Práctica 2
Informática 2
Facultad de Ingeniería
Universidad de Antioquia

Introducción

Esta práctica tiene como propósito fortalecer la comprensión de conceptos fundamentales del lenguaje C/C++, incluyendo el manejo de punteros, arreglos, funciones y representación en memoria. Además, se introduce el uso básico de Arduino y la plataforma Tinkercad como herramientas de simulación y programación para proyectos electrónicos sencillos. Las actividades propuestas incluyen ejercicios introductorios, problemas de aplicación y desafíos que requieren un pensamiento lógico más avanzado.

1 Objetivos

- Familiarizarse con la notación binaria y hexadecimal para representar números enteros.
- Introducir el concepto de dirección de memoria, arreglos y punteros en C/C++.
- Definir e implementar funciones propias.
- Crear y utilizar librerías.
- Introducir el uso de Arduino y Tinkercad.

2 Ejercicios

Los ejercicios que se presentan a continuación serán implementados por el docente. Con esto, se realizará una introducción al manejo de memoria en C++ y al uso de Tinkercad.

2.1 Manejo de punteros y representación en memoria.

1. Codifique el siguiente programa:

```
#include <iostream>

using namespace std;

void fun_a(int *px, int *py);
void fun_b(int a[], int tam);

int main() {
    int array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
    fun_b(array, 10);
}

void fun_a(int *px, int *py) {
    int tmp = *px;
```

```
    *px = *py;
    *py = tmp;
}

void fun_b(int a[], int tam) {
    int f, l;
    int *b = a;
    for (f = 0, l = tam -1; f < l; f++, l--) {
        fun_a(&b[f], &b[l]);
    }
}
```

Con ayuda del debugger, examine la representación en memoria del arreglo `array` y responda:

- ¿Cuál es su dirección en memoria? ¿Cuántos bytes se dedican para almacenar cada elemento de `array`?
- ¿Cuál es la dirección y el contenido en memoria del elemento `array[3]`?
- Describa el efecto que tiene la función `fun_b` sobre el arreglo `array`.

2. La siguiente función contiene errores. Corríjalos para que funcione correctamente:

```
void fun_c(double *a, int n, double *promedio, double *suma){
    int i;
    suma = 0.0;
    for (i = 0; i < n; i++) {
        suma += (a + i);
    }
    promedio = suma / n;
}
```

Implemente un caso de prueba para la función anterior que permita verificar su correcto funcionamiento.

3. Se tiene la siguiente declaración e inicialización:

```
unsigned short b[4][2] = {{77, 50}, {5, 2}, {28, 39}, {99, 3}};
```

Complete la numeración de las direcciones de memoria (en notación hexadecimal) para cada uno de los 8 elementos de `b`. Puede seleccionar un valor arbitrario para la primera posición de memoria del arreglo.

Determine el valor de las siguientes expresiones:

- `b`
- `b+2`
- `*(b+2)`
- `*(b+2)+1`
- `*(*(b+2)+1)`
- `b[3][1]`
- `*b++`

2.2 Tinkercad

1. Realice un circuito en Tinkercad donde se controle la intensidad de un LED con un potenciómetro. Utilice: Breadboard Small, Potenciometer, Resistor de 220Ω, LED y Arduino. Conecte correctamente y programe el Arduino. Considere el montaje de la Figura 1.

2. Realice un circuito en Tinkercad para mostrar mensajes en un LCD 16×2 usando Arduino. Cambie la opción de componentes de *Basic* a *All* en Tinkercad. Siga el esquemático de la Figura 2a y el montaje de la Figura 2b.

