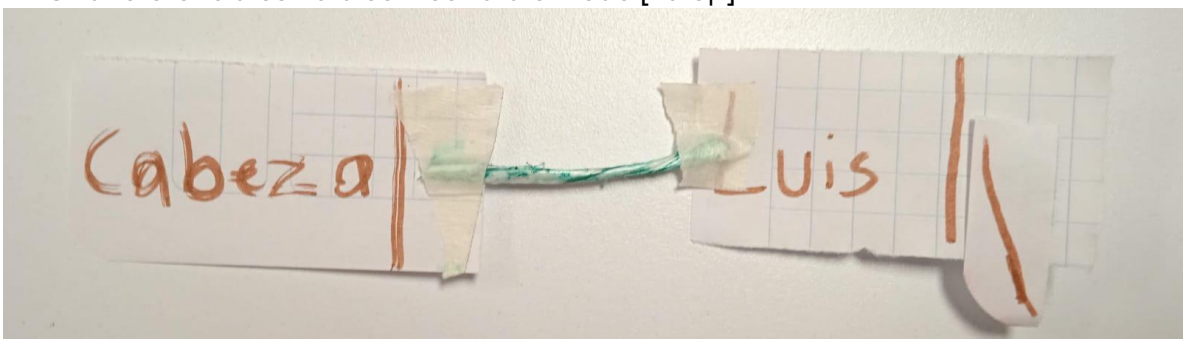
Práctica Manual y Algorítmica: Listas Enlazadas (Simples, Dobles y Circulares)

Alumno: Jesús Talat Otero Hernández

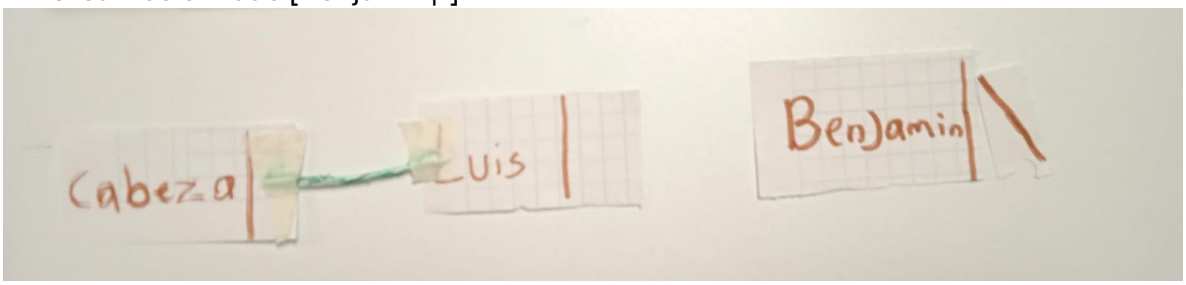
12 tarjetas o fichas de colores (mínimo 4 por lista).
• Hilos o cordones (rojo y azul).

¿¿Como agregó a Luis a mi lista??

1.- Creamos el nodo [Luis|\].
2.- Accedemos a cabeza.
3.- Validamos si la referencia de la cabeza tiene un enlace nulo.
4.- Si la referencia es nula se insertará el nodo [Luis|\].



Ahora inserte un nuevo nodo [Benjamin|\]
1.- Creamos el nodo [Benjamin|\]



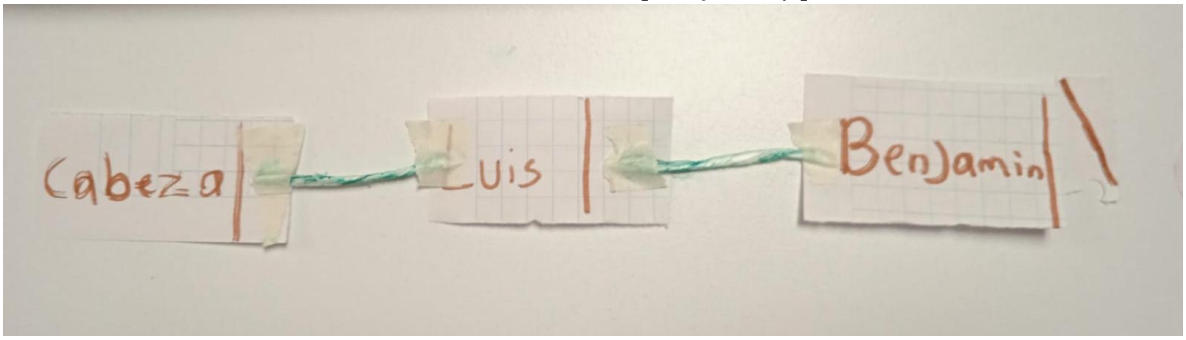
2.- Accedemos a la referencia de la cabeza para verificar no es nula.

```
        [Cabeza|-]-->[Luis|\]
             ^
    [temp|-]-|
        temp = cabeza;
```

