

Funciones necesarias:

Para el ultrasonidos será necesaria una función para medir la distancia a los objetos. La función devolverá la distancia y recibirá los pines del Trigger, que manda el pulso de ultrasonidos, y del Echo, que los recibe:

```
int medir_distancias (int TriggerPin, int EchoPin)
{
    long tiempo, distancia_cm;
    digitalWrite (TriggerPin, LOW);
    delayMicroseconds (4);
    digitalWrite (TriggerPin, HIGH);
    delayMicroseconds (10); //El Trigger envía una señal durante 10 microsegundos.
    digitalWrite (TriggerPin, LOW);

    tiempo = pulseIn (EchoPin HIGH); //Mide el tiempo que tarda el EchoPin en pasar
                                     de LOW a HIGH.
    distancia_cm = (tiempo * 1 / 29.2) / 2; //Se calcula la distancia a partir del tiempo y la
                                             velocidad del sonido y se pasa a
                                             centímetros.

    return (distancia_cm);
}
```

Para los leds se pueden hacer funciones que los enciendan y los apaguen:

```
void encender (int LedPin)
{
    digitalWrite (LedPin, HIGH);
}

void apagar (int LedPin)
{
    digitalWrite (LedPin, LOW);
}
```

Se necesitará una función para elegir cuál es el led que se enciende según el dato que le llegue del puerto serie:

```
int elegir_pin (char c)
{
    int resultado;

    switch (c)
    {
        case '0': resultado=LedPinAmarillo; break;
        case '1': resultado=LedPinAzul; break;
        case '2': resultado=LedPinVerde; break;
        case '3': resultado=LedPinRojo; break;
    }
}
```

```

}
return (resultado);
}

```

Para el teclado es necesaria una función que lea las letras que se presionan. Se utiliza la función `teclado.getKey()` que pertenece a la librería `Keypad.h`.

Para la pantalla LCD utilizaremos una función para escribir en la pantalla lo que se haya introducido por teclado y otra para borrar la pantalla cuando se introduzca una contraseña incorrecta o se pulse un botón determinado:

-Función para escribir en la pantalla:

La función tendrá como entrada una cadena de caracteres en la que se habrían guardado las teclas presionadas en el teclado por el usuario.

```

void escribir_en_lcd (char caracter, int *posicion_x, int *posicion_y)
{
    int i;

    for (i=0;i<N;i++) //N es el número de caracteres que tiene la contraseña.
    {
        lcd.setCursor (*posicion_x, *posicion_y); //Se coloca el cursor en la
                                                    (posicion_x,1).
        lcd.print (caracter); //Se imprime el caracter en la pantalla.
        (*posicion_x)++; //Se aumenta la posición de las x para continuar
                        escribiendo.
    }
}

```

-Función para borrar la pantalla:

Para borrar la pantalla hay una función en la librería `LiquidCrystal` que es `lcd.clear ()`. Esta función borra la pantalla y coloca el cursor arriba a la izquierda de la pantalla.

Además, usaremos una función para comparar el código correcto con el introducido por el usuario. Esta función devolverá 1 o 0 en función de si la clave es correcta o no:

```

int comprobacion_clave (char cadena[ ], char codigo[ ])
{
    int resultado=1, i;

    for (i=0;i<N;i++)

```

```

    {
        if (cadena[i]!=codigo[i])
            resultado=0;
    }
    return (resultado);
}

```

Funciones para contar el tiempo:

Será necesaria la librería time.h. Para estas funciones también usaremos funciones de la librería time.h como now() que guarda el tiempo en ese momento.

Usaremos una estructura tiempo:

```

typedef struct
{
    float tiempo_inicial, tiempo_final;
    float intervalo_de_tiempo;
}tiempo;

```

La primera función será para inicializar el tiempo inicial y el final a 0:

```

tiempo inicializar_tiempo (tiempo t)
{
    t.tiempo_inicial=now();
    t.tiempo_final=now();
    t.intervalo_de_tiempo=(float) (t.tiempo_final-t.tiempo_inicial);
    return t;
}

```

La segunda función guarda el tiempo actual en tiempo_final y calcula el intervalo de tiempo:

```

tiempo tiempo_transcurrido (tiempo t)
{
    t.tiempo_final=now();
    t.intervalo_de_tiempo= (float) (t.tiempo_final-t.tiempo_inicial);
    return t;
}

```