```
subalgorithm s1(n) is:
    for i ← 1, n execute
        j ← n
        while j ≠ 0 execute
            j ← ⌊j/2⌋
        end-while
    end-for
end-subalgorithm
```

_____

```
subalgorithm s2(n) is:
    for i ← 1, n execute
        j ← i
        while j ≠ 0 execute
            j ← ⌊j/2⌋
        end-while
    end-for
end-subalgorithm
```

_____

```
subalgorithm s3(x, n, a) is:
    found ← false
    for i ← 1, n execute
        if xᵢ = a then
            found ← true
        end-if
    end-for
end-subalgorithm
```

_____

```
subalgorithm s4(x, n, a) is:
    found ← false
    while found = false and i < n
execute
        if xᵢ = a then
            found ← true
        end-if
        i ← i + 1
    end-while
end-subalgoritm
```

```
Subalgorithm s5(x, n) is:
    k← 0
    for i ← 1, n execute
        for j ← 1, xᵢ execute
            k ← k + xⱼ
        end-for
    end-for
end-subalgorithm
```

_____

```
subalgorithm s6(n) is:
    for i ← 1,n execute
        @elementary operation
    end-for
    i ← 1
    k ← true
    while i <= n – 1 and k ex.
        j ← i
        k₁ ← true
        while j <= n and k₁ ex.
            @elementary op(k₁
        can be modified)
            j ← j + 1
        end-while
        i ← i + 1
        @elementary operation
    (k can be modified)
    end-while
end-subalgorithm
```

_____

```
subalgorithm p(x,s,d) is:
    if s < d then
        m ← [(s+d)/2]
        for i ← s, d-1, execute
            @elementary operation
        end-for
        for i ← 1,2 execute
            p(x, s, m)
        end-for
    end-if
end-subalgorithm
```

```
Subalgorithm s7(n) is:
      s ← 0
      for i ← 1, n² execute
            j ← i
            while j ≠ 0 execute
                  s ← s + j
                  j ← j − 1
            end-while
      end-for
end-subalgorithm
```

—————————————————————————————

```
Subalgorithm s8(n) is:
      s ← 0
      for i ← 1, n² execute
            j ← i
            while j ≠ 0 execute
                  s ← s + j − 10 *
      [j/10]
                  j ← [j/10]
            end-while
      end-for
end-subalgorithm
```

—————————————————————————————

```
subalgorithm operation(n, i) is
      if n > 1 then
            i ← 2 * i
            m ← [n/2]
            operation(m, i-2)
            operation(m, i-1)
            operation(m, i+2)
            operation(m, i+1)
      else
            write i
      end-if
end-subalgorithm
```