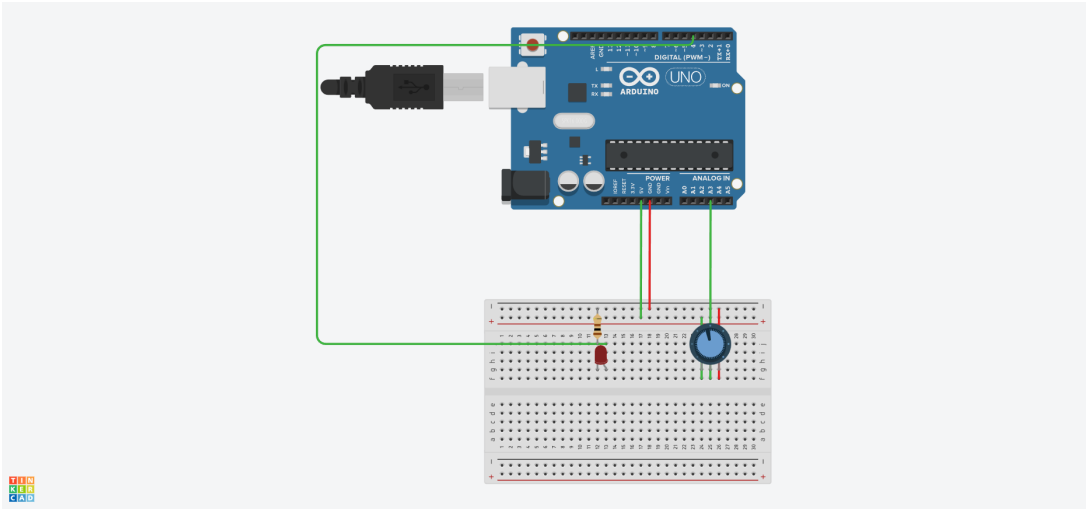


Figura 1: Circuito para encender un LED en Tinkercad.

3 Problemas

Los siguientes problemas están pensados para ser desarrollados utilizando arreglos y su directa relación con punteros. Varios de los problemas deben ser realizados en Tinkercad, por favor lea atentamente cada enunciado.

1. Desarrolle un programa que permita determinar la mínima combinación de billetes y monedas para una cantidad de dinero determinada. Los billetes en circulación son de $50.000, $20.000, $10.000, $5.000, $2.000 y $1.000, y las monedas son de $500, $200, $100 y $50. Si por medio de los billetes y monedas disponibles no se puede lograr la cantidad deseada, el sistema deberá decir lo que resta para lograrla. Use arreglos y ciclos para realizar el programa.

   Por ejemplo, Si se ingresa 47810, el programa debe imprimir:

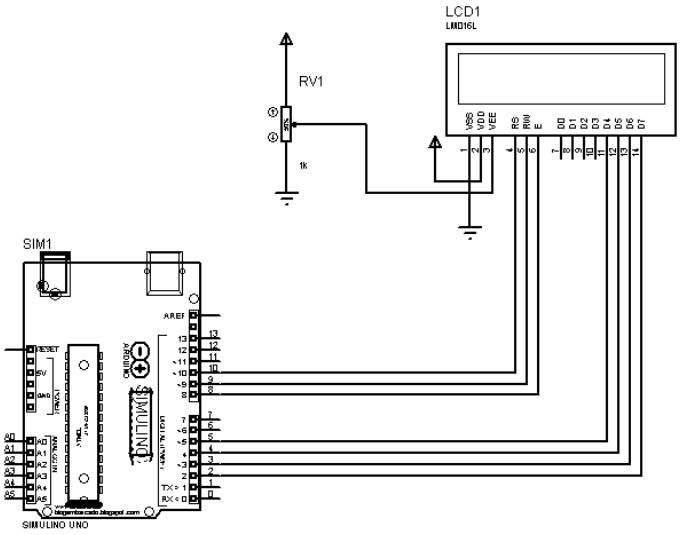
   ```
   50000:0
   20000:2
   10000:0
   5000:1
   2000:1
   1000:0
   500:1
   200:1
   100:1
   50:0
   Faltante: 10
   ```

   Realice una versión en Arduino de este programa (en un Arduino físico o Tinkercad), la cantidad de dinero debe ser ingresada con la ayuda del monitor serial.

