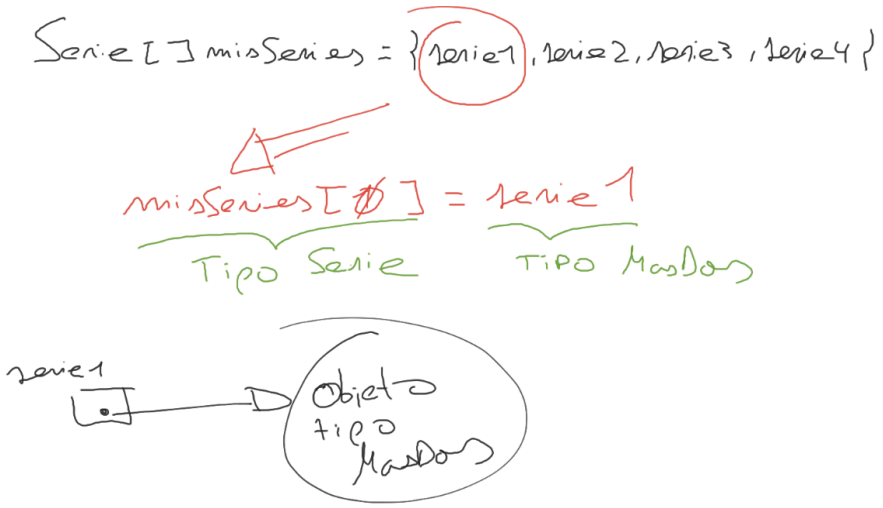
Ejercicio U5_B6C_E1:

```
class App{
    public static void main(String[] args) {
        Serie  serie1 = new MasDos();
        Serie serie2= new MasDos();
        Serie serie3 = new MasTres();
        Serie serie4= new MasTres();
        serie2.establecerInicio(200);
        serie4.establecerInicio(300);
        Serie[] misSeries={serie1,serie2,serie3,serie4};
        for(int i=0;i<misSeries.length;i++){
            System.out.print("\nSerie"+i+": ");//por esta impresión no uso for mejorado
            for(int j=0;j<5;j++)
                System.out.print(misSeries[i].obtenerSiguiente()+" ");
        }
    }
}
```

La combinación de diferentes técnicas de programación: polimorfismo +array + bucle evita la generación de código duplicado



Ejercicio U5_B6C_E2:

Para simplificar todas las clases en el mismo paquete.

```
abstract class Figura{
    String color;
    Figura(String color){
        this.color=color;
    }
    abstract double area();
}

class Triangulo extends Figura{

   double base;
   double altura;

    public Triangulo(double base, double altura, String color) {
        super(color);
        this.base = base;
        this.altura = altura;
    }

    @Override
    double area() {
```

```
      return base*altura/2;
   }

}
class Rectangulo extends Figura{
   double largo;
   double ancho;

   public Rectangulo(double largo, double ancho, String color) {
      super(color);
      this.largo = largo;
      this.ancho = ancho;
   }

   @Override
   double area() {
      return largo*ancho;
   }

}

class Circulo extends Figura{
   double radio;

   public Circulo(double radio, String color) {
      super(color);
      this.radio = radio;
   }

   @Override
   double area() {
      return radio*radio*Math.PI;
   }

}
class App{
   public static void main(String[] args) {

      Figura[] listaFiguras={new Circulo(1.0,"verde"),
         new Circulo(2.0,"rojo"),
         new Circulo(3.0,"Amarillo"),
         new Triangulo(1.0,1.0,"verde"),
         new Triangulo(2.0,2.0,"rojo"),
         new Rectangulo(3.0,4.0,"rosa"),
         new Rectangulo(2.0,3.0,"lila")};

      for(Figura f:listaFiguras){
         if(f.area()>4.0){
            f.color="negro";
         }
         System.out.println(f.area()+", "+f.color);
      }
  }
}
```

Observa que como al trabajar en el mismo paquete App tiene acceso a color ya que es protected.

Ejercicio U5_B6C_E3: añadir contar triángulos. Modificamos el for anterior como sigue

```
      int numeroTriangulos=0;
       for(Figura f:listaFiguras){
          if(f.area()>4.0){
             f.color="negro";
          }
```

```
        System.out.println(f.area()+", "+f.color);
        if (f instanceof Triangulo){
            numeroTriangulos++;
        }
    }
    System.out.println("Número de triángulos en la lista: "+ numeroTriangulos);
```

Ejercicio U5_B6C_E4:resuelve el último ejercicio con getClass() en lugar de instance of

```
int numeroTriangulos=0;
    for(Figura f:listaFiguras){
        if(f.area()>4.0){
            f.color="negro";
        }
        System.out.println(f.area()+", "+f.color);
        if (f.getClass().getName().equals("Triangulo")){
            numeroTriangulos++;
        }
    }
```