3.- Accedemos a la de Luis y verificamos si es nula.

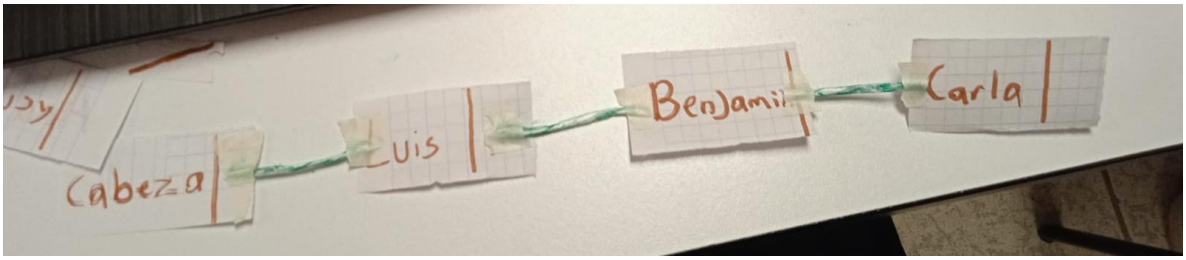
```
        [Cabeza|-]-->[Luis|\]
                          ^
                  [temp|-]-|
```

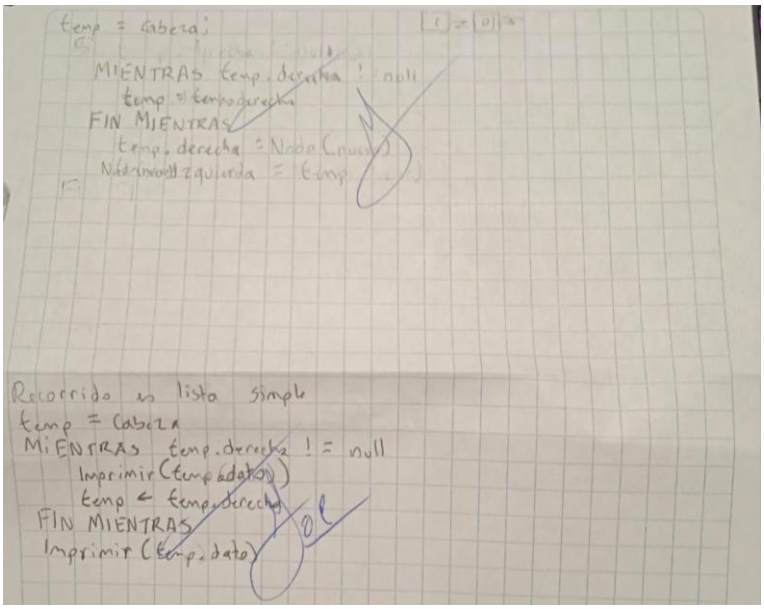
4.- Si la referencia de Luis es nula, insertamos [Benjamin|\]



PseudoCodigo para realizar los recorridos y poder insertar a Carla

```
temp = cabeza;
MIENTRAS temp.derecha != nulo
        temp = temp.derecha;
FIN MIENTRAS
//Tomamos la referencia del último nodo con null y le indicamos que el puntero apuntara a un nuevo nodo.
temp.derecha = Nodo(carla)

temp = cabeza;
while(temp.derecha != null){
        temp = temp.derecha;
}
temp.derecha = carla;
```