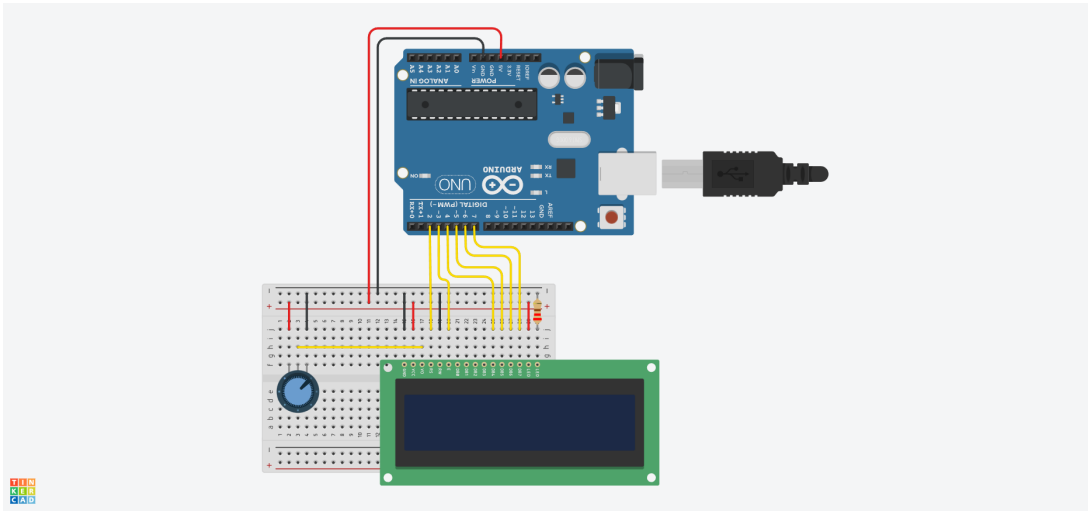
2. Elabore un programa que genere un arreglo de 200 letras mayúsculas aleatorias, lo muestre en consola y luego imprima cuántas veces se repite cada letra en el arreglo.

   Por ejemplo, frente al arreglo de 10 elementos: ABARSECAAB, el programa debe imprimir:

   ```
   ABARSECAAB
   A: 4
   B: 2
   C: 1
   E: 1
   S: 1
   ```



(a) Montaje del LCD 16 × 2.



(b) Circuito para utilizar LCD en Tinkercad.

Figura 2: Ejemplo de conexión de LCD 16 × 2 con Arduino Uno.

3. Esciba un programa que compare 2 cadenas de caracteres y retorne verdadero si son iguales y falso en caso contrario. Utilice arreglos de *char*. Tenga en cuenta la longitud del arreglo y el carácter que indica la finalización de una cadena de caracteres.

4. Escriba un programa que reciba una cadena de caracteres numéricos (arreglo de char), la convierta a un número entero y retorne dicho número.

   Por ejemplo, si recibe la cadena "123", debe retornar un `int` con valor 123.

5. Haga una función que reciba un número entero (`int`) y lo convierta a cadena de caracteres. Use parámetros por referencia para retornar la cadena. Escriba un programa de prueba que utilice dicha función.

   Por ejemplo, si el programa recibe un `int` con valor 123, la cadena que se retorne debe ser "123".

6. Escriba un programa que reciba una cadena de caracteres y cambie las letras minúsculas por mayúsculas; los demás caracteres no deben ser alterados.

   Por ejemplo, se recibe Man-zana debe mostrar MAN-ZANA. la salida del programa debe ser:

   ```
   Original: Man-zana. En mayúscula: MAN-ZANA.
   ```

Realice una versión en Arduino de este programa (en un Arduino físico o Tinkercad). Use el monitor serial de Arduino o Tinkercad para ingresar los valores necesarios e imprima el resultado utilizando el LCD.

7. Escriba un programa que reciba una cadena de caracteres y elimine los caracteres repetidos.

   Por ejemplo, si se recibe "bananas" debe mostrar "bans". La salida del programa debe ser:

   ```
   Original: bananas. Sin repetidos: bans.
   ```

8. Escriba un programa que reciba una cadena de caracteres y separe los números del resto de caracteres, generando una cadena que no tiene números y otra con los números que había en la cadena original.

   Por ejemplo, si se recibe abc54rst, el programa debe imprimir las cadenas: abcrst y 54. La salida del programa debe ser:

   ```
   Original: abc54rst.
   Texto: abcrst. Numero: 54.
   ```

9. Escribir un programa que reciba un número n y una cadena de caracteres numéricos; el programa debe separar la cadena de caracteres en grupos de n cifras, sumarlos e imprimir el resultado. En caso de no poder dividirse exactamente en grupos de n cifras, se debe rellenar con ceros a la izquierda del primer número.

