuitos sin la necesidad de soldar cables y dispositivos.

Veamos:



Imágenes explicativas del funcionamiento de una protoboard

En las anteriores imágenes podemos observar la fotografía de una protoboard y su mapa de conexiones. Tenemos entonces cuatro buses horizontales y entre ellos dos filas de nodos o pistas.

Cada bus tiene un canal metálico interno y lo mismo ocurre con los nodos. Entonces:

- Si conectas un dispositivo en el extremo derecho del bus1, otro dispositivo en el centro del bus1 y uno más en el extremo izquierdo del mismo bus, los tres dispositivos tendrán contacto. Lo mismo ocurre para cada bus.
- Un bus no tiene contacto con otro bus a no ser que los unas por medio de un cable (conductor).
- Vemos en la imagen dos conjuntos de nodos (señalados como **2** y **3**). El conjunto de nodos **2** no tiene contacto con el nodo **3** a no ser que los unas por medio de un cable (conductor).
- Un nodo no tiene contacto con otro nodo a no ser que los unas por medio de un cable.
- Cada punto de un mismo nodo tiene conexión entre sí.

Nota importante.

Un dispositivo electrónico tiene como mínimo dos terminales (paticas). Es importante mencionar que dichas terminales no deben entrar en contacto dentro de un circuito eléctrico (no van ni en el mismo bus ni en el mismo nodo).

Ya estamos a punto de crear nuestro primer circuito eléctrico, para tal motivo utilizaremos:

LED: Su nombre se deriva de las palabras en inglés Ligth Emisor Diod (Diodo Emisor de Luz). Un LED es un dispositivo electrónico semiconductor con determinada composición de materiales que, al ser recorrido por la corriente eléctrica, emite luz, la cual variará su color de acuerdo al material del cual esté compuesto.

Nota: Un led permite el paso de corriente en una sola dirección, por eso debemos aprender a identificar su ánodo+ y cátodo- de acuerdo con la siguiente imagen.

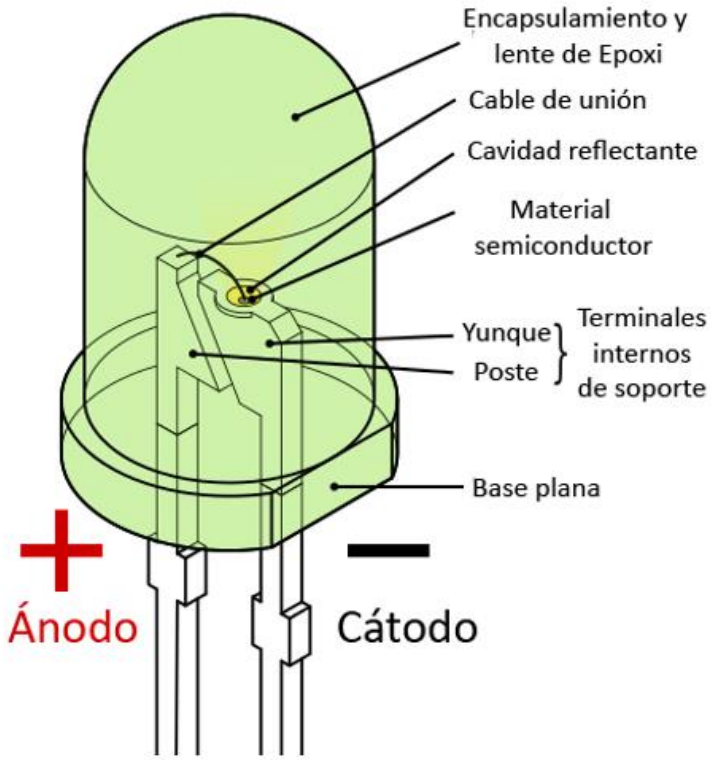


Imagen partes de un Diodo emisor de luz

Resistencia:

Todos los materiales resisten en cierta medida el flujo de corriente y se incluyen en una de dos amplias categorías:

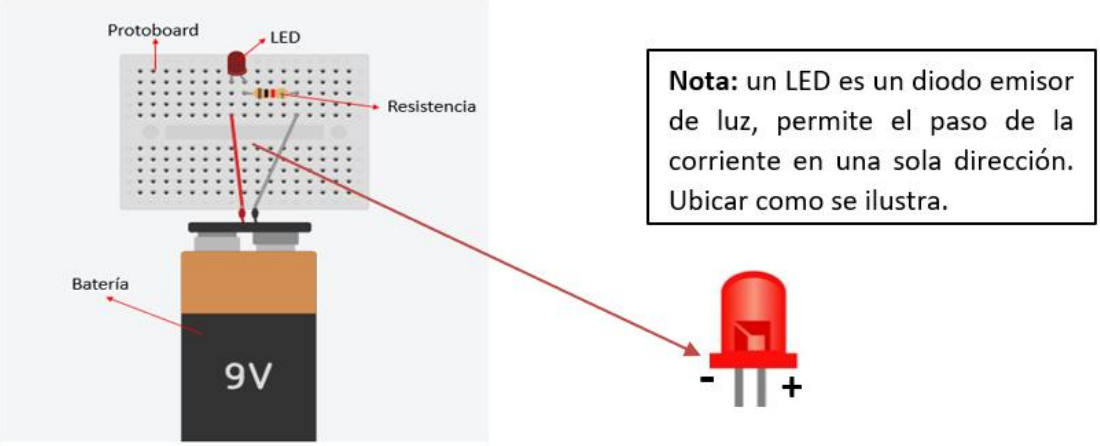
- **Conductores:** materiales que ofrecen muy poca resistencia, donde los electrones pueden moverse fácilmente. Ejemplos: plata, cobre, oro y aluminio.
- **Aislantes:** materiales que presentan alta resistencia y restringen el flujo de electrones. Ejemplos: goma, papel, vidrio, madera y plástico.

En electrónica contamos con un resistor o Resistencia, un componente especialmente diseñado para poner la resistencia requerida entre dos puntos de un circuito eléctrico.

Vamos a la práctica, luego te enseñaremos a saber qué valor de resistencia tenemos de acuerdo a su color; además, en próximas sesiones, aprenderás a calcular la resistencia que requieres en un circuito determinado.

ACTIVIDAD:

Materiales: Protoboard, led blanco, batería de 9v, resistencias de 300, 1000, y 4700 ohms



Esquema de conexión electrónica (funcionamiento de un Led + resistencia + batería)

De acuerdo con lo que pudiste observar, llena la siguiente tabla en un entregable grupal:

Valor de la resistencia	Nivel de luminosidad (Alto, Medio, Bajo)
300 ohms	
1000 ohms	
4700 ohms	

Tabla ejercicio práctico valores de resistencia VS nivel de luminosidad LED

¿Cómo sé qué valor de resistencia debo usar en un circuito?

Empecemos por conocer el valor de las resistencias, con ayuda de la siguiente tabla (conocida como tabla de colotes de resistencias):

Asda



Imagen valores de los colores en resistencias de ¼ W

1ra banda – Café valor: 1

2da banda – Negro: valor: 0

3ra banda – rojo: Multiplicador: x100

Tenemos entonces que 10 x 100= 1000 ohms

El color dorado en la banda final nos indica un 5% de tolerancia. Quiere decir que el valor de la resistencia puede variar entre 950 y 1050 ohms.

Ejemplo:

En nuestra practica utilizamos resistencias de 300 ohms (naranja, negro, café), 1000 ohms (café, negro, rojo) y 4700 ohms (amarillo, violeta, rojo).

Actividad:

Registra el valor para las combinaciones que te proponemos a continuación:

Código de colores	**Valor**
Café, negro, café	
Rojo, rojo, café	
Café, negro, naranja	

Tabla trabajo con resistencias y sus valores

Nota: Estas actividades se pueden recibir como entregables, si así se considera. Agregando la siguiente pregunta:

"Los aprendizajes que estas recibiendo son avanzados para el grado en qué te encuentras. Son inicios de estudios superiores de electrónica. ¿Cómo te sientes con esto y con el conocimiento que estás recibiendo?, ¿Qué te están pareciendo las actividades?"

