Práctica 6 – S1: **cálculo de la raíz cuadrada (I)**

PROBLEMA: calcular el n-ésimo término de la siguiente recurrencia

$t_1 = \frac{1 + x}{2}, \quad t_{i+1} = \frac{t_i + x / t_i}{2}$

```
//PRECONDICIÓN: n >= 1 AND x >= 0.0
public static double calcularTermino(int n, double x) {
    double res = (1 + x) / 2; int i = 1;  // calculado t1
    while ( i != n    ){ // i < n
      i++;
      res = (res + x / res) / 2;          // calculado ti
    }
    // PARADA: i == n                     // calculado tn
    return res;
}
```

Práctica 6 – S1: **cálculo de la raíz cuadrada (II)**

PROBLEMA: calcular la **diferencia entre los términos n−1 y n** de la recurrencia

$t_1 = \frac{1 + x}{2}$, $t_{i+1} = \frac{t_i + x / t_i}{2}$

```
//PRECONDICIÓN: n >= 1 AND x >= 0.0
public static double calcularDifTerminos(int n, double x) {
    double res = (1 + x) / 2; int i = 1;
    double dif = res;                          // calculado t1 y dif1
    while (i < n) {
        double resAnterior = res;
        i++;
        res = (res + x / res) / 2;
        dif = resAnterior - res;               // calculado ti y difi
    }
                                               // calculado tn y difn
    return dif;
}
```

Práctica 6 – S1: **cálculo de la raíz cuadrada (III)**

PROBLEMA: calcular el término de la recurrencia $t_1 = \frac{1 + x}{2}$, $t_{i+1} = \frac{t_i + x / t_i}{2}$ cuya **diferencia con el que le precede sea menor que un cierto epsilon**

```
//PRECONDICIÓN: n >= 1 AND x >= 0.0 AND 0.0 <= epsilon <= 1
public static double calcularMejorTermino(      double x , double epsilon) {
    double res = (1 + x) / 2;
    double dif = res; // o epsilon + 1
    while (dif >= epsilon) {
        double resAnterior = res;

        res = (res + x / res) / 2;
        dif = resAnterior - res;
    }
    // PARADA: dif < epsilon
    return res;
}
```

Práctica 6 – S2 (a): **cálculo del log de z, 1/2 ≤ z < 1 (I)**

PROBLEMA parecido**:** sea **la serie** $log(z) = -2\sum_{i=1}^{\infty} \frac{y^{2i-1}}{2i-1}$ con $y = \frac{(1 - z)}{(1 + z)}$

Calcular su n-ésimo término, n ≥ 1

NOTA: se sabe que $u_1 = y$, $u_{i+1} = y^2 \frac{2i - 1}{2i + 1} u_i$

```
//PRECONDICIÓN: n >= 1 AND 1/2 <= z < 1
public static double calcularTermino(int n, double z) {
    double y = (1 - z) / (1 + z); double yAl2 = y * y;
    double res = y; int i = 1;      // calculado u1
    while (   i < n    ){
        res = yAl2 * res * (2 * i - 1) / (2 * i + 1);
        i++;                          // calculado ui+1
    }
   // PARADA: i == n                 // calculado un

    return res;
}
```

Práctica 6 – S2 (a): **cálculo del log de z, 1/2 ≤ z < 1 (II)**

PROBLEMA parecido**:** sea la serie $log(z) = -2\sum_{i=1}^{\infty} \frac{y^{2i-1}}{2i-1}$, con $y = \frac{(1 - z)}{(1 + z)}$

Calcular su n-ésimo **valor**, n ≥ 1

NOTA: se sabe que $u_1 = y$, $u_{i+1} = y^2 \frac{2i - 1}{2i + 1} u_i$

```
//PRECONDICIÓN: n >= 1 AND 1/2 <= z < 1
public static double calcularValor(int n, double z) {
    double y = (1 - z) / (1 + z); double yAl2 = y * y;
    double termino = y; int i = 1;      // calculado u1
    double res = termino;               // y  sumado u1
    while (i < n) {
        termino = yAl2 * termino * (2 * i - 1) / (2 * i + 1);
        i++;                            // calculado ui+1
        res = res + termino;            // y  sumado ui+1
    }
    // PARADA: i == n                   // calculado un
    return -2 * res;                    // y  sumado un
}
```

Práctica 6 – S2 (a): **cálculo del log de z, 1/2 ≤ z < 1 (III)**

PROBLEMA parecido: sea la serie $log(z) = -2\sum_{i=1}^{\infty}\frac{y^{2i-1}}{2i-1}$ con $y = \frac{(1 - z)}{(1 + z)}$

Calcular su valor **hasta aquel término que sea menor que un epsilon dado**

NOTA: se sabe que $u_1 = y$, $u_{i+1} = y^2 \frac{2i - 1}{2i + 1} u_i$

```
//PRECONDICIÓN: n >= 1 AND 1/2 <= z < 1 AND 0.0 <= epsilon <= 1
public static double calcularMejorValor(    double z, double epsilon) {
    double y = (1 - z) / (1 + z); double yAl2 = y * y;
    double termino = y; int i = 1;       // calculado u1
    double res = termino;                // y  sumado u1
    while (termino >= epsilon) {
        termino = yAl2 * termino * (2 * i - 1) / (2 * i + 1);
        i++;                             // calculado ui+1
        res = res + termino;             // y  sumado ui+1
    }
    // PARADA: termino < epsilon         // calculado un
    return -2 * res;                     // y  sumado un
}
```

Práctica 6 – S2 (b): **cálculo del log de x,** x > 0

PROBLEMA (Actividad #3 de la práctica): sabiendo calcular ya el valor del logaritmo de z para valores tales que $1/2 \leq z < 1$ (OJO: en la práctica el método `calcularMejorValor` se llama `logBase`), calcular el valor del logaritmo de cualquier $x > 0$

```
// PRECONDICIÓN: x > 0 AND epsilon > 0
public static double log(double x, double epsilon) {
    if (x == 1) { return 0; }

    // Cálculo de m y x tales que x = 2^m*z y 1/2 <= z < 1
    int m = 0; double z = x;

    while (z >= 1) { z = z / 2; m++; }

    while (z < 0.5) { z = z * 2; m--; }


    // Cálculo de log x
    return m * LOG_2 + logBase(z, epsilon);
}
```