   Por ejemplo, si $N = 3$ y se lee el arreglo 87512395, la suma sería 087+512+395+994. La salida del programa debe ser:

   ```
   Original: 87512395.
   Suma: 994.
   ```

10. Escribir un programa que permita convertir un número en el sistema romano al sistema arábigo usado actualmente. A continuación se encuentran los caracteres usados en el sistema romano y su equivalente arábigo:

    ```
    M: 1000
    D: 500
    C: 100
    L: 50
    X: 10
    V: 5
    I: 1
    ```

    Los números romanos se forman usando estos caracteres con base en 3 reglas:

    (a) Si un carácter está seguido por uno de igual o menor valor, su valor se suma al total.

    (b) Si un carácter está seguido por uno de mayor valor, su valor se resta del total.

    (c) No puede haber más de 3 caracteres repetidos seguidos.

    Por ejemplo, los números romanos y su equivalente: CC=200, CD=400, DC=600, DCLXVI=666, CLXXIV=174. La salida del programa debe ser:

    ```
    El número ingresado fue: DCLXVI
    Que corresponde a: 666.
    ```

Realice una versión en Arduino de este programa (en un Arduino físico o Tinkercad). Use el monitor serial de Arduino o Tinkercad para ingresar los valores necesarios e imprímalos usando el LCD.

11. Escriba un programa que permita manejar las reservas de asientos en una sala de cine. Los asientos de la sala de cine están organizados en 15 filas con 20 asientos cada una. El programa debe mostrar una representación de la sala que indique qué asientos están disponibles y cuáles se encuentran reservados. Además, debe permitir realizar reservas o cancelaciones al ingresar a la fila (letras A-O) y el número del asiento (números 1-20). Un ejemplo de visualización de una sección de la sala es el siguiente:

```
  1 2 3 4 5 6 7 8 9 ...
A + - - - - - - - -
B - - - - - - - - -
C - - - - - - - - +
D - - - - - - - - -
E - - - - - - - - -
F - - - - - - - - -
G - - - - - - - - -
H - - - - - - - - -
.
.
.
```

Donde "+" representa los asientos reservados y "-" representa los asientos disponibles.

12. Un cuadrado mágico es una matriz de números enteros no repetidos, en la que la suma de los números en cada columna, cada fila y cada diagonal principal tiene el mismo resultado. Escriba un programa que permita al usuario ingresar una matriz cuadrada, imprima la matriz y verifique si la matriz es un cuadrado mágico. Un ejemplo de cuadrado mágico es el siguiente:

$$\begin{pmatrix} 4 & 9 & 2 \\ 3 & 5 & 7 \\ 8 & 1 & 6 \end{pmatrix}$$

13. Se tiene una fotografía digitalizada de una porción de la galaxia NGC 1300 que está ubicada a 61.000.000 de años luz del planeta Tierra. La representación digital de la imagen está constituida por una matriz de números enteros; en la cual, cada uno representa la cantidad de luz en ese punto de la imagen.

$$\begin{pmatrix} 0 & 3 & 4 & 0 & 0 & 0 & 6 & 8 \\ 5 & 13 & 6 & 0 & 0 & 0 & 2 & 3 \\ 2 & 6 & 2 & 7 & 3 & 0 & 10 & 0 \\ 0 & 0 & 4 & 15 & 4 & 1 & 6 & 0 \\ 0 & 0 & 7 & 12 & 6 & 9 & 10 & 4 \\ 5 & 0 & 6 & 10 & 6 & 4 & 8 & 0 \end{pmatrix}$$

Se puede determinar si el elemento $a_{i,j}$ de la matriz representa una estrella si se cumple que:

$$\frac{a_{i,j} + a_{i,j-1} + a_{i,j+1} + a_{i-1,j} + a_{i+1,j}}{5} > 6 \tag{1}$$

Elabore y pruebe una función que reciba un puntero a la matriz de enteros como argumento y que retorne el número de estrellas encontradas en la imagen. Ignore las estrellas que puedan existir en los bordes de la matriz.