Recordemos la importancia de argumentar las respuestas.

Resumen malla Curricular – Realidad Aumentada

1. Línea de Fundamentos de AR

1.1 Introducción a la Realidad Aumentada

- *Justificación:* establece la base conceptual y técnica de qué es la AR, su historia, aplicaciones y tecnologías asociadas.
- *Materiales:* dispositivos móviles, gafas de RA, software introductorio (CoSpaces, Delightex).

1.2 Superposición de Objetos y Marcadores

- *Justificación:* introduce el uso de marcadores como técnica fundamental para anclar objetos virtuales en el mundo físico.
- *Materiales:* tarjetas marcadoras, cámara de dispositivos, software AR Toolkit o Vuforia.

2. Línea de Interacción y Experiencia de Usuario

2.1 Interacción y UI en AR

- *Justificación:* enseña cómo diseñar interfaces intuitivas dentro de entornos aumentados, conectando usabilidad con experiencia inmersiva.
- *Materiales:* Unity/Unreal Engine, librerías de UI AR, gafas o tablets.

2.2 Experiencias AR Multiusuario

- *Justificación:* desarrolla competencias en sincronización de entornos aumentados entre múltiples usuarios, clave para aplicaciones colaborativas en educación, salud o industria.
- *Materiales:* dispositivos móviles en red local, frameworks ARCore/ARKit, servidores de sincronización.

3. Línea de Contexto y Espacialidad

3.1 AR Basada en Localización

- *Justificación:* permite a los estudiantes integrar georreferenciación y GPS en experiencias AR, aplicable a turismo, patrimonio o juegos como Pokémon GO.

- *Materiales:* smartphones con GPS, plataformas como Mapbox, Unity con plugins de geolocalización.

3.2 Reconocimiento de Planos

- *Justificación:* enseña cómo detectar superficies físicas y usarlas como anclajes de objetos virtuales, base de la AR inmersiva y realista.
- *Materiales:* ARCore, ARKit, Meta Quest, entornos simulados.

4. Línea de Realismo y Nuevas Interfaces

4.1 Iluminación y Realismo en AR

- *Justificación:* los estudiantes aprenden a trabajar con sombras, reflejos y texturas para lograr mayor naturalidad en la integración del objeto virtual con el entorno físico.
- *Materiales:* motores gráficos avanzados (Unreal/Unity), librerías de sombreado, dispositivos móviles de alto rendimiento.

4.2 RA en Gafas Inteligentes

- *Justificación:* culmina la malla introduciendo el desarrollo en gafas inteligentes, integrando todos los conocimientos previos con aplicaciones en medicina, industria y educación.
- *Materiales:* Meta Quest 3, Unreal Engine, kits de desarrollo AR SDK.

Ejemplo Guía Conceptual Realidad aumentada RA

Introducción a la Realidad Aumentada y Unreal Engine – Modulo 1

Pregunta orientadora

¿Qué es la realidad aumentada?
¿Qué es la Unreal Engine?
¿Cómo aplicar la realidad aumentada usando Unreal Engine?

Objetivo

- Conocer la plataforma de Unreal Engine
- Familiarícese con el ambiente
- Identificar los dispositivos básicos, funciones, manejo y sus características

Resumen

En esta clase introductoria, los estudiantes aprenderán sobre los conceptos básicos de la realidad aumentada y cómo se integra en Unreal Engine. Explorarán la interfaz de Unreal

Engine y comprenderán cómo comenzar a crear proyectos de realidad aumentada. Además, se presentarán a los componentes esenciales de una experiencia AR en Unreal Engine.

Descripción

Formación basada en Retos (se propone un reto desde el inicio del módulo y se le dan a los estudiantes todas las herramientas para que lo solucionen)

Al completar esta actividad, los estudiantes:

- Que es realidad aumentada
- Aplicaciones de la realidad Aumentada
- Que es y para qué sirve unreal Engine 5
- Como descargar e instalar el software unreal engine 5
- Interfaz de unreal engine
- Traducir a un idioma en particular el software
- Iniciar un proyecto de Realidad aumentada con unreal engine
- Entenderán los diferentes entornos y las herramientas básicas de AR

¿Qué es la realidad Aumentada? (RA)

La Realidad Aumentada (RA) es una tecnología que permite la superposición de objetos digitales, como imágenes, videos, sonidos o información en tiempo real, sobre el entorno físico que nos rodea. A través de dispositivos tecnológicos, como smartphones, tabletas, gafas inteligentes o cascos AR, la RA enriquece la percepción del mundo real con datos generados por computadora.

Ejemplos de Aplicabilidad:

- **Aplicaciones de Juegos:** Juegos como "Pokémon GO" permiten a los jugadores cazar y atrapar personajes virtuales que aparecen en el mundo real a través de la pantalla de su teléfono.
- **Publicidad Interactiva:** Las campañas de publicidad pueden utilizar la RA para permitir a los usuarios escanear anuncios con sus dispositivos móviles y ver contenido adicional, como videos o descuentos.
- **Educación y Capacitación:** Las aplicaciones de RA pueden proporcionar experiencias educativas inmersivas, como modelos 3D de anatomía que los estudiantes pueden explorar.
- **Mantenimiento y Reparación:** Los técnicos de campo pueden usar gafas inteligentes con RA para recibir información en tiempo real sobre cómo realizar reparaciones.
- **Navegación:** Las aplicaciones de navegación en automóviles pueden superponer direcciones e información del tráfico en el parabrisas del vehículo.

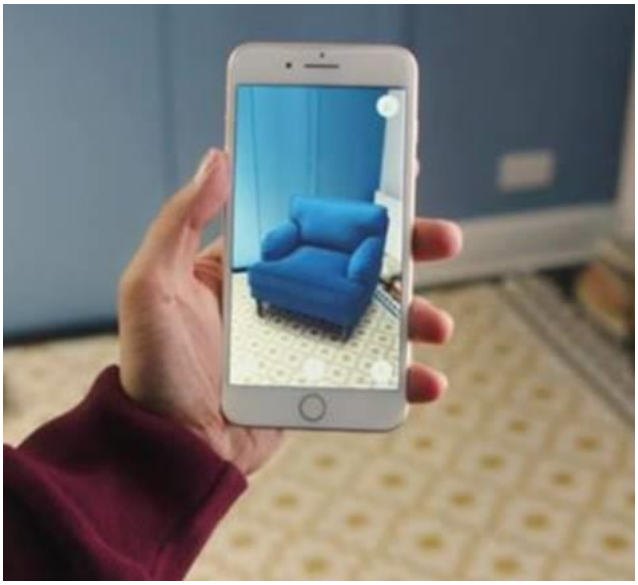


Imagen demostrativa de realidad aumentada

10 Ejemplos de aplicaciones con Realidad Aumentada (link)

Unreal Engine es un motor de desarrollo de videojuegos y entornos virtuales creado por Epic Games. Aunque es ampliamente conocido por su uso en la creación de videojuegos, también es una potente herramienta para el desarrollo de aplicaciones de **Realidad Aumentada (RA)**. A continuación, se resume qué es Unreal Engine y sus puntos fuertes en el ámbito de la RA, con un enfoque para una guía de referencia:

¿Qué es Unreal Engine? Unreal Engine es un motor de desarrollo de software que permite crear experiencias interactivas y visuales de alta calidad en 2D, 3D y RA. Es utilizado por desarrolladores, estudios de videojuegos y profesionales de la industria para crear desde videojuegos hasta simuladores de entrenamiento y aplicaciones de RA. Unreal Engine ofrece un conjunto de herramientas y un entorno de desarrollo que simplifica la creación de experiencias interactivas inmersivas.

