Nombre Materia
Telecomunicaciones

Nombre del docente
Edwin Celestino Garcia Alcacer

Nombre trabajo
Reporte

Nombre del alumno
Alejandro Guevara de Luna

Unidad
7

Telégrafo

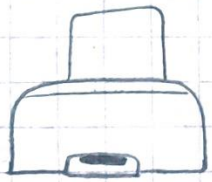
Máquina escribir

1440

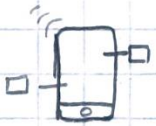
1854

1832

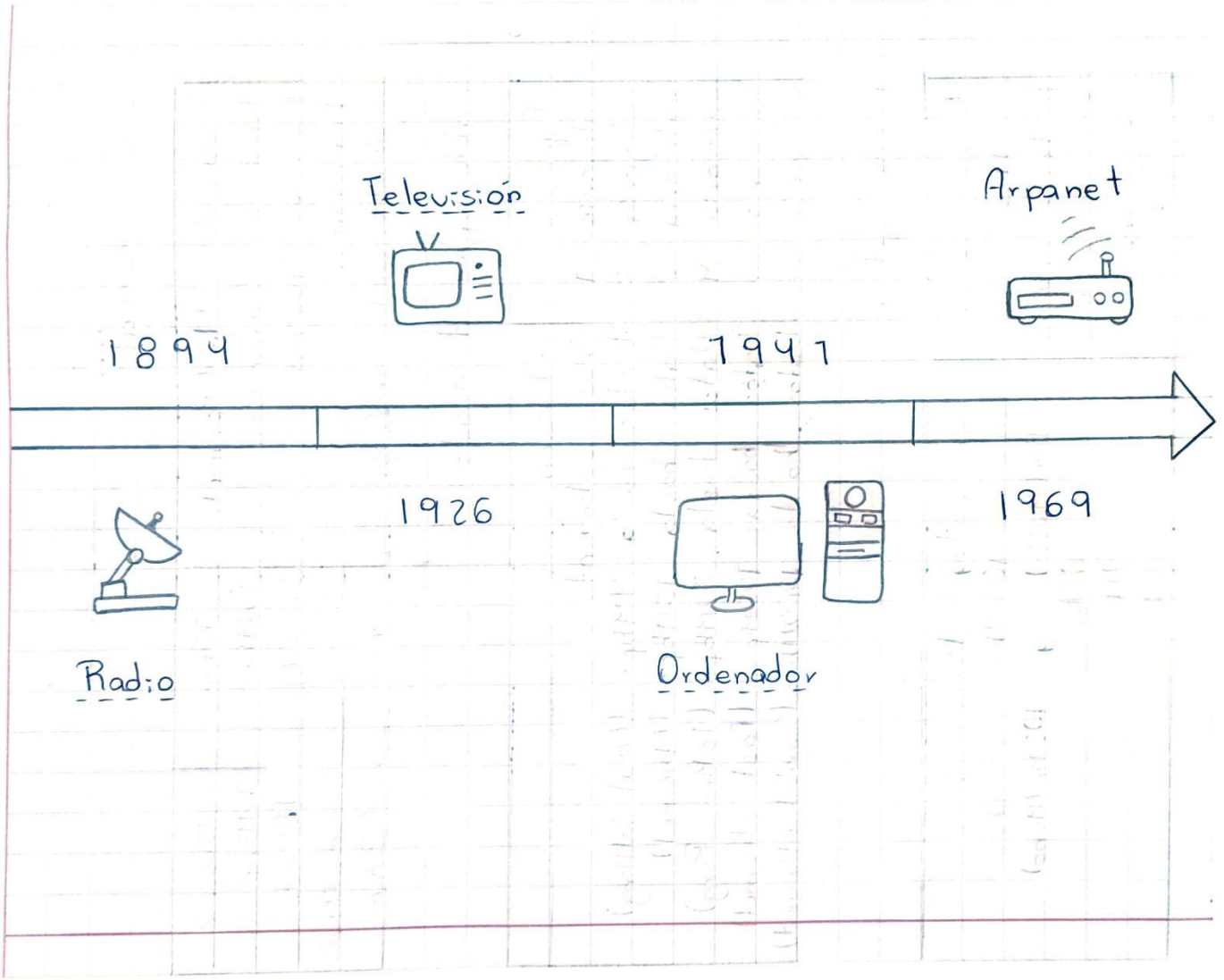
1867



Imprenta



Telefono


Televisión
Arpanet
1894
1941
1926
1969
Radio
Ordenador

Cable coaxial:

Tipo de cable / Parámetro	10base5	10Base2
Tasa de transmisión	10Mbps	10Mbps
Longitud Máxima	500 mts	185 mts
Impedancia	50 ohms	50 ohms RG58
Diámetro del conductor	2.17mm	0.9 mm

Cable par trenzado:

Tipo	Aplicación
Categoría 1	Voz solamente
Categoría 2	Datos hasta 4Mbps (Local talKing)
Categoría 3	Datos hasta 10Mbps (Ethernet)
Categoría 4	Datos hasta 20Mbps (Token Ring)
Categoría 5	Datos hasta 100Mbps (Fast ethernet)
Categoría 5e	Datos hasta 1000Mbps (Gigabit ethernet)

