- **Máximo común divisor**:

```
public static int calcular_mcd_recursiva(int n1, int n2){
   if(n2 == 0){
      return n1;
   }else{
      return calcular_mcd_recursiva(n2, n1 % n2);
   }
}
```

- **Ordenar array de enteros**

```
public void ordenarArrayEnteros_recursiva(int[] ent, int empieza){
   int pequeño, temp, posPeque;
   pequeño = ent[empieza];
   posPeque = empieza;

   for(int i = empieza; i<10; i++){
      if(pequeño > ent[i]){
         temp = pequeño;
         pequeño = ent[i];
         ent[i] = temp;
         ent[posPeque] = pequeño;
      }
   }
   if(empieza+1 < ent.length){
      ordenarArrayEnteros_recursiva(ent, empieza+1);
   }
}
```

- **Suma de N primeros números enteros**

```
public static int calcular_suma_N_enteros_recursiva(int n1){
   if(n1 == 1){
      return 1;
   }else{
      return n1 + calcular_suma_N_enteros_recursiva(n1-1);
   }
}
```

- **Suma de N primeros pares**

```
public static int calcular_suma_N_pares_recursiva(int n1){
   if(n1 == 2){
      return 2;
   }else{
      return n1 + calcular_suma_N_pares_recursiva(n1-2);
   }
}
```

- **Decimal a binario**

```
public static void entero_positivo_a_binario_rercursivo(int n1){
   int div, resto;
   if(n1 < 2){
      System.out.print(n1);
   }else{
      entero_positivo_a_binario_rercursivo(n1 / 2);
      System.out.print(n1 % 2);

   }
}
```

- **Binario a entero**

```
public static int binario_a_entero_rercursivo(String n1){
   String a;
   if(n1.length() == 1){
      return Integer.parseInt(n1);
   }else{
      return (int) (Math.pow(2, n1.length() - 1) * Integer.parseInt("" +
   n1.charAt(0))) + binario_a_entero_rercursivo(n1.substring(1,
   n1.length()));
   }
}
```

- **Invertir array**

```
public void invertirArrayEnteros_recursiva(int[] ent, int empieza){
      int temp;
      if(empieza < ent.length / 2){
       temp = ent[empieza];
       ent[empieza] = ent[ent.length -1 - empieza];
       ent[ent.length -1 - empieza] = temp;
       invertirArrayEnteros_recursiva(ent, empieza + 1);
      }else{
       if(empieza % 2 != 0){
              temp = ent[empieza];
              ent[empieza] = ent[ent.length - 1 - empieza];
              ent[ent.length - 1 - empieza] = temp;
       }
      }

}
```

- **Detectar palíndromo**

```
public boolean DetectarPalindromo_recursiva(String pa, int empieza){

      if(empieza < pa.length() / 2){
       if(pa.charAt(empieza) == pa.charAt(pa.length() - 1 - empieza)){
              return DetectarPalindromo_recursiva(pa,  empieza + 1);
       }else{
              return false;
       }
      }else{
       if(empieza % 2 != 0){
              if(pa.charAt(empieza) == pa.charAt(pa.length() - 1 -
   empieza)){
                     return DetectarPalindromo_recursiva(pa,  empieza + 1);
              }else{
                     return false;
              }
       }else{
              return true;
       }
      }

}
```