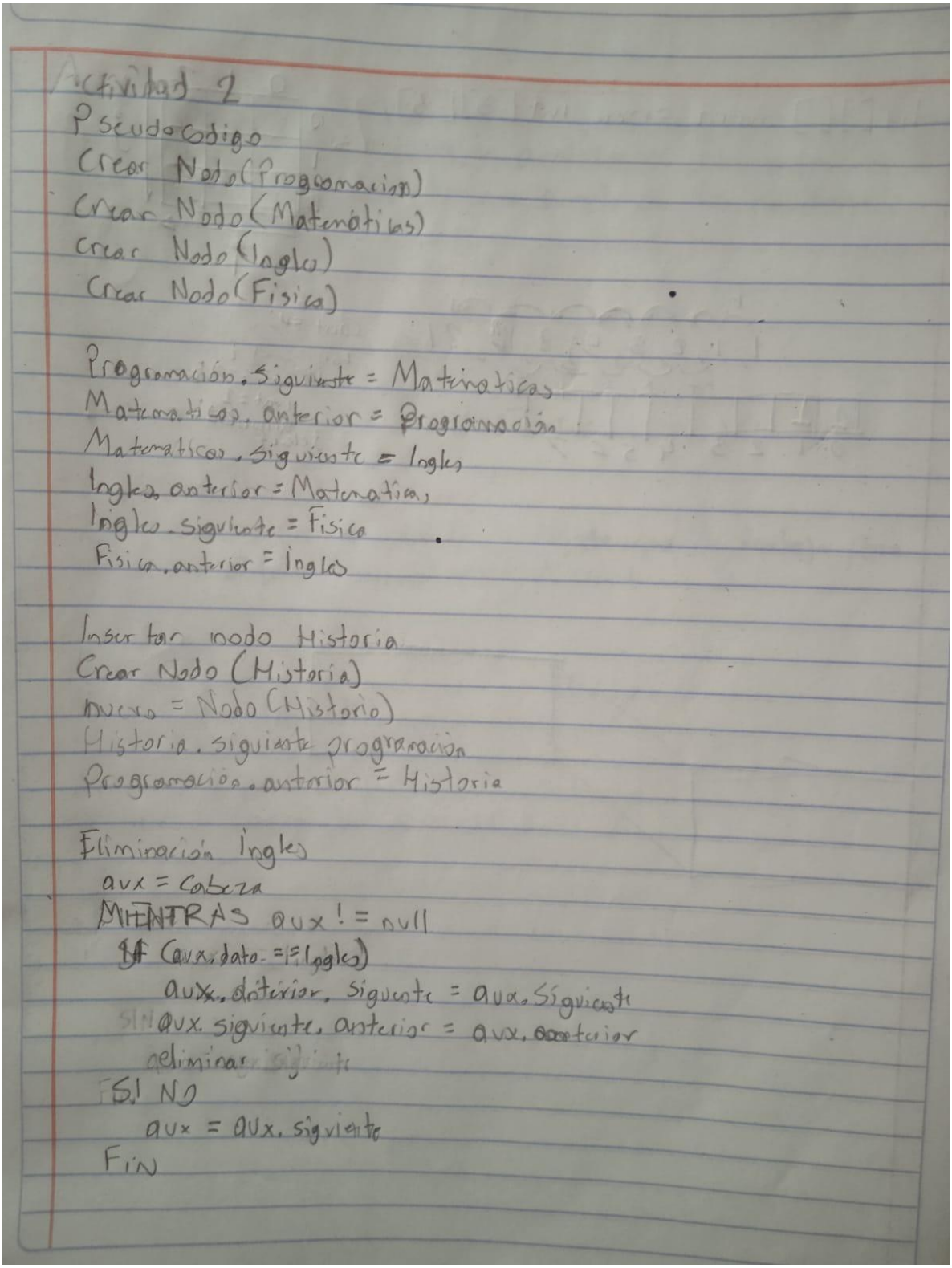




Recorrido en lista simple

```
temp = Cabeza
MIENTRAS temp.derecha != null
    Imprimir(temp.dato)
    temp ← temp.derecha
FIN MIENTRAS
Imprimir(temp.dato)
```

Actividad 2:



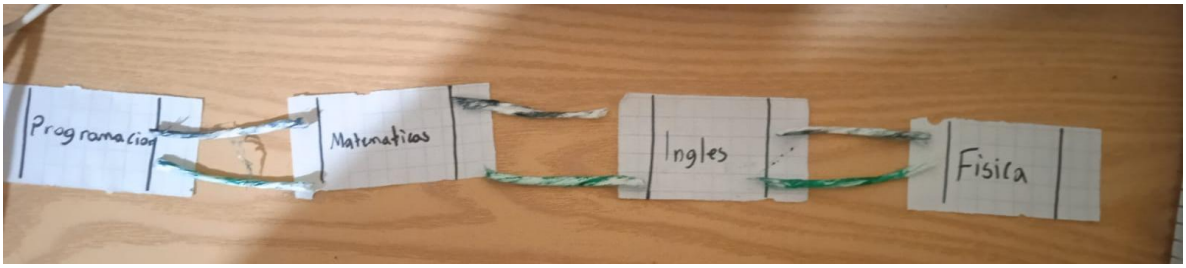
Actividad 2
Pseudocodigo
Crear Nodo(Programacion)
Crear Nodo(Matematicas)
Crear Nodo(Ingles)
Crear Nodo(Fisica)

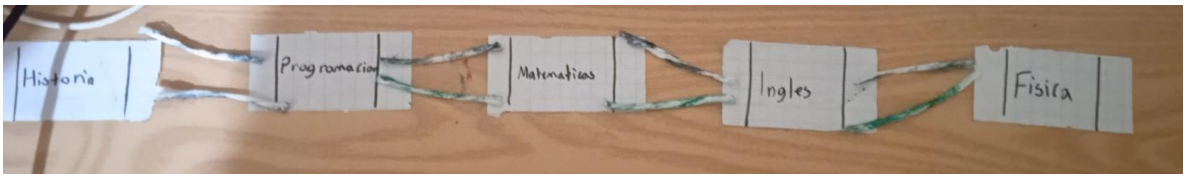
Programacion.siguiente = Matematicas
Matematicas.anterior = Programacion
Matematicas.siguiente = Ingles
Ingles.anterior = Matematicas
Ingles.siguiente = Fisica
Fisica.anterior = Ingles

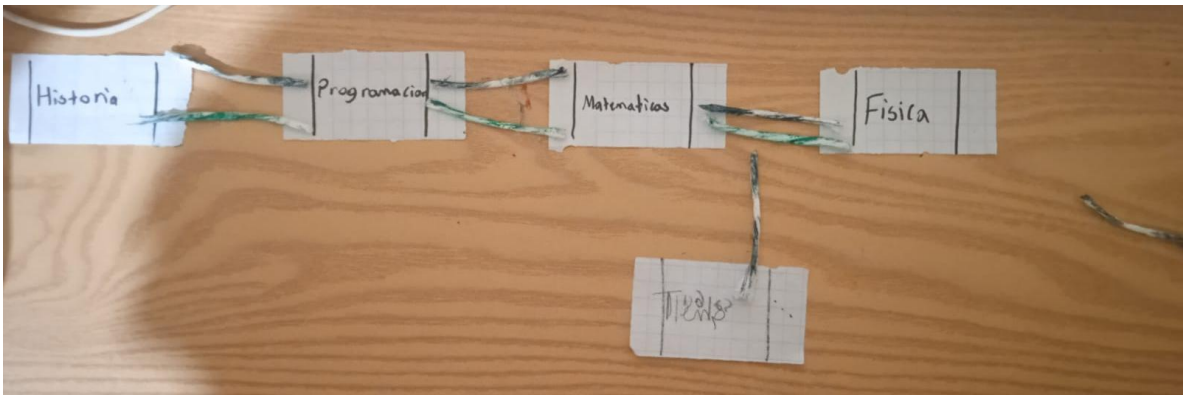
Insertar nodo Historia
Crear Nodo (Historia)
nuevo = Nodo (Historia)
Historia.siguiente programacion
Programacion.anterior = Historia

Eliminacion Ingles
aux = Cabeza
MIENTRAS aux != null
If (aux.dato == Ingles)
aux.anterior.siguiente = aux.siguiente
aux.siguiente.anterior = aux.anterior
eliminar
SI NO
aux = aux.siguiente
FIN