Puntos Fuertes de Unreal Engine en la Realidad Aumentada:

- **Gráficos de Alta Calidad:** Unreal Engine es conocido por su capacidad de renderización avanzada, lo que permite crear experiencias de RA con gráficos excepcionales. Esto es esencial para lograr una superposición realista de objetos virtuales en el mundo real.
- **Facilidad de Uso:** A pesar de su potencia, Unreal Engine ofrece un flujo de trabajo intuitivo que facilita el desarrollo de aplicaciones de RA. La interfaz de usuario es accesible incluso para aquellos que no son expertos en programación.
- **Compatibilidad con Múltiples Plataformas:** Unreal Engine permite desarrollar aplicaciones de RA para una amplia gama de plataformas, incluidos dispositivos móviles (Android y iOS), gafas inteligentes y dispositivos de realidad virtual (VR).
- **ARCore y ARKit Integration:** Unreal Engine se integra directamente con ARCore (para Android) y ARKit (para iOS), lo que facilita el desarrollo de aplicaciones de RA para dispositivos móviles.

- **Programación Visual con Blueprints:** Unreal Engine ofrece un sistema de programación visual llamado Blueprints, que permite a los desarrolladores crear lógica y funcionalidades en sus aplicaciones de RA sin necesidad de programar en lenguajes de código tradicionales.
- **Amplia Comunidad y Recursos:** Unreal Engine cuenta con una gran comunidad de desarrolladores y una amplia cantidad de recursos, tutoriales y documentación disponibles en línea. Esto facilita el aprendizaje y la resolución de problemas durante el desarrollo de aplicaciones de RA.
- **Detección de Planos y Superposición Precisa:** Unreal Engine ofrece herramientas avanzadas para la detección de planos y la colocación precisa de objetos virtuales en el mundo real, lo que es esencial para crear experiencias de RA convincentes.

Como descargar unreal engine?

Como DESCARGAR e INSTALAR Unreal Engine 5 / 2023/ 2022 ✅ GRATIS!

Descargar e instalar android 4.0 con APK (se descarga el ejecutabla . Exe de la siguiente pagina https://developer.android.com/studio/archive?hl=es-419)



Imagen de referencia de instalación APK Android Studio

Ruta de instalación de Android estudio

C:\Program Files\Android\Android Studio

Guía de configuración para utilizar SDK de Android con unreal engine5

Guia Instalación Android parte 1
Guía de instalación parte 2

Guia Instalación Android parte 2
Poner celular Android en modo desarrollador para poder correr las aplicaciones en el celular

Cómo activar el MODO DESARROLLADOR en ANDROID 13 - opciones para desarrolladores

Poner un juego en tercera persona y exportarlo como apk para el celular

Actividad: Aplica los conocimientos vistos en clase.

Materiales: Computador

1. Crear un proyecto básico en Unreal Engine
2. Configurar computadora, permisos y software para conexión entre plataforma
3. Instalar y configurar Android Estudio.
4. Exportar un video juego básico al celular con sistema operativo Android

Tarea para la casa: Investigación de programación por bloques, QUE ES BLUEPRINTS Y QUE TIPOS EXISTEN.

Resumen malla Curricular – Diseño e Impresión 3D

1. Línea de Fundamentos del 3D

1.1 Introducción a la impresión 3D y aplicaciones

- *Justificación:* ofrece a los estudiantes una visión general de la tecnología 3D, sus orígenes y campos de aplicación en la industria, la salud, la educación y el arte.
- *Materiales:* presentaciones introductorias, ejemplos físicos de prototipos, videos ilustrativos.

1.2 Software de modelado libre

- *Justificación:* acerca a los estudiantes a herramientas accesibles y gratuitas (Tinkercad, Blender, FreeCAD) como puerta de entrada al modelado digital.
- *Materiales:* computadores con software instalado, acceso a internet.

2. Línea de Modelado Digital

2.1 Principios de modelado 2D (formas básicas, bocetos y extrusión inicial)

- *Justificación:* permite entender cómo los diseños parten de figuras planas antes de convertirse en objetos 3D.
- *Materiales:* Tinkercad, papel y lápiz para bocetos.

2.2 Objetos 3D (extrusión, operaciones booleanas, simetrías y patrones)

- *Justificación:* introduce operaciones avanzadas que permiten diseñar objetos personalizados y funcionales.
- *Materiales:* Blender/FreeCAD, bibliotecas de modelos (Thingiverse).

2.3 Creación de objetos personalizados

- *Justificación:* fomenta la creatividad al transformar ideas en diseños propios listos para imprimir, conectando con proyectos sociales, educativos o de emprendimiento.
- *Materiales:* software CAD, acceso a repositorios de diseños.

3. Línea de Fabricación y Prototipado

3.1 Preparación de modelos para impresión (slicing, parámetros de impresión, soportes)

- *Justificación:* enseña a los estudiantes cómo traducir modelos digitales en archivos imprimibles, entendiendo la importancia de capas, densidad y tiempos.
- *Materiales:* Cura, PrusaSlicer u otro slicer compatible.

3.2 Manejo de impresoras 3D

- *Justificación:* forma en la operación segura y eficiente de impresoras FDM, calibración, mantenimiento básico y resolución de fallas comunes.
- *Materiales:* impresoras 3D (Prusa, Ender, Anycubic), filamento PLA/ABS, kits de mantenimiento.

3.3 Impresión de prototipos

- *Justificación:* pone en práctica el ciclo completo del diseño al objeto físico, afianzando competencias de prototipado rápido.
- *Materiales:* impresoras, filamentos, herramientas de post-procesado (lijas, cortadores).

4. Línea de Aplicación y Evaluación

4.1 Exposición de proyectos 3D

- *Justificación:* desarrolla habilidades de comunicación al presentar el proceso de diseño y los prototipos creados.
- *Materiales:* prototipos impresos, presentaciones visuales.

4.2 Reflexión y aplicación social

- *Justificación:* conecta la tecnología con la resolución de problemáticas reales (educación, sostenibilidad, salud, emprendimiento), evaluando impacto e implicaciones sociales.
- *Materiales:* espacios de diálogo, bitácoras de aprendizaje.

Ejemplo Guía Conceptual diseño 3D

Sesión Nro. 23 – Modelado de piezas 3D

Pregunta orientadora

¿Cómo realizar nuestros propios modelos 3D para impresión?

Objetivo

- Conocer la herramienta tinkercad y aprender a realizar un modelo 3D para impresión

Resumen

Existen diferentes herramientas para el modelado de piezas con la posibilidad de realizar su impresión en 3D. En nuestro caso utilizaremos la herramienta TinkerCad, la cual es una plataforma online de uso libre.

Con ello pondremos en práctica lo aprendido en las sesiones anteriores con los isométricos.

Al completar esta actividad, los estudiantes:

- Conocerán la plataforma TinkerCad y podrán registrarse
- Realizarán su primer modelo en 3D
- Aprenderán a conservar el archivo que les posibilitará realizar la impresión en 3D

Conceptos

Modelado 3D

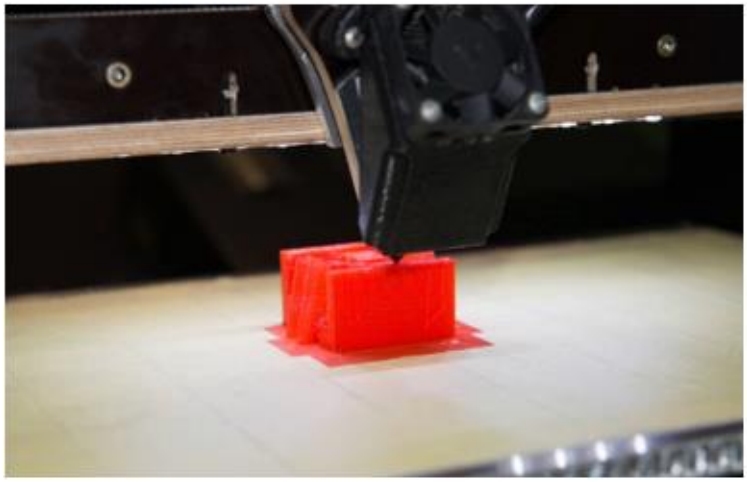
El modelado 3D es una técnica utilizada en diferentes áreas y brinda la posibilidad de llevar nuestras ideas a un objeto tangible, pudiendo transmitir mejor lo que imaginamos o hacer uso del modelo tras una impresión en 3D.