14. Elabore un programa que llene una matriz 5×5 con los números del 1 al 25 y la imprima, luego imprima la matriz rotada 90, 180 y 270 grados. Por ejemplo, las matrices original y rotada 90 grados:

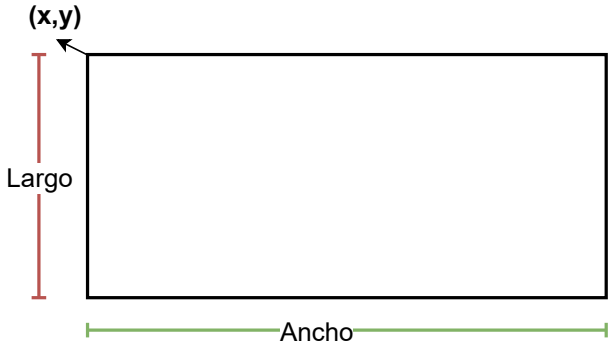
$$M = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix}, \qquad M_{90^\circ} = \begin{pmatrix} 21 & 16 & 11 & 6 & 1 \\ 22 & 17 & 12 & 7 & 2 \\ 23 & 18 & 13 & 8 & 3 \\ 24 & 19 & 14 & 9 & 4 \\ 25 & 20 & 15 & 10 & 5 \end{pmatrix}$$

15. Elabore un programa que permita hallar la intersección entre un par de rectángulos. Represente los rectángulos como arreglos de 4 datos de la siguiente manera:

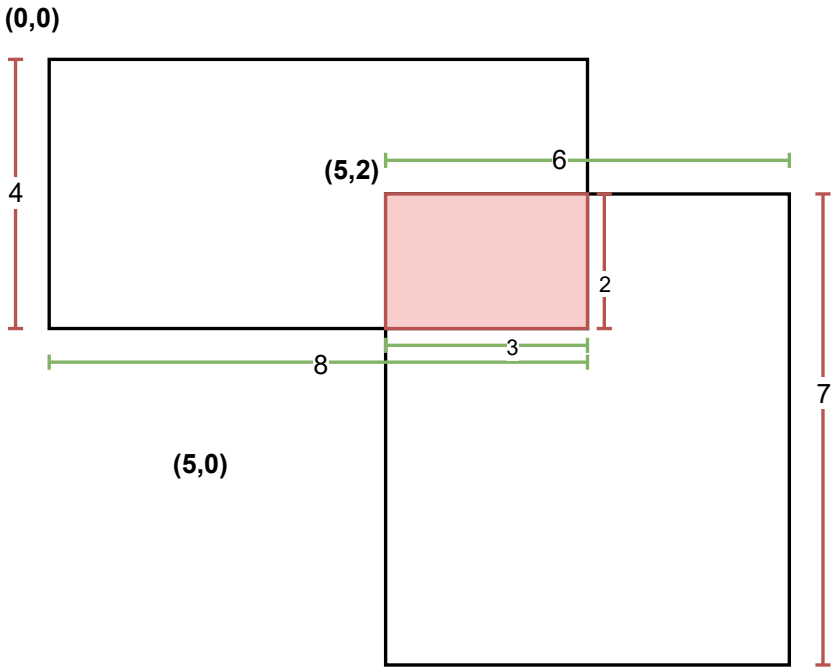
    (a) Los primeros 2 datos corresponden a las coordenadas de la esquina superior izquierda del rectángulo (x,y).

    (b) Los siguientes 2 datos representan el ancho y la altura del rectángulo.

    Esto se representa en la Figura 3a. Tenga en cuenta que los valores en el eje vertical aumentan hacia abajo y en el eje horizontal aumentan hacia la derecha.



(a) Rectángulo y coordenadas.



(b) Intersección entre rectángulos.

Figura 3: Representación gráfica del ejercicio 15.

El programa debe recibir 2 arreglos que representen los rectángulos A y B. Por referencia, debe retornar un rectángulo C (con la misma estructura descrita anteriormente) que corresponda a la intersección entre A y B.

Por ejemplo, si se ingresan los rectángulos A y B representados por los arreglos {0, 0, 8, 4} y {5, 2, 6, 7}, el rectángulo C debe ser el arreglo {5, 0, 3, 4}, como se puede ver en la Figura 3b.