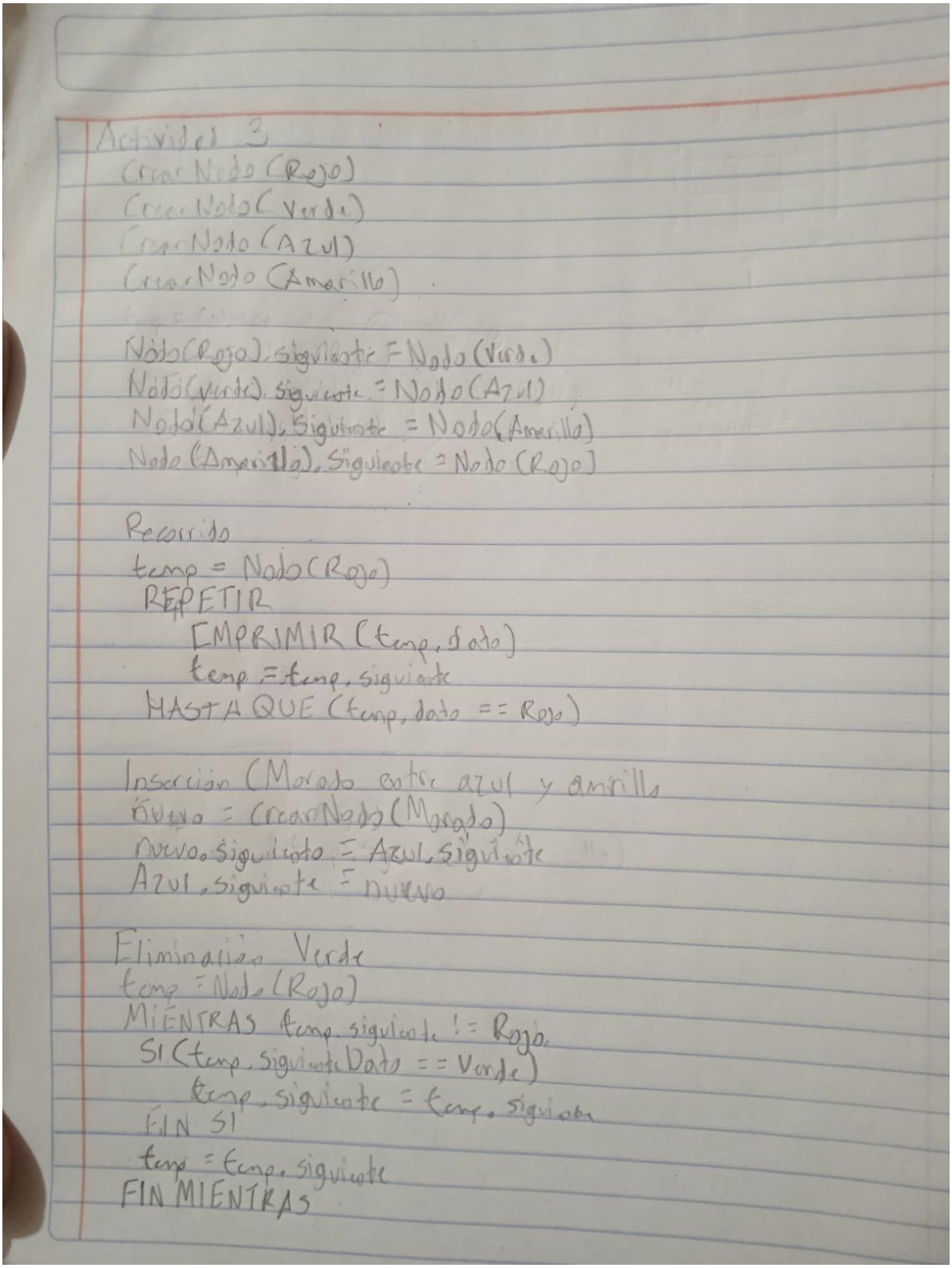
```
Nodo programacion = new Nodo("Programación");
      Nodo matematicas = new Nodo("Matemáticas");
      Nodo ingles = new Nodo("Inglés");
      Nodo fisica = new Nodo("Física");
// Insertar nodo Historia antes de Programación
      Nodo historia = new Nodo("Historia");
      historia.siguiente = programacion;
      programacion.anterior = historia;

      // Nueva cabeza
      Nodo cabeza = historia;

      // Eliminar nodo Inglés
      Nodo aux = cabeza;
      while (aux != null) {
        if (aux.dato.equals("Inglés")) {
          if (aux.anterior != null)
            aux.anterior.siguiente = aux.siguiente;
          if (aux.siguiente != null)
            aux.siguiente.anterior = aux.anterior;
          break;
        }
        aux = aux.siguiente;
      }
```