Para el modelado 3D existen diferentes herramientas; nosotros haremos uso de Tinkercad, Software online, con el que podremos modelar los isométricos que hemos elaborado previamente e imprimirlos; adquiriendo así herramientas para futuros proyectos.

Ejemplos de piezas impresas en 3D:



HTTPS://D500.EPIMG.NET/CINCODIAS/IMAGENE *(imagen demostrativa)*



HTTPS://MAQUINAC.COM/WP-CONTENT/UPLOADS/ *(imagen demostrativa)*



https://www.impresion3daily.es/wp-conten (imagen demostrativa)

Impresión 3D:

La impresión 3D también se conoce como manufactura aditiva, porque a diferencia de los substractivos tradicionales de la manufactura, la impresión 3D no remueve material, lo adiciona, capa por capa.
Para poder imprimir algo, primero se requiere un modelo 3D del objeto que desea crear, el cual puede diseñar un software de modelación 3D, o usar un scanner 3D para escanear el objeto a imprimir.

¿Pero cómo funciona la modelación 3D? Aunque existen diversas tecnologías para este tipo de impresión, la mayoría crea un objeto al aplicar sucesivas capas de algún material. Usualmente, las impresoras de escritorio 3D usan filamentos de plástico que alimentan la impresora por la boquilla. El filamento se derrite en la cabeza lo cual filtra el plástico líquido en la placa capa por capa. Una vez la impresora comienza lo único que debes hacer es esperar. (U3D, s.f.)

No.		Actividad	Descripción	Recursos	Tiempo
1	Exploración de conceptos previos	Dialogo con los estudiantes	Se establece un dialogo con los estudiantes, buscando identificar si conocen plataformas de modelado 3D y si han tenido ya una experiencia en ello. También es momento de saber si todos tienen un correo Gmail para poder acceder a TinkerCad, en caso contrario proceder a crearlo.		15 min
3	Acercamiento y construcción de conceptos	Creación de cuentas en tinkerCad y reconocimiento de la interfaz	Ingresar a la página https://www.tinkercad.com y acompañar a los estudiantes en el proceso de registro con sus cuentas de google	Computadores con acceso a internet	15 min
4		Creación de isométrico y almacenamiento del archivo para impresión	Orientar a los estudiantes en el paso a paso para la creación de un isométrico y/o figura en 3D dentro de la plataforma TinkerCad en la sesión Diseños 3D	Computadores con acceso a internet	60 min
7		Valoración y cierre	Se evalúa todo el proceso llevado a cabo hasta el momento. Se pide a los estudiantes que graben un video en el que cuenten cómo les pareció las sesiones de dibujo técnico y diseño 3D y qué creen que esto les aporta a sus vidas. Los videos se almacenan como evidencia.	Celulares	20 min

Descripción de las actividades

Exploración de conceptos previos

1. **Dialogo con estudiantes:** Se establece un dialogo con los estudiantes, buscando identificar si conocen plataformas de modelado 3D y si han tenido ya una experiencia en ello. También es momento de saber si todos tienen un correo Gmail para poder acceder a TinkerCad, en caso contrario proceder a crearlo.

Acercamiento y construcción de conceptos

1. **Creación de cuentas en TinkerCad y reconocimiento de la interfaz:** Ingresar a la página https://www.tinkercad.com y acompañar a los estudiantes en el proceso de registro con sus cuentas de google. En este https://youtu.be/96YLykQ26WE explican cómo registrarse

2. **Creación de isométrico y almacenamiento del archivo para impresión:** Orientar a los estudiantes en el paso a paso para la creación de un isométrico y/o figura en 3D dentro de la plataforma TinkerCad en la sesión Diseños 3D. Además almacenar los archivos para impresión en 3D.

Valoración y cierre

Se evalúa todo el proceso llevado a cabo hasta el momento. Se pide a los estudiantes que graben un video en el que cuenten cómo les pareció las sesiones de dibujo técnico y diseño 3D y qué creen que esto les aporta a sus vidas. Los videos se almacenan como evidencia.

-

Resumen malla Curricular – Programación de microcontroladores

1. Línea de Fundamentos de Programación

1.1 ¿Qué es programación? ¿Qué es un algoritmo?

- *Justificación:* introduce los conceptos esenciales de la programación como herramienta para resolver problemas. Explica qué es un algoritmo y cómo convierte instrucciones humanas en acciones que la máquina puede ejecutar. Este tema construye las bases del pensamiento lógico y computacional.

- *Materiales:* papel para diagramas, computador, editor de texto, ejemplos de algoritmos cotidianos.

2. Línea de Introducción a microcontroladores

2.1 ¿Qué es Arduino y que otros tipos de microcontroladores axisten ?

- *Justificación:* presenta la plataforma Arduino como una herramienta de hardware y software libre, explicando su arquitectura, pines, microcontrolador y funciones básicas. Permite a los estudiantes pasar de la lógica abstracta a la interacción con el mundo físico.
- *Materiales:* placa Arduino UNO, protoboard, LEDs, resistencias, pulsadores, cables Dupont, Arduino IDE.

2,2 Consola o Monitor Serial

- *Justificación:* enseña la comunicación entre Arduino y el computador. Permite visualizar datos en tiempo real, depurar programas y controlar la placa mediante entradas desde el monitor serial. Esta herramienta es esencial para todo proceso de desarrollo y experimentación.
- *Materiales:* Arduino UNO, computador con IDE, cable USB, pulsador, LEDs, resistencias.

3. Línea de Lógica y Control

3.1 Condicionales

- *Justificación:* introduce la estructura fundamental de toma de decisiones en programación. Los condicionales permiten que el programa ejecute diferentes acciones según las entradas recibidas, acercando a los estudiantes a la noción de control inteligente.
- *Materiales:* Arduino UNO, sensores básicos (PIR, LDR), LEDs, protoboard, resistencias.

3.2 Contadores y Acumuladores

- *Justificación:* desarrollan la capacidad de registrar, calcular y procesar datos en el tiempo. Estas estructuras son la base de múltiples aplicaciones como conteo de pulsos, medición de eventos o sistemas de estadísticas simples.
- *Materiales:* Arduino UNO, protoboard, pulsadores, resistencias, LEDs, computador con Tinkercad o Arduino IDE.

4. Línea de Modularización y Reutilización de Código

4.1 Funciones

- *Justificación:* introduce la programación modular, donde se organizan instrucciones en bloques reutilizables. Las funciones permiten simplificar programas complejos,

mejorar la lectura del código y fomentar buenas prácticas en desarrollo de software embebido.

- *Materiales:* Arduino UNO, protoboard, sensores y actuadores diversos según el proyecto.

Ejemplo Guía Conceptual Programación

Tema: Funciones Secundarias o personalizadas.

Sesión: 12 - 13 -14

Objetivo: Aprender a implementar funciones en Arduino

Definición o introducción:

Una función es un bloque de código que sirve para realizar una tarea en especifica, esta puede ser invocada en el algoritmo para realizar una tarea.

Algunos de los beneficios que se tienen al trabajar con funciones son:

1) Organización de código: Las funciones permiten dividir el código en partes más pequeñas y manejables, lo que facilita su comprensión y mantenimiento. Cada función puede encargarse de una tarea específica, lo que simplifica el diseño del programa.

2) Reutilización del Código: Al dividir el código en funciones, se pueden reutilizar esas funciones en diferentes partes del programa o incluso en otros proyectos. Esto ahorra tiempo y esfuerzo al no tener que volver a escribir el mismo código una y otra vez

3) Mejora de la Legibilidad: Utilizar funciones con nombres descriptivos puede mejorar la legibilidad del código, haciendo que sea más fácil entender qué hace cada parte del programa.

Materiales necesarios

- Protoboard
- 3 leds diferentes colores
- LED RGB
- 3 Resistencias 330 Ω
- Cables
- Pelacables
- Pinzas
- Arduino
- Cable de datos para Arduino

Sesión 12: Funciones que reciben y devuelven datos

En esta primera parte veremos un tipo de función, donde requerimos que se manden datos para la rutina de la función y se realicen las operaciones y procesos necesarios luego de realizarlos se devuelve con la palabra “return” los datos ya procesados.