Fibra óptica

Ancho banda Mhz	Muy alto 100Mhz
Distancia medida	2Km (Multi:) y 100 Km (Mono)
Inmunidad elatrica	Alta
Seguridad	Alta
Coste	Alta

Radio enlaces VHF y UHF

Banda	Rango/Frecuencias	Servicios
VLF	3Kz - 30Khz	Conducción electricidad
LF	30Khz - 300Khz	Control aereo, navegación
MF	300Khz - 3Mhz	Radio AM
HF	3Mhz - 30Mhz	Radio SW
VHF	30Mhz - 300Mhz	Radio FM, TV
UHF	300Mhz - 3Ghz	TV, telefonia, Moviles
SHF	3Ghz - 30Ghz	Satélite y microondas
EHF	30Ghz adelante	
Infrarojo	3×1012 , 4.3×1014hz	
Luz visible	4.3×1014 , 7.5×1014hz	
Ultravioleta	7.5×1014 , 3×1017hz	

Microondas terrestres.

Frecuencia	Distancia
15Ghz	24 Kilometros
18Ghz	24 Kilometros
23Ghz	24 Kilometros
26Ghz	24 Kilometros
2-8Ghz	30 y 45 Kilometros

Resumen Manual programación Arduino

Estructura

La estructura es bastante simple y se compone como mínimo de dos partes:

```
void setup(){
    estamentos
}
void loop(){
    estamentos
}
```

setup()

Es el encargado de recoger la configuración. Solo se ejecuta una sola vez y es cuando el programa empieza

loop()

Es el siguiente despues del setup() cicla constantemente el código repitiendolo una y otra vez.

Funciones

Las funciones se declaran asociadas a un tipo de valor "type" por ejemplo :int. Si la función no devuelve ningún valor entonces la palabra se sustituira por un void que significa "función vacia".

La siguiente función devuelve un número entero: delayVal() se utiliza para poner un valor de retraso en un programa que lee una variable analógica de un potenciómetro ejemplo de código.

```
int delayVal(){
    int v;
    v = analogRead(pot);
    v /= 4
    return v;
}
```

```
int v;              // Crea una variable v
v=analogRead(pot)   // Lee el valor del potenciómetro
v /=4               // convierte 0-1023 a 0-255
return v;           // devuelve el valor final
```

{} entre llaves

Las llaves sirven para definir el principio y final de un bloque de instrucciones como setup() y loop().

```
type funcion(){
    estamentos
}
```

; Punto y coma

Se utiliza para separar instrucciones en el lenguaje de programación

```
int x = 2;
```

/*.....*/ Bloque de comentarios

Son áreas de texto que el programa ignora, pero son utiles para los programadores para documentar el código.

```
/* Bloque de comentario
   funciona en varias
   lineas */
```

Variables

Una variable es una manera de nombrar y almacenar un valor numerico para su uso posterior en el programa durante su ejecucion

```
int nombrevariable = 7;          // Declara una variable
nombrevariable = analogRead(7);  // Recoge valor analógico
```

Tipos de variables

byte Valor de 8 bits, Rango 0 a 255
int valor de 16 bits no decimales permitidos
long valor de 32 bits largos. no decimales
float valor de 32 bits, permite decimales

Arrays

Un array es un conjunto de valores a los que se accede con un número indice.

```
int nombre[] = {valor 1, valor 2, valor 3}
```

Operadores de comparación

Se utilizan para comparar una variable o constante, son parte de la estructura de if

```
x == y    // x es igual a y
x != y    // x no es igual a y
x < y     // x es menor que y
x > y     // x es mayor que y
x <= y    // x en menor o igual que y
x >= y    // x es mayor o igual que y
```

Operadores lógicos

Los operadores lógicos son usualmente una forma de comparar dos expresiones y devolver un verdadero o falso.
Existen tres operadores logicos AND(&&), OR(||) y NOT(!)

```
if (x > 0 && x < 5)   // Cierto sólo si dos expresiones son ciertas

if (x > 0 || y > 0)   // Cierto si alguna expresión es cierta

if (!x < 0)           // cierto si la expresión es falsa
```

High / low

Estos son utiles para salidas para lectura y escritura digital de las pastillas Alto significa 5 voltios y bajo significa 0 voltios.

```
digitalWrite(13, HIGH); //Activa la salida 13 con 5 voltios
```

Input / Output

Son constantes son utilizados para definir al comienzo del programa, para funcionar se utiliza el código de pinMode.

```
pinMode(13, OUTPUT); //Designamos el pin 13 es salida
```

analogRead(pin)

Lee el valor de un determinado pin definido como entrada analógica con una resolución de 10 bits. El rango del valor es de 0 a 1023

```
valor = analogRead(pin); //Asigna a valor lo que lee en pin
```

AnalogWrite(pin, value)

Funciona para escribir un pseudo-valor analógico, utilizando el procedimiento de modulación por ancho de pulso

```
analogWrite(pin, valor); //escribe 'valor' en el pin
```

Delay(ms)

Detiene la ejecución del programa la cantidad de tiempo que el usuario indica.

```
delay(1000)  // espera 1 segundo
```

Serial.println(data)

Imprime los datos del puerto serie

```
Serial.println(analogRead(0));
```

Serial.Read()

Lee o captura un byte(un caracter) des el puerto serie.

```
incomingByte = Serial.read(); //lee byte entrada
```