Historia
Programacion
Matematicas
Fisica

Actividad 3



Actividad 3
CrearNodo(Rojo)
CrearNodo(Verde)
CrearNodo(Azul)
CrearNodo(Amarillo)

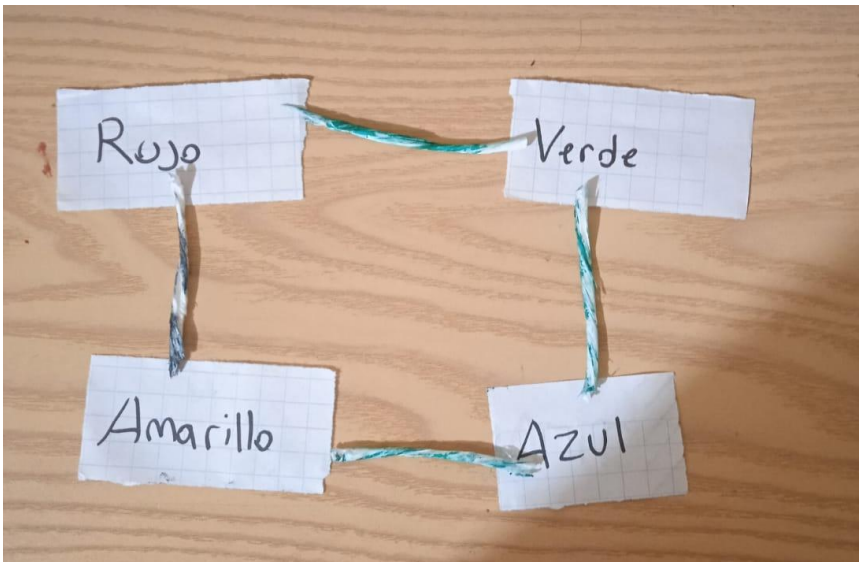
Nodo(Rojo).siguiente = Nodo(Verde)
Nodo(Verde).siguiente = Nodo(Azul)
Nodo(Azul).Siguiente = Nodo(Amarillo)
Nodo(Amarillo).Siguiente = Nodo(Rojo)

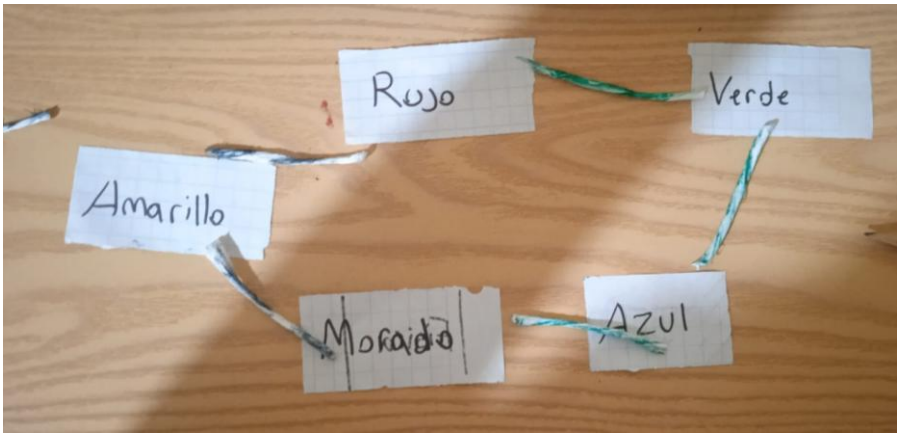
Recorrido
temp = Nodo(Rojo)
REPETIR
    IMPRIMIR(temp.dato)
    temp = temp.siguiente
HASTA QUE(temp.dato == Rojo)

