// ProceduralTerrain

```
float height11 = 0f;

float amplitude = 1f;          We then divide our current x coordinate
float frequency = 1f;          by the product of Scale * frequency

for (int i = Octaves; i > 0; i--) {       Example
    float octave_x0 = x        / Scale * frequency;
    float octave_z0 =  z       / Scale * frequency;     Scale = 15      x = 1
    float octave_x1 = (x + 1f) / Scale * frequency;
    float octave_z1 = (z + 1f) / Scale * frequency;     15 x 1f = 15   1 / 15 = 0.06667f

    height00 += Mathf.PerlinNoise(octave_x0, octave_z0) * amplitude;
    height01 += Mathf.PerlinNoise(octave_x0, octave_z1) * amplitude;
    height10 += Mathf.PerlinNoise(octave_x1, octave_z0) * amplitude;
    height11 += Mathf.PerlinNoise(octave_x1, octave_z1) * amplitude;

    amplitude *= Persistance;
    frequency *= Lacunarity;
}

int x0 =  x        * CellSize;
```

```
float height11 = 0f;

float amplitude = 1f;
float frequency = 1f;
```

We then divide our current x coordinate
by the product of Scale * frequency

```
for (int i = Octaves; i > 0; i--) {
    float octave_x0 = x         / Scale * frequency;
    float octave_z0 =  z        / Scale * frequency;
    float octave_x1 = (x + 1f) / Scale * frequency;
    float octave_z1 = (z + 1f) / Scale * frequency;
```

Example

Scale = 15   x = 3

$15 \times 1f = \boxed{15}$ $3 / \boxed{15} = 0.2f$

```
    height00 += Mathf.PerlinNoise(octave_x0, octave_z0) * amplitude;
    height01 += Mathf.PerlinNoise(octave_x0, octave_z1) * amplitude;
    height10 += Mathf.PerlinNoise(octave_x1, octave_z0) * amplitude;
    height11 += Mathf.PerlinNoise(octave_x1, octave_z1) * amplitude;

    amplitude *= Persistance;
    frequency *= Lacunarity;
}

int x0 =  x        * CellSize;
```