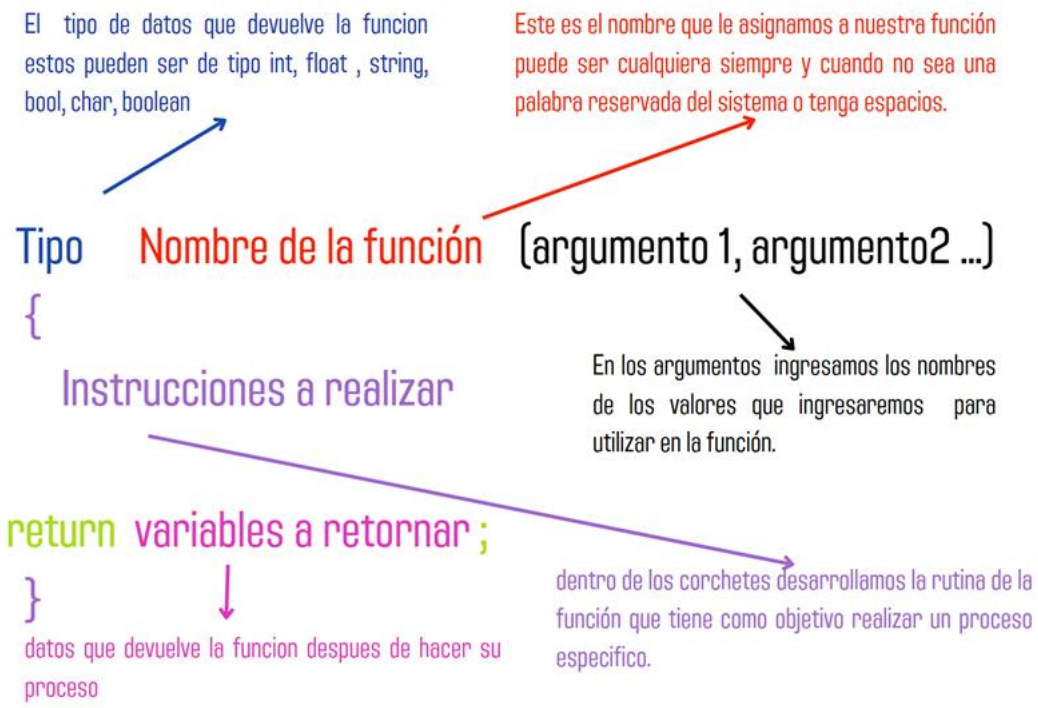


Diagrama de flujo de Funciones que reciben y devuelven datos

Paso a paso funciones que reciben y devuelven datos

En este sencillo ejemplo entenderemos como opera este tipo de funciones en este caso haremos un grupo de 2 funciones del mismo tipo que nos ayudaran a hallar la suma – resta y multiplicación - división de dos números ingresados al sistema por el monitor serial para esto solo necesitamos un Arduino uno como se muestra en la siguiente imagen:

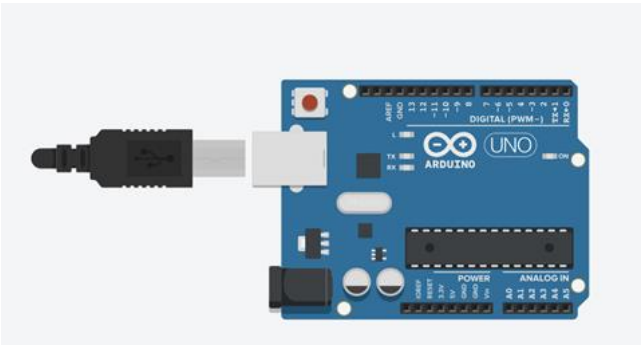


Imagen de referencias tinkercad para la activavion del arduino uno

Nos metemos en el entorno de programación en la opción texto, y borramos todo lo que trae por defecto es software o aplicativo de tinkercad, posteriormente inicializamos en la función principal void setup () el monitor serial para esto utilizamos la línea de código Serial.begin(9600) con esta línea ya el programa sabe que vamos a utilizar el monitor serial. (recuerda tener en cuenta los corchetes de la función y terminar la línea de código en punto y coma)

```
void setup() {
  Serial.begin(9600);
}
```

Imagen explicativa código C++

Podemos poner la o las funciones que utilizaremos después del segmento comprendido en el void setup , o luego del void loop pero no en otra posición, en este caso la utilizaremos después de void setup y crearemos 4 funciones diferentes que hagan las operaciones básicas suma, resta, multiplicación y división, como se muestra en la siguiente imagen:

las funciones creadas son de tipo int (enteros) exceptuando la división que es de tipo float (esto para saber el resultado preciso de la división entre dos números que puede ser decimales) cada nombre de la función es creado de la forma más clara para entender que hace, aunque podrían llevar cualquier nombre. Luego dentro de los paréntesis argumentamos con el tipo de variable que entra a la función y el nombre de dicha variable en este caso son a y b. Dentro de los corchetes realizamos nuestro proceso que en este caso es solo utilizar las operaciones básicas entre dos números a y b para finalizar retornamos el resultado de cada operación con la palabra reservada del sistema “return” que nos da la salida de los datos que necesitamos para posteriormente utilizarlos en nuestro función principal void loop

```
void setup() {
  Serial.begin(9600);
}

int suma(int a, int b) {
  int resultado1 = a + b;
  return resultado1; // Devuelve el resultado de la suma
}
int resta (int a, int b){
 int resultado2= a - b;
 return resultado2;// Devuelve el resultado de la resta
}
int multiplicacion (int a, int b){
 int resultado3= a * b;
 return resultado3;// Devuelve el resultado de la multiplicacion
}
float division (float a, float b){
 float resultado4= a / b;
 return resultado4;// Devuelve el resultado de la division
}
```

Imagen explicativa código C++

Luego en nuestro codigo principal void loop que se esta repitiendo constantemente debemos pedir al usuario que ingrese los datos de los 2 numeros que vamos a operar fijese que los nombres de las variables donde estaran guardados nuestros 2 numeros no son iguales a los nombres que uitilizamos en las funciones que creamos esto se puede hacer poque lo que importa son los datos que ingresamos no las denominaciones que les demos, lo que si es importante es utilizar el mismo tipo de datos, ya sea int, float, char string entre otros debe corresponder a los datos que leemos desde las funciones ya que de no ser asi posiblemente no nos de el resultado con la exactitud que deseamos.

El codigo va de la siguiente manera; en el primer segmento se lee el primer numero, en el segundo segmento leemos el segundo numero en el tercer segmento invocamos nuestras funciones creadas y asignamos el resultado a una variable, y en el cuarto segmento mostramos nuestro resultado por el monitor serial.

```
23 void loop() {
24   Serial.println("Por favor ingrese el primer numero:");
25   while (!Serial.available()); // Espera hasta que haya datos disponibles en el puerto serie
26   int numero1 = Serial.parseInt(); // Lee el primer número ingresado
27
28   Serial.println("Por favor ingrese el segundo numero:");
29   while (!Serial.available()); // Espera hasta que haya datos disponibles en el puerto serie
30   int numero2 = Serial.parseInt(); // Lee el segundo número ingresado
31
32   int result1 = suma(numero1, numero2); // Llama a la función suma y guarda el resultado
33   int result2 = resta(numero1, numero2); // Llama a la función resta y guarda el resultado
34   int result3 = multiplicacion(numero1, numero2); // Llama a la función multiplicacion y guarda el resultado
35   float result4 = division(numero1, numero2); // Llama a la función divicion y guarda el resultado
36
37   Serial.print("los numeros escogidos son: ");
38   Serial.print(numero1);
39   Serial.print("  ");
40   Serial.println(numero2);
41   Serial.println("suma, resta, multiplicacion y division los resultados en orden son: ");
42   Serial.println(result1);
43   Serial.println(result2);
44   Serial.println(result3);
45   Serial.println(result4);
46 }
47
```

Imagen explicativa código C++

Resultado: este sería el resultado de la ejecución del código anterior.

