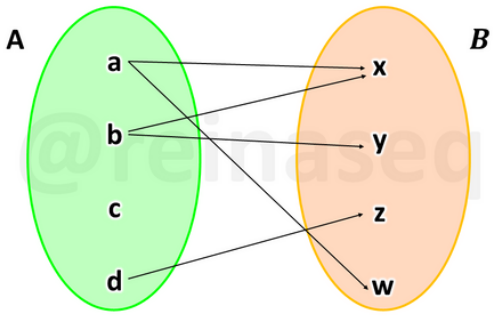
# Relaciones binarias

En la realidad humana es posible percibir relaciones entre todo lo que existe de acuerdo a cualidades, propiedades, condiciones, entre otros, que permiten al menos vincular dos situaciones, personas, animales, cosas; es típico en nuestra expresión verbal utilizar frases como

"es amigo de", "es jefe de", "vive en", "obtuvo la calificación", "su salario es", "se desempeña en", "su profesión es", "se alimenta con", "es novia de",

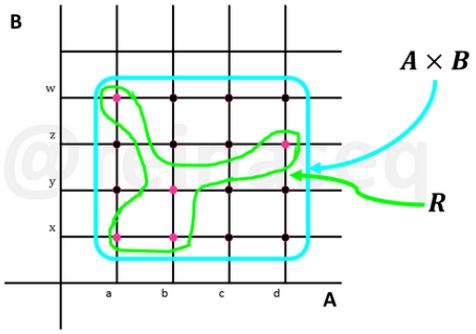
Desde la perspectiva gráfica, las Relaciones Binarias pueden ser representadas de tres formas posibles: Diagrama de Venn (ya conocida en anteriores publicaciones), Gráfico Cartesiano y Matriz. A continuación se explica y muestra cada uno:

- Diagrama de Venn: En esta representación generalmente se dibujan dos formas ovaladas una al lado de la otra las cuales representan la de la izquierda el primer conjunto o de partida y la de la derecha el segundo conjunto o de llegada, en cada uno en su interior se escriben los elementos que pertenecen a los mismos en detalle, adicionalmente, del primero se acostumbran a dibujar flechas que salen de los elementos de éste hacia los del otro conjunto siempre y cuando los mismos estén vinculados o relacionados. Lo dicho puede visualizarse en la siguiente imagen en la cual se ilustra lo referido al ejemplo dado anteriormente




- Gráfico Cartesiano: Esta representación es muy conocida en el ámbito de la Matemática, la misma consiste en la intersección en el plano de dos ejes mutuamente perpendiculares (es decir, entre sí forman ángulos de 90 grados), de los cuales el eje horizontal se le denomina eje de las abscisas y en él se ubican los elementos del primer conjunto y el vertical se conoce como eje de las ordenadas en

el cual se sitúan los elementos del segundo conjunto. Posteriormente, se trazan paralelas de cada uno de los elementos representados en los ejes lo cual forma cuadrícula, de la misma los diferentes vértices que unen los elementos de los conjuntos dados son los pares ordenados que forman parte del Producto Cartesiano entre los mismos. De este, se destacan de alguna forma particular los elementos o pares que integran la relación R. La representación sería como se muestra a continuación




- Matriz: En esta forma, se acostumbra a dibujar una tabla de doble entrada ubicando en la primera columna de la izquierda los elementos del primer conjunto y en la fila superior los elementos que pertenecen al segundo conjunto. En el primer recuadro ubicado en la esquina superior izquierda se escribe la relación o propiedad que permitirá vincular los elementos de los conjuntos dados. A continuación se procederá a llenar los recuadros interceptores entre elementos de la siguiente manera: se escribirá cero “0” si no están relacionados y uno “1” si en efecto hay relación entre los mismos. A continuación se muestra la representación de esta forma gráfica

| $R$ | x | y | z | w |
|---|---|---|---|---|
| a | 1 | 0 | 0 | 1 |
| b | 1 | 1 | 0 | 0 |
| c | 0 | 0 | 0 | 0 |
| d | 0 | 0 | 1 | 0 |

Implementación en C de una relación binaria

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

//        Estructura de la relación binaria.
//  Una matriz de valores booleanos para marcar verdadero en donde exista la relación
//  binaria
typedef struct {
        bool **relacion;
        int cantidad,ancho,alto;
} RelacionBinaria;

// Estructura de una imagen a color en el formato PPM
typedef struct {
        int ancho, alto;
        int **rojo, **verde, **azul; // tres matrices para almacenar el color en formato RGB
}ImagenPPM;