16. En una malla de 2×2, realizando únicamente movimientos hacia la derecha y hacia abajo, hay 6 posibles caminos para llegar de la esquina superior izquierda a la inferior derecha, como se observa en la Figura 4.

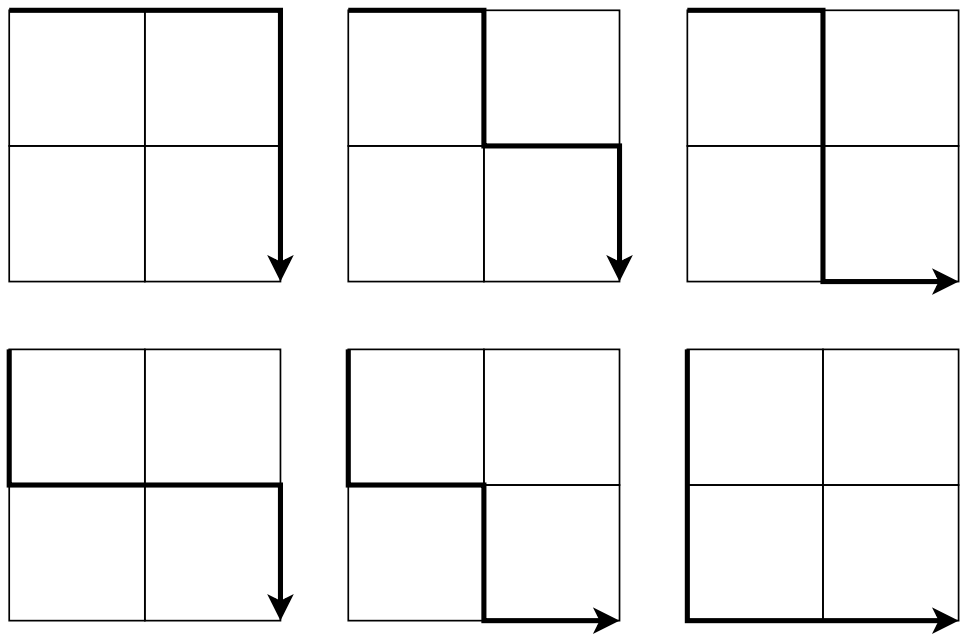


Figura 4: Caminos posibles en una malla 2×2.

Escriba un programa que reciba un número N y calcule el número de caminos posibles en una cuadrícula de $N \times N$. La salida del programa debe ser:

```
Para una malla de 2x2 puntos hay 6 caminos.
```

17. Dos números a y b (con $a \neq b$) son amigables si la suma de los divisores de **a** (excluyéndose a sí mismo) es igual a **b**, y al sumar los divisores de **b** el resultado es **a**. Ej.: los divisores de 220 son 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110; y suman 284. Los divisores de 284 son 1, 2, 4, 71 y 142, y suman 220. Entonces, 220 y 284 son amigables. Escriba un programa que reciba un número y halle la suma de todos los números amigables menores al número ingresado. La salida del programa cuando se ingresa el número 300 debe ser:

```
El resultado de la suma es: 504.
```

Realice una versión en Arduino de este programa (en un Arduino físico o Tinkercad). Los datos deben ser ingresados con la ayuda del serial. Use el monitor serial de Tinkercad para ingresar los valores necesarios e imprima usando el LCD.

18. Las permutaciones lexicográficas son permutaciones ordenadas numéricamente o alfabéticamente. Por ejemplo, las permutaciones lexicográficas de 0, 1 y 2 son: 012, 021, 102, 120, 201, 210. Escriba un programa que reciba un número n y halle la enésima permutación lexicográfica de los números entre 0 y 9. Por ejemplo, para $N = 100000$, la permutación lexicográfica es 2783915460. La salida del programa debe ser:

```
La permutación número 1000000 es: 2783915460.
```

19. **Problema de bonificación** Realice un programa con un Arduino físico para simular ciertos botones del teclado del computador. Cuando se presione un pulsador físico, se debe escribir en pantalla alguna letra. Ponga al menos 4 pulsadores para las letras 'W', 'A', 'S' y 'D'.

    Nota: Considere el uso de la librería `Keyboard.h` de Arduino.