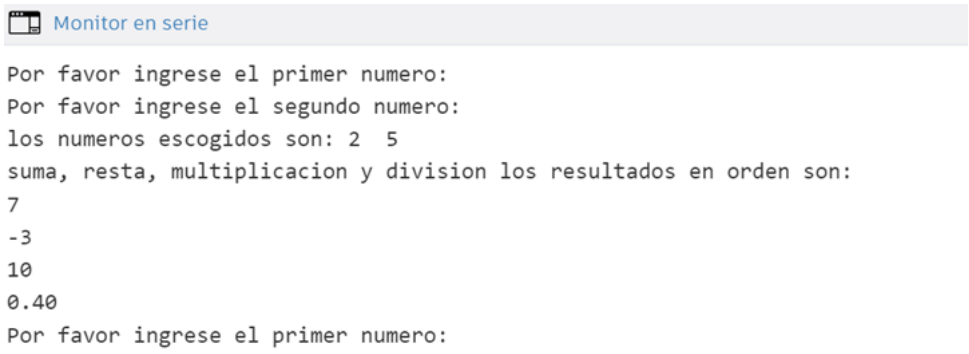


Imagen explicativa datos mostrados por monitor serial

Código: a continuación, el algoritmo completo de funciones que reciben y devuelven datos.

```
void setup() {
 Serial.begin(9600);
}
int suma(int a, int b) {
 int resultado1 = a + b;
 return resultado1; // Devuelve el resultado de la suma
}
int resta (int a, int b){
int resultado2= a - b;
return resultado2;// Devuelve el resultado de la resta
}
int multiplicacion (int a, int b){
int resultado3= a * b;
return resultado3;// Devuelve el resultado de la multiplicacion
}
float division (float a, float b){
float resultado4= a / b;
```

```
return resultado4;// Devuelve el resultado de la division
}

void loop() {
 Serial.println("Por favor ingrese el primer numero:");
 while (!Serial.available()); // Espera hasta que haya datos disponibles en el puerto serie
 int numero1 = Serial.parseInt(); // Lee el primer número ingresado

 Serial.println("Por favor ingrese el segundo numero:");
 while (!Serial.available()); // Espera hasta que haya datos disponibles en el puerto serie
 int numero2 = Serial.parseInt(); // Lee el segundo número ingresado

 int result1 = suma(numero1, numero2); // Llama a la función suma y guarda el resultado
 int result2 = resta(numero1, numero2); // Llama a la función resta y guarda el resultado
 int result3 = multiplicacion(numero1, numero2); // Llama a la función multiplicacion y guarda el resultado
 float result4 = division(numero1, numero2); // Llama a la función divicion y guarda el resultado

 Serial.print("los numeros escogidos son: ");
 Serial.print(numero1);
 Serial.print("  ");
 Serial.println(numero2);
 Serial.println("suma, resta, multiplicacion y division los resultados en orden son: ");
 Serial.println(result1);
 Serial.println(result2);
 Serial.println(result3);
 Serial.println(result4);
}
```

Proyecto tipo reto para funciones con entrada y salida de datos

Reto: Calculadora de Operaciones Básicas por Consola

Descripción:

Crea un programa en Arduino que actúe como una calculadora simple de operaciones básicas (suma, resta, multiplicación y división). El programa debe solicitar al usuario que ingrese dos números y luego le permitirá elegir la operación que desea realizar. Después de realizar la operación, el programa debe mostrar el resultado al usuario.

Instrucciones:

- Solicita al usuario que ingrese dos números.
- Luego, muestra un menú de opciones que permita al usuario elegir la operación que desea realizar:
- 1 para suma
- 2 para resta
- 3 para multiplicación
- 4 para división
- Lee la opción ingresada por el usuario.
- Realiza la operación seleccionada y muestra el resultado.
- Vuelve al paso 2 y permite al usuario realizar más operaciones si lo desea.
- Agrega una opción adicional en el menú para salir del programa.

Sesión 13: Funciones que no reciben datos ni retornan datos

En esta segunda parte el tipo de función que utilizaremos es una que realiza una rutina sin depender de los datos, simplemente es necesario llamar la función personalizada y esta realiza un procedimiento especifico en nuestra función principal void loop.

El objetivo de esta segunda practica será hacer 2 funciones que puedan generar el encendido y apagado de un led, para ello construiremos un circuito y un algoritmo que solucione este planteamiento.

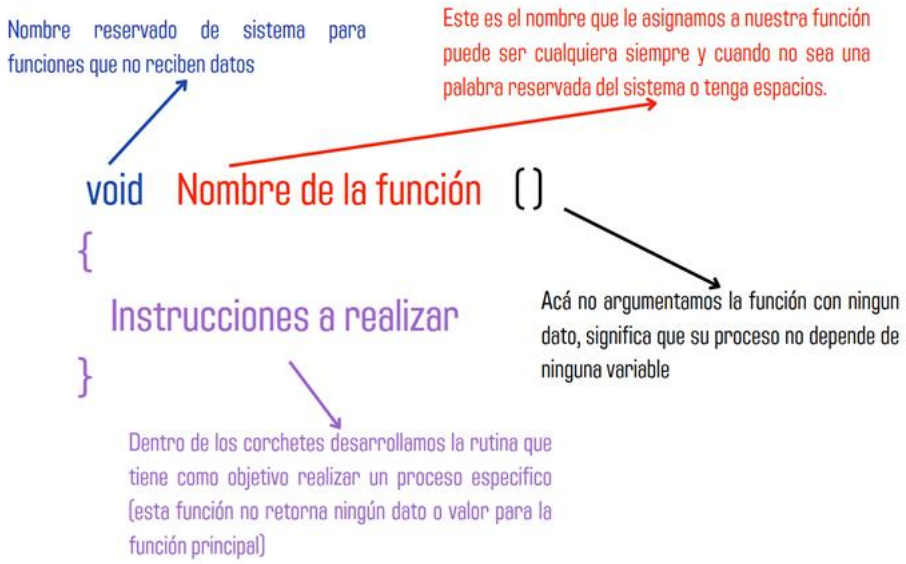


Diagrama de flujo de Funciones que no reciben y no devuelven datos

Paso a paso funciones que no reciben y no devuelven datos

Para nuestra segunda practica utilizaremos una protoboard, 3 leds, 3 resistencias de 330 ohmios, cables y un Arduino como se muestra en la siguiente figura, conectaremos a los pines digitales del Arduino números 9, 10 y 11 los cuales irán en los pines Ánodo de los leds, uniremos de los cátodos de cada led una resistencia y en el otro pin de la resistencia la llevaremos a GND (tierra)

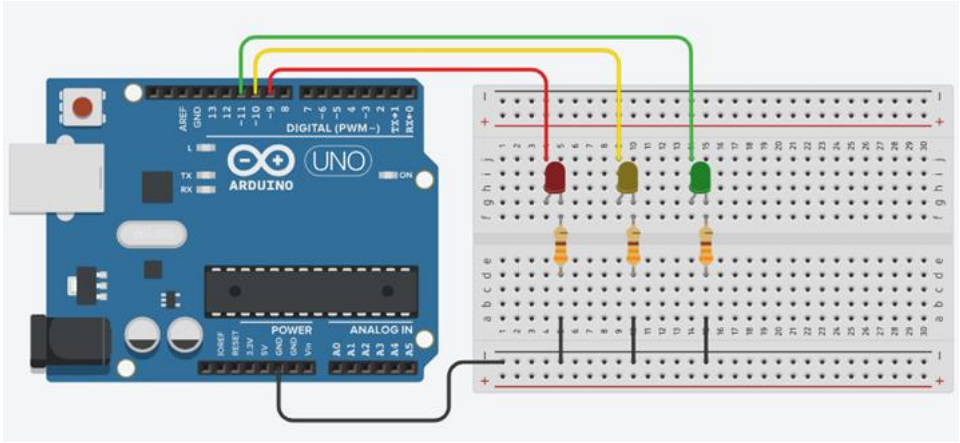


Diagrama eléctrico conexión de 3 leds + resistencias a Arduino

Para nuestro algoritmo lo primero que haremos es definir los nombres de los leds (recuerda que deben ser tipo int) en este caso los definimos todos en una línea para ahorrar en nuestro código, posteriormente inicializaremos nuestros pines 9, 10 y 11 llamados led3, led2 y led1 como salidas en la función principal void setup .

```
int  led1 = 11, led2=10, led3= 9 ;

void setup()
{
 pinMode(11,OUTPUT);
 pinMode(10,OUTPUT);
 pinMode(9,OUTPUT);
}
```

