Algorithm 1: Equal-Filling Control Algorithm: Let i be a tank in the network of $\mathcal{N}$ tanks. In scenario theta, $\mathcal{N} = 2$ and Max depth in each tank is 2.0m

Let λ be the target flow to be achieved
for *all* $\mathcal{N}$ *tanks* **do**
  Compute the *filling degree*; $f_i = {}^{\text{depth}_i}/_{\text{Max depth}_i}$
Estimate the *average filling degree*; $\bar{f} = \sum_i^N {}^{f_i}/_N$
for *all* $\mathcal{N}$ *tanks* **do**
  Let $\psi_i = f_i - \bar{f}$
  if $\psi_i < 0.0$ **then**
    $\psi_i = 0.0$
  else if $\psi_i = 0.0$ **then**
    $\psi_i = \bar{f}$
for *all* $\mathcal{N}$ *tanks* **do**
  Assign valve positions; $v_i \propto \lambda \times \{{}^{\psi_i}/\sum_i^N \psi_i\}$

```python
def controller(depths,
               N=2,
               LAMBDA=0.3,
               MAX_DEPTH=2.0):
    # Compute the filling degree
    f = depths/MAX_DEPTH

    # Estimate the average filling degree
    f_mean = np.mean(f)

    # Compute psi
    psi = np.zeros(N)
    for i in range(0, N):
        psi[i] = f[i] - f_mean
        if psi[i] < 0.0:
            psi[i] = 0.0
        elif psi[i] == 0.0:
            psi[i] = f_mean

    # Assign valve positions
    actions = np.zeros(N)
    for i in range(0, N):
        if depths[i] > 0.0:
            k = 1.0 / np.sqrt(2 * 9.81 * depths[i])
            action = k * LAMBDA * psi[i]/np.sum(psi)
            actions[i] = min(1.0, action)

    return actions
```