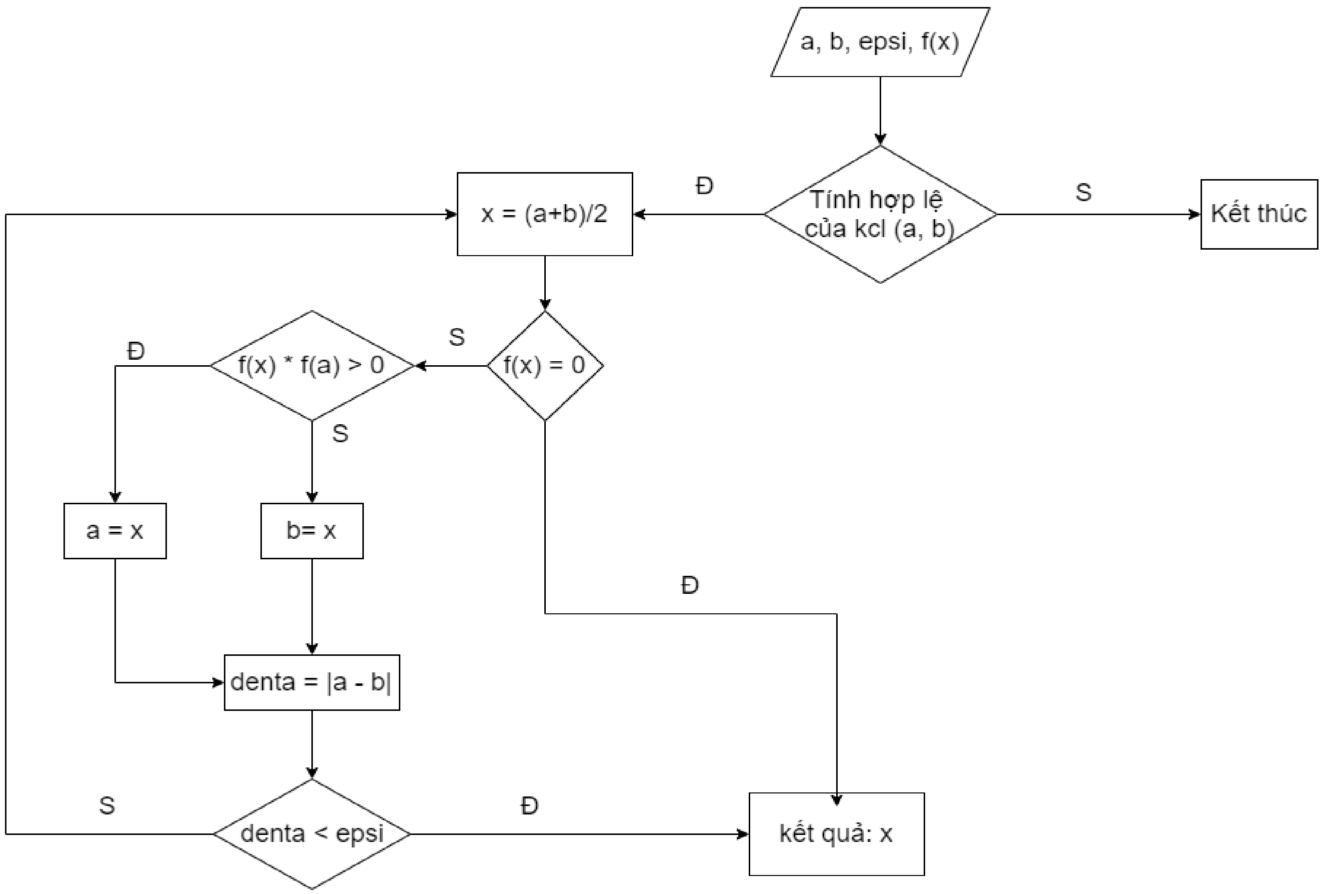
I. Thuật toán tổng quát

1. Công thức hậu nghiệm

Input: Phương trình f(x), khoảng cách li nghiệm (a, b), độ chính xác epsi

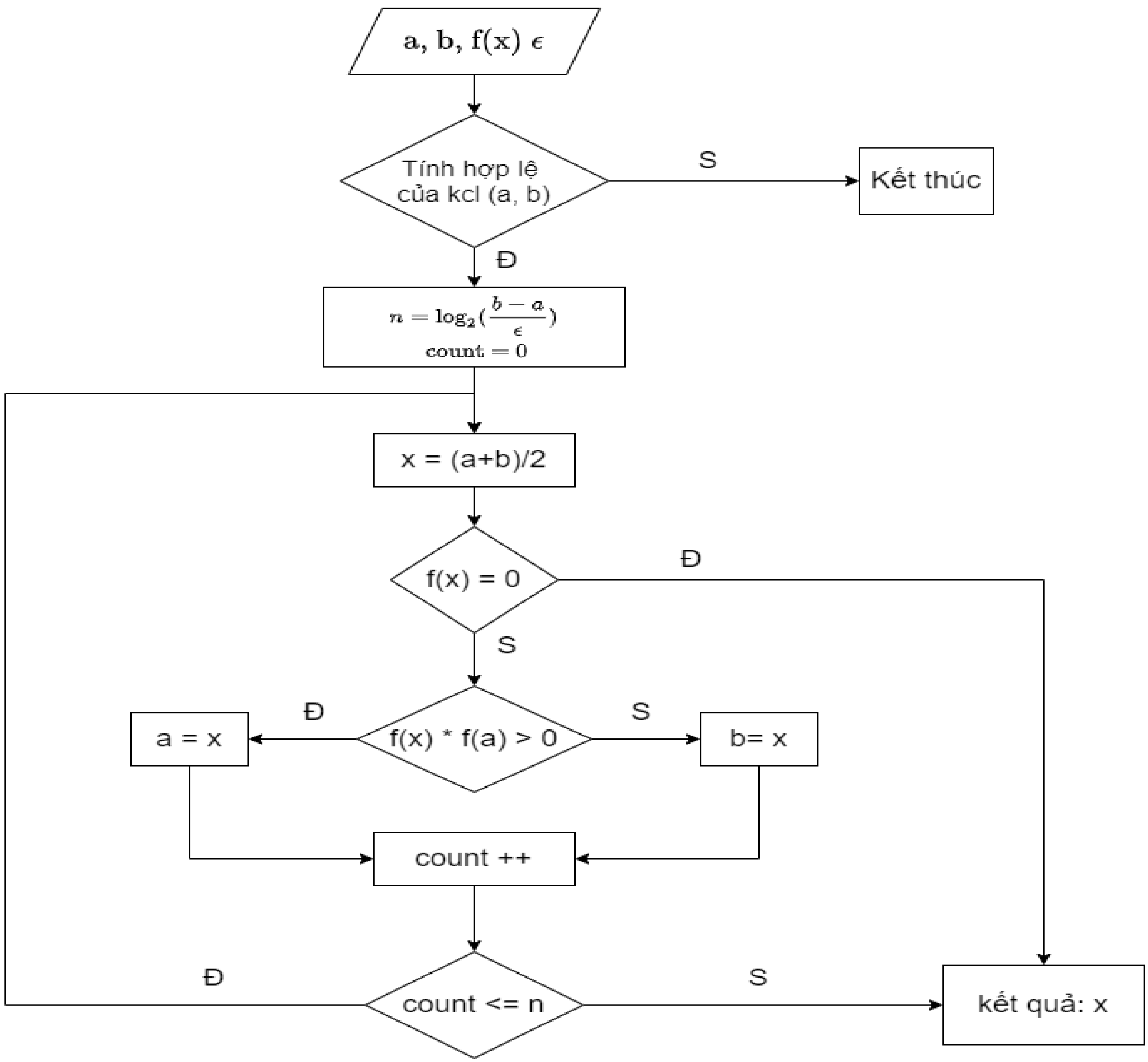
Output: Nghiệm của phương trình x



2. Công thức tiên nghiệm

Input: Phương trình f(x), khoảng cách li nghiệm (a, b), độ chính xác epsi.

Output: Nghiệm gần đúng của phương trình f(x) = 0



II. Thuật toán chi tiết (Giả mã)

1. Giải theo công thức hậu nghiệm

```
Input: hàm số f(x); khoảng phân ly nghiệm a, b: double; sai số epsi: double
Output: nghiệm của phương trình f(x) = 0

double solveBisection1:
      double c, denta
      if f(a) = 0:
            return a
      else if f(b) = 0:
            return b
      do:
           c = (a + b) / 2
           if f(c) = 0:
                return c
           if f(a) * f(c) > 0:
                 a = c
           else
                 b = c
           denta = |b – a|
      while (denta >= epsi)
      return c
```

2. Công thức tiên nghiệm

```
Input: hàm số f(x); khoảng phân ly nghiệm a, b: double; sai số epsi: double
Output: nghiệm của phương trình f(x) = 0


double solveBisection1:
      double c
      n = log2((b - a) / (2 * epsi))
      int count = 0
      for i = 0 to n:
           c = (a + b) / 2
           if f(c) * f(a) > 0:
                a = c
           else:
                b = c
           count ++

     return c
```

III. Ưu và nhược điểm của phương pháp

1. Ưu điểm
   - Dễ cài đặt thuật toán trên máy tính
   - Kết quả chính xác theo sai số epsilon người dùng nhập vào
   - Điều kiện rằng buộc chỉ cần có khoảng phân ly nghiệm a, b
2. Nhược điểm
   - Cần phải tìm khoảng phân ly nghiệm trước khi chạy thuật toán
   - Tốc độ hội tụ chậm hơn các phương pháp Lặp đơn, Newton, Dây cung