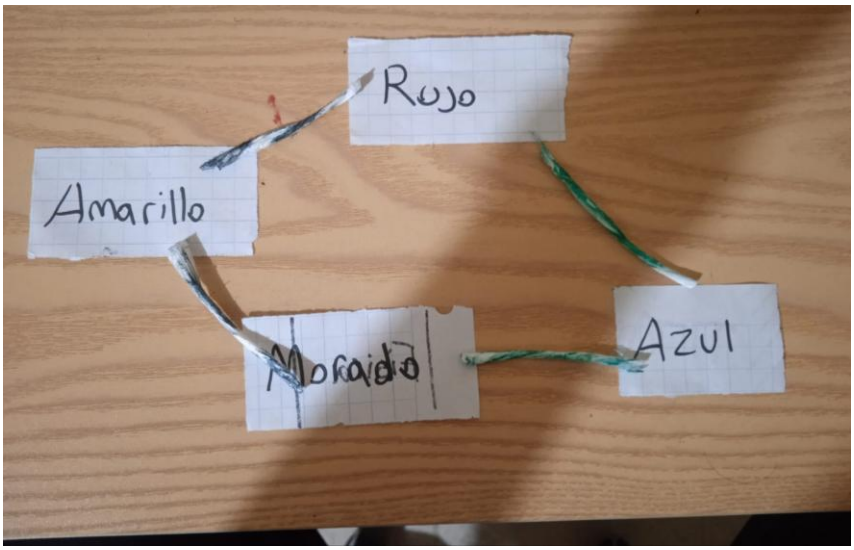
Insercion(Morado entre azul y amrillo
Nuevo = CrearNodo(Morado)
Nuevo.siguiente = Azul.siguiente
Azul.siguiente = Nuevo

Eliminacion Verde
temp = Nodo(Rojo)
MIENTRAS temp.siguiente != Rojo
SI(temp.siguiente.Dato == Verde)
    temp.siguiente = temp.siguiente
FIN SI
temp = temp.siguiente
FIN MIENTRAS

```
// Crear nodos
      Nodo rojo = new Nodo("Rojo");
      Nodo verde = new Nodo("Verde");
      Nodo azul = new Nodo ("Azul");
      Nodo amarillo = new Nodo("Amarillo");

      // Enlazar en forma circular
      rojo.siguiente = verde;
      verde.siguiente = azul;
      azul.siguiente = amarillo;
      amarillo.siguiente = rojo;

      // Recorrido
      Nodo temp = rojo;
      System.out.println("Recorrido inicial:");
      do {
         System.out.println(temp.dato);
         temp = temp.siguiente;
      } while (temp != rojo);

      // Insertar Morado entre Azul y Amarillo
      Nodo morado = new Nodo ("Morado");
      morado.siguiente = azul.siguiente;
      azul.siguiente = morado;

      // Eliminar Verde
      temp = rojo;
      while (temp.siguiente != rojo) {
         if (temp.siguiente.dato.equals("Verde")) {
            temp.siguiente = temp.siguiente.siguiente;
            break;
         }
         temp = temp.siguiente;
      }
```


Rojo
Verde
Amarillo
Azul


Rojo
Verde
Amarillo
Morado
Azul


Rojo
Amarillo
Morado
Azul

Reflexión final (en equipo o individual)

1. Describe con tus propias palabras los conceptos de lista simple, doble y circular.
   La lista simple solamente conoce al que le sigue, el doble conoce tanto al anterior como al que sigue y el circulo no tiene como tal un final, esa hay que definirla nosotros
2. ¿Qué tipo de lista es más eficiente para insertar y eliminar en cualquier posición?
   La simple.
3. ¿Qué ventaja tiene la lista circular frente a la simple?
   Guarda la información de una manera mucha más dinámica.
4. ¿Qué sucede si se rompe un enlace en una lista doble?
   Se genera un error ya que un nodo no conocera uno de los datos.
5. ¿Cómo se representa el “NULL” en una lista circular?
   Una forma seria de que el ciclo de la lista termine cuando lleguemos al nodo que es nuestra cabeza.