Imagen explicativa código C++

Como había mencionado en la sesión pasada las funciones se pueden crear antes del void loop o después, en este caso las crearemos después del void loop entonces en nuestra función principal llamaremos las funciones creadas que encenderán y apagaran los leds, recuerda que estas funciones no tienen ni entrada ni salida de datos asi que basta con llamarlas y ejecutarlas en nuestro algoritmo, ponemos un delay para poder ver el funcionamiento de nuestros leds las funciones se llaman enceder_led y apagar_led.

```
void loop()
{

  encender_led();
  delay(400);
  apagar_led();
  delay(400);
}
```

Imagen explicativa código C++

Ahora vamos para nuestras funciones las cuales están ubicadas después de void loop, recuerda que como esta función no recibe ni retorna datos entonces va con la palabra del sistema **void,** para formular nuestra función debemos hacer que las variables internas de void encender_led y void apagar_led coincidan con las variables que hemos definido en nuestro algoritmo (led1, led2 y led3), ya que vamos a encender y apagar unos leds en específico, la estructura es muy sencilla, ya que es una rutina que no depende de ningún otro proceso.

```
void encender_led (){
  digitalWrite(led1,HIGH);
  digitalWrite(led2,HIGH);
  digitalWrite(led3,HIGH);

}

void apagar_led (){
  digitalWrite(led1,LOW);
  digitalWrite(led2,LOW);
  digitalWrite(led3,LOW);

}
```

Imagen explicativa código C++

Resultado: el código ejecutado daría como resultado el parpadeo cada 400 mili segundos de tres leds como se muestra en la siguiente imagen:

LEDs encendidos

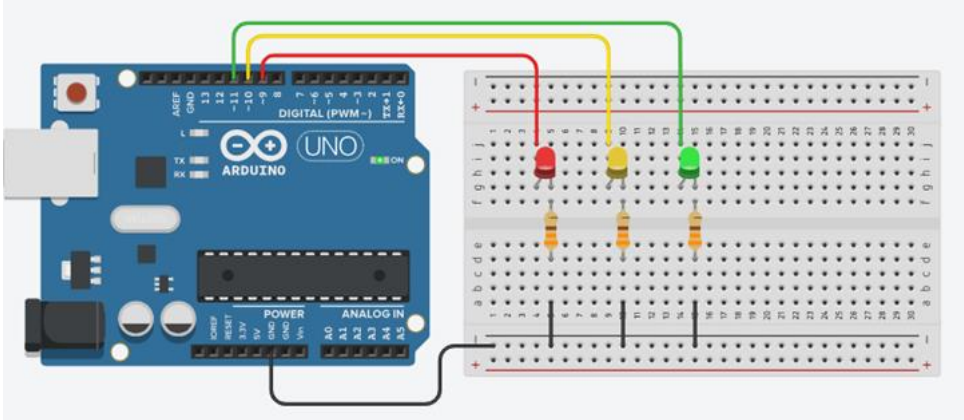


Diagrama eléctrico con leds encendidos

LEDs apagados

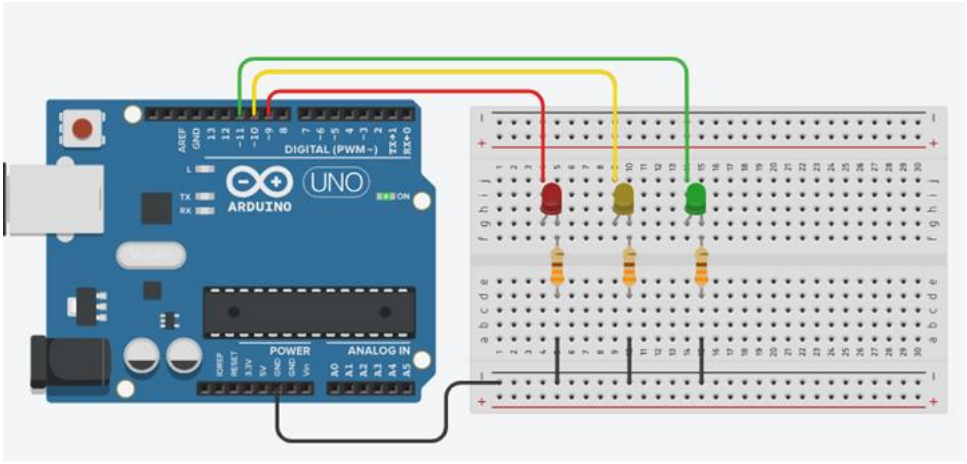


Diagrama eléctrico con leds apagados

Código completo: a continuación, mostramos el código completo del algoritmo presentado para el ejemplo de funciones que no reciben ni devuelven datos

```cpp
int  led1 = 11, led2=10, led3= 9 ;

void setup()
{
 pinMode(11,OUTPUT);
 pinMode(10,OUTPUT);
 pinMode(9,OUTPUT);
}

void loop()
{
  encender_led();
  delay(400);
  apagar_led();
  delay(400);
}

void encender_led (){
  digitalWrite(led1,HIGH);
  digitalWrite(led2,HIGH);
  digitalWrite(led3,HIGH);
}

void apagar_led (){
  digitalWrite(led1,LOW);
  digitalWrite(led2,LOW);
  digitalWrite(led3,LOW);
}
```

Imagen explicativa código C++

Reto: Luces Intermitentes con una Función que no recibe ni devuelve datos

Descripción:

Crea un programa en Arduino que simule el parpadeo intermitente de tres luces LED de manera secuencial lo que significa que cada uno se prende y apaga en instantes de tiempo diferentes. Utiliza solo una función para controlar el parpadeo de todos los LEDs.

Instrucciones:

- Utiliza tres LEDs conectados a pines diferentes de tu placa Arduino (por ejemplo, pines 13, 12 y 11).
- Escribe una función llamada intermitente() que controle el parpadeo intermitente de los LEDs. Esta función no necesita recibir ni devolver datos.
- Dentro de la función intermitente(), alterna entre encender y apagar todos los LEDs de manera secuencial en intervalos de tiempo regulares.
- Utiliza la función delay() para controlar la duración de encendido y apagado de los LEDs.
- Llama a la función intermitente() en el loop() para que el parpadeo de los LEDs se repita continuamente.

Sesión 14: Funciones que reciben y no devuelven datos

En estas funciones podemos enviar parámetros (argumentos) a la función personalizada y dependiendo de estos parámetros que enviemos la rutina cambiara o se ajustara dentro de la función a lo que necesitemos, aunque esto no significa que necesitemos retornar otro valor, ya que el proceso realizado en función nunca cambia.

Veamos en la siguiente sesión como se utiliza las funciones que reciben, pero no devuelven datos para configurar un LED RGB y obtener sus colores principales

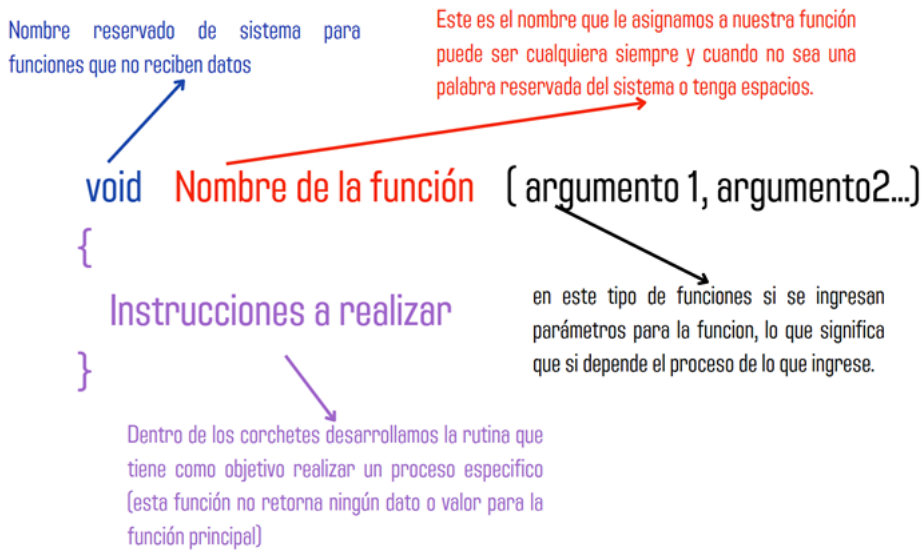


Diagrama de flujo de Funciones que reciben y no devuelven datos

Paso a paso funciones que reciben y no devuelven datos.