// Construir el espacio en memoria para almacenar la imagen PPM almacenada en el
// archivo cuyo nombre se encuentra almacenado en la variable nombreArchivo
void inicializarImagen(ImagenPPM *I,char *nombreArchivo){
        FILE *archivo = fopen(nombreArchivo,"r"); // Abrir el archivo para lectura
        int ancho, alto, escala, tipo;char letra;
        fscanf(archivo,"%c%d %d %d %d",&letra,&tipo,&ancho, &alto, &escala);
        I->ancho=ancho;I->alto=alto;
        I->rojo = (int **)malloc(alto * sizeof(int*));
        I->verde = (int **)malloc(alto * sizeof(int*));
        I->azul = (int **)malloc(alto * sizeof(int*));
        for(int y=0;y<alto;y++){
                I->rojo[y] = (int *)calloc(ancho,sizeof(int));
                I->verde[y] = (int *)calloc(ancho,sizeof(int));
                I->azul[y] = (int *)calloc(ancho,sizeof(int));
                for(int x=0;x<ancho;x++){
                fscanf(archivo,"%d %d %d",&I->rojo[y][x],&I->verde[y][x],&I->azul[y][x]);
                }
        }
        fclose(archivo);
}
void ImprimirImagenPPM(ImagenPPM *I){
        printf("P3\n%d %d\n255\n",I->ancho,I->alto);
        for(int y=0;y<I->alto;y++){
                for(int x=0;x<I->ancho;x++){
                printf("%d %d %d ",I->rojo[y][x],I->verde[y][x],I->azul[y][x]);
                }
                printf("\n");
        }

}
```

```
// Inicializar la relación binaria, todos los espacios de la matriz deben contener el valor false
void inicializarRelacion(RelacionBinaria *r, int ancho, int alto) {
        r->cantidad = 0;r->alto=alto;r->ancho = ancho;
        r->relacion = (bool **)calloc(r->alto,sizeof(bool *)); // Memoria dinámica para una matriz
        for(int i = 0; i < r->alto; i++) {
            r->relacion[i] = (bool *)calloc(r->ancho,sizeof(bool)); // Memoria para un renglón
            for(int j = 0; j < ancho; j++) {
                r->relacion[i][j] = false;
            }
        }
}
// La relación binaria se refiere a las coincidencia con el color RGB y las variables de
// entrada: rojo, verde y azul. Se marca con true la posición o píxel en la matriz existente en
// la variable tipo relaciónBinaria que coincide con el color que se recibe como argumento en
// la función y los colores en la variable del tipo ImagenPPM.
void construirRelacionCoincidencia(RelacionBinaria *r,ImagenPPM *I, int rojo, int verde, int
azul) {
        for(int i = 0; i < I->alto; i++) {
          for(int j = 0; j < I->ancho; j++) {
              if(I->rojo[i][j] == rojo && I->verde[i][j] == verde && I->azul[i][j] == azul) {
                  r->relacion[i][j] = true;
                  r->cantidad++;
              }
          }
        }
}
// Se imprime la relación binaria como una imagen PPM en el formato binario
void imprimirRelacion(const RelacionBinaria *r) {
        printf("P1\n%d %d\n",r->ancho,r->alto);

        for(int i = 0; i < r->alto; i++) {
        for(int j = 0; j < r->ancho; j++) {
                printf("%d ", r->relacion[i][j]);
        }
        printf("\n");
        }
}

// Ejemplo de uso de las estructuras y sus propiedades.
int main(int argc, char *argv[]) {
        ImagenPPM IMG;
        inicializarImagen(&IMG,argv[1]);
        RelacionBinaria r;

        inicializarRelacion(&r, IMG.ancho,IMG.alto);
        construirRelacionCoincidencia(&r, &IMG, 253,152,39);
        imprimirRelacion(&r);
```

```
        return 0;
}
```

Forma de ejecución del programa:
./a.out archivo.ppm > resultadoRelacion.ppm; eog resultadoRelacion.ppm


Imagen de entrada en formato ppm

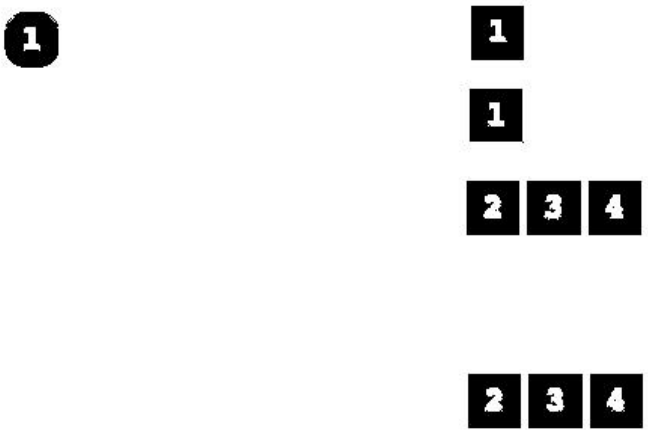



Imagen resultante, proporcionando el color RGB: 253,152,39

**Actividades:**

1. **Determinar la cantidad de colores diferentes en la imagen de entrada.**
2. **Obtener para cada color diferente una imagen resultante que determina la relación binaria: El color R, G, B existe en la imagen de entrada.**