

```

def synthetic_programs():

    control_flow = {0: 'if',
                    1: 'else',
                    2: 'for',
                    3: 'body',
                    4: 'end'}

    nlines = 2
    depth = 1
    maxdepth = depth

    # set up probs
    probs_with_else = [1.0, 2.5, 1.0, 2.5, 1.0]
    probs_with_else = [p / sum(probs_with_else) \
                       for p in probs_with_else]

    probs_no_else = [1.0, 0, 1.0, 2.5, 1.0]
    probs_no_else = [p / sum(probs_no_else) \
                     for p in probs_no_else]

    depth_allow_else = {depth: False}

    while True:
        if depth_allow_else[depth]:
            probs = probs_with_else
        else:
            probs = probs_no_else

        # sample the statement type
        s = sample(Categorical(probs),
                  address=f"stat_{depth}_{nlines}")
        statement = control_flow[s]

        if statement == 'body':
            s = sample(Categorical([0.33, 0.33, 0.34]),
                      address=f"op_{depth}_{nlines}")
            num = sample(Normal(0.0, 1.0),
                        address=f"mod_{depth}_{nlines}")

            nlines += 1
            probs[0] *= 0.5
            probs[1] *= 0.5
            probs[2] *= 0.5
            probs[3] *= 0.5
            probs = [p / sum(probs) for p in probs]

        elif statement == 'if':
            s = sample(Categorical([0.5, 0.5]),
                      address=f"cond_{depth}_{nlines}")
            num = sample(Normal(0.0, 1.0),
                        address=f"comp_{depth}_{nlines}")

            depth += 1
            nlines += 1

            if depth > maxdepth:
                maxdepth = depth

            probs[0] *= 0.5
            probs[1] *= 0.5
            probs[2] *= 0.5
            probs = [p / sum(probs) for p in probs]

            depth_allow_else[depth] = True

        elif statement == 'else':
            if depth > maxdepth:
                maxdepth = depth

            probs[0] *= 0.5
            probs[1] *= 0.5
            probs[2] *= 0.5
            probs = [p / sum(probs) for p in probs]

            depth_allow_else[depth] = False

        elif statement == 'for':
            depth += 1
            nlines += 1
            range_val = sample(Categorical([1,5,2,8,4,2,5,7,98]),
                              address=f"range_{depth}_{nlines}")

            depth_allow_else[depth] = False

            if depth > maxdepth:
                maxdepth = depth

            probs[0] *= 0.5
            probs[1] *= 0.5
            probs[2] *= 0.5
            probs = [p / sum(probs) for p in probs]

        else: # is end
            probs[0] *= 0.5
            probs[1] *= 0.5
            probs = [p / sum(probs) for p in probs]
            del depth_allow_else[depth]
            depth -= 1

        if depth == 0:
            break

    return maxdepth

```