Para esta sesión los materiales que necesitaremos un LED RGB, un Arduino uno, una resistencia de 330 ohmios y cables para las conexiones, utilizaremos los pines 2, 3 y 4 digitales de nuestro microcontrolador, conectaremos los pines correspondientes del Arduino a cada uno de los pines del LED RGB (si no recuerdas cómo funciona este tipo de LED te dejo una imagen del esquema interno y su funcionamiento es muy simple dependiendo del pin del LED que se energice tendra un color u otro si se energizan varios pines se obtendrán colores diferentes, los tres colores principales son RED , GREEN , BLUE de ahí su nombre RGB y como todos los leds para evitar que se queme conectaremos una resistencia desde el pin cátodo del led) veamos la conexión eléctrica en la segunda imagen

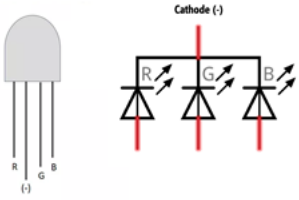


Diagrama eléctrico de funcionamiento de LED RGB

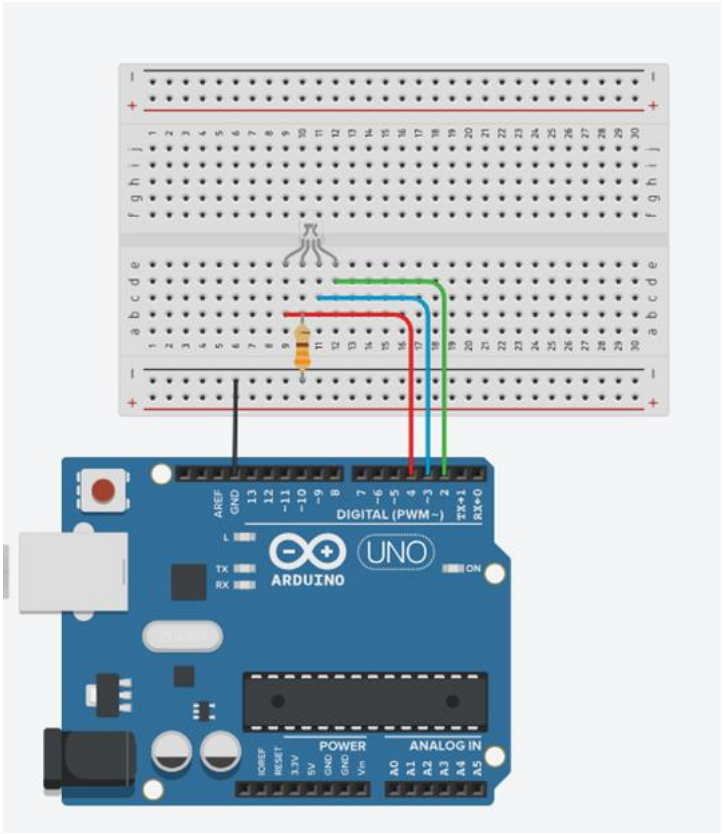


Diagrama eléctrico conexión con Arduino de LED RGB

Como la mayoría de nuestros softwares iniciamos asignando nombres a nuestros pines y declarando si son de entrada y salida (en este caso de salida OUTPUT)

```
const int red=4 , blue=3, green=2 ;
void setup()
{
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}
```

Imagen explicativa código C++

En este caso vamos a crear nuestra función personalizada antes del void loop, (recuerda pueden ir antes o después, nunca antes del void setup) su funcionamiento es muy sencillo leemos unos datos en la función llamada led_rgb estos datos están almacenados en una variable llamada color que es con la cual trabajamos en nuestra función, ahora solo resta encender y apagar nuestro led, pero lo hacemos con la variable color (esta variable es el parámetro que recibe la función personalizada y le da su funcionamiento particular)

```
void led_rgb (int color){

digitalWrite(color,HIGH);
delay(500);
digitalWrite(color,LOW);
delay(500);
}
```

Imagen explicativa código C++

Utilización de la función personalizada en la función loop () para esto basta con escribir el nombre de nuestra función y enviarle el parámetro que necesitamos que en este caso le enviamos las variables red, blue y green que son las asignadas a los pines 4, 3 y 2 digitales de nuestro Arduino

```
void loop()
{
 led_rgb(red);
 led_rgb(blue);
 led_rgb(green);
}
```

Imagen explicativa código C++

Resultado: el código y circuito hecho anteriormente dan como resultado el encendido y apagado secuencial del led RGB donde primero se enciende en color rojo, luego azul y por último verde, todo esto realizado con la misma función led_rgb pero con diferentes argumentos

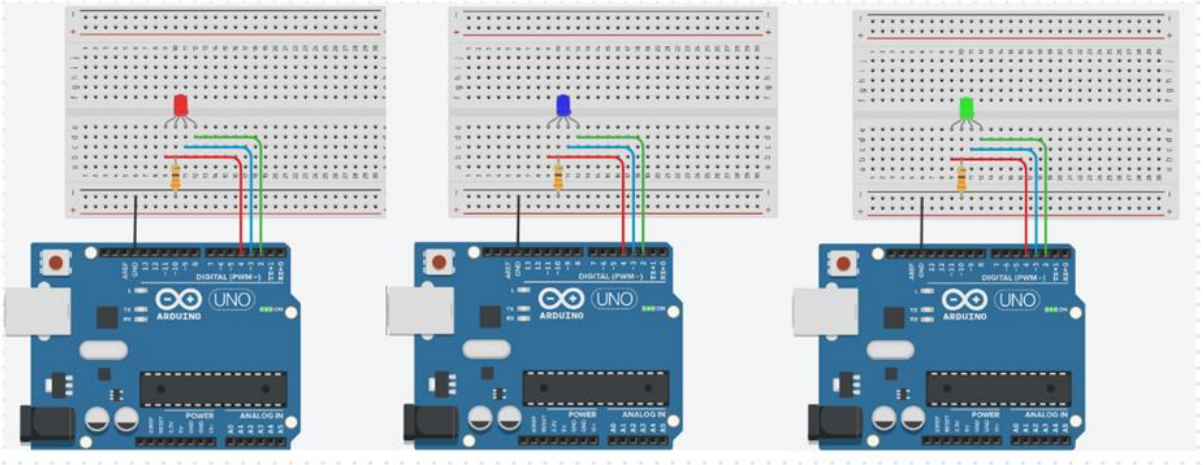


Imagen comparativa variación de color a partir de programación

Código completo: a continuación, mostramos el código completo del algoritmo presentado para el ejemplo de funciones que reciben y no devuelven datos

```
const int red=4 , blue=3, green=2 ;
void setup()
{
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}

void led_rgb (int color){

digitalWrite(color,HIGH);
delay(2000);
digitalWrite(color,LOW);
delay(2000);
}

void loop()
{
 led_rgb(red);
 led_rgb(blue);
 led_rgb(green);
}
```

Imagen explicativa código C++

Reto: completar colore del led RGB con una Función que recibe, pero no devuelve datos

Descripción:

Crea un programa en Arduino que permita a los estudiantes completar el color de un LED RGB utilizando una función que recibe, pero no devuelve datos. Los estudiantes deberán implementar una función que acepte tres parámetros correspondientes a los valores de intensidad de los colores rojo, verde y azul, y que configure el LED RGB con el color resultante.