Materia:

DISEÑO ELECTRÓNICO BASADO EN SISTEMAS EMBEBIDOS

Alumno:

Posadas Pérez Isaac Sayeg

Paniagua Rico Juan Julian

García Azzúa Jorge Roberto

Grado y grupo:

8°G

Profesor:

Garcia Ruiz Alejandro Humberto

Tarea 8:

Arreglos en Arduino

Arreglos en Arduino: Concepto, Definición y Ejemplos

Los arreglos (o arrays) son estructuras de datos fundamentales en la programación que permiten almacenar múltiples valores en una sola variable. En Arduino, los arreglos son especialmente útiles para gestionar y manipular colecciones de datos, como lecturas de sensores, valores de entrada, o configuraciones. Esta investigación abordará el concepto de arreglos, su sintaxis, tipos y ejemplos de uso en proyectos de Arduino.

1. Concepto de Arreglos

Un arreglo es una colección de elementos del mismo tipo que se almacenan en ubicaciones contiguas de memoria. Cada elemento del arreglo se identifica mediante un índice, que comienza en 0. Esto significa que un arreglo con `n` elementos tendrá índices que van desde `0` hasta `n-1`.

Ventajas de Usar Arreglos

- **Eficiencia en el manejo de datos**: Los arreglos permiten almacenar y acceder a múltiples valores de forma organizada y eficiente.
- **Acceso rápido**: Los elementos de un arreglo se pueden acceder directamente mediante su índice, lo que permite una recuperación rápida de datos.
- **Facilidad para la iteración**: Los arreglos facilitan la iteración sobre un conjunto de datos, lo que es útil en situaciones como la visualización de datos o la aplicación de algoritmos.

2. Sintaxis de los Arreglos en Arduino

Declaración de un Arreglo

La declaración de un arreglo en Arduino se realiza especificando el tipo de datos, el nombre del arreglo y el tamaño entre corchetes.

Sintaxis:

```
tipo_de_dato nombre_del_arreglo[tamaño];
```

Ejemplo de Declaración de un Arreglo

```
int valores[5]; // Declara un arreglo de enteros con 5 elementos
```

3. Inicialización de Arreglos

Los arreglos se pueden inicializar en el momento de la declaración o después.

Inicialización en la Declaración

```
int valores[5] = {10, 20, 30, 40, 50}; // Inicializa el arreglo con valores específicos
```

Inicialización Posterior

```
int valores[5]; // Declara el arreglo
valores[0] = 10; // Asigna valores a los elementos
valores[1] = 20;
valores[2] = 30;
valores[3] = 40;
valores[4] = 50;
```

4. Acceso y Modificación de Elementos

Los elementos de un arreglo se acceden mediante su índice. La modificación de un elemento se realiza de la misma manera.

Ejemplo de Acceso y Modificación

```
void setup() {
  Serial.begin(9600); // Inicia la comunicación serial

  int valores[5] = {10, 20, 30, 40, 50}; // Declara e inicializa el arreglo

  Serial.println(valores[2]); // Imprime el tercer elemento (30)

  valores[2] = 100; // Modifica el tercer elemento
```

```
    Serial.println(valores[2]); // Imprime el nuevo valor (100)
}
```

5. Recorrido de Arreglos

Es común recorrer los elementos de un arreglo utilizando bucles. Un bucle `for` es una manera eficaz de hacerlo.

Ejemplo de Recorrido

```
void setup() {
    Serial.begin(9600); // Inicia la comunicación serial

    int valores[5] = {10, 20, 30, 40, 50}; // Declara e inicializa el arreglo

    for (int i = 0; i < 5; i++) {
        Serial.print("Elemento ");
        Serial.print(i);
        Serial.print(": ");
        Serial.println(valores[i]); // Imprime cada elemento del arreglo
    }
}
```

6. Arreglos Multidimensionales

Arduino también permite la creación de arreglos multidimensionales, como matrices. Estos son útiles para almacenar datos en dos o más dimensiones.

Ejemplo de Arreglo Bidimensional

```
void setup() {
    Serial.begin(9600); // Inicia la comunicación serial

    int matriz[2][3] = {
        {1, 2, 3},
        {4, 5, 6}
    }; // Declara e inicializa una matriz 2x3

    for (int i = 0; i < 2; i++) {
```

```
      for (int j = 0; j < 3; j++) {
        Serial.print("Elemento [");
        Serial.print(i);
        Serial.print("][");
        Serial.print(j);
        Serial.print("]: ");
        Serial.println(matriz[i][j]); // Imprime cada elemento de la matriz
      }
    }
}
```

7. Consideraciones al Usar Arreglos

- **Tamaño Fijo**: En Arduino, el tamaño de los arreglos debe ser conocido en tiempo de compilación, lo que significa que no se pueden redimensionar durante la ejecución.
- **Desbordamiento de Índice**: Es crucial no exceder los límites del arreglo, ya que esto puede causar comportamientos inesperados o fallos en el programa.
- **Tipos de Datos Homogéneos**: Todos los elementos de un arreglo deben ser del mismo tipo de dato.

8. Conclusiones

Los arreglos son herramientas esenciales y fundamentales en la programación de Arduino ya que nos permiten almacenar y manipular conjuntos de datos de manera eficiente. Con Esto en cuenta se puede declarar, inicializar y acceder a los elementos de los arreglos, los desarrolladores pueden crear proyectos más complejos y organizados. La capacidad de manejar arreglos es necesaria para el desarrollo de aplicaciones en el ámbito de los sistemas embebidos.

Bibliografía

1. Banzi, M., & Shiloh, M. (2014). *Getting Started with Arduino*. Maker Media, Inc.
    - Este libro proporciona una introducción a la plataforma Arduino, incluyendo la programación con arreglos.
2. Monk, S. (2015). *Programming Arduino: Getting Started with Sketches*. McGraw-Hill Education.
    - Un recurso que presenta ejemplos prácticos sobre la programación en Arduino, incluyendo el uso de arreglos.

3. Arduino. (n.d.). *Arduino Reference*. Retrieved from https://www.arduino.cc/reference/en/

    - Documentación oficial de Arduino que detalla las estructuras de datos, incluidos los arreglos.
4. Adafruit. (n.d.). *Learning System*. Retrieved from https://learn.adafruit.com/

    - Un recurso en línea que ofrece tutoriales sobre programación de Arduino y ejemplos prácticos.
5. Simon Monk. (2014). *Arduino Cookbook*. O'Reilly Media.

    - Un libro que ofrece una colección de recetas y ejemplos prácticos para trabajar con Arduino en diferentes proyectos.