E1.py

```python
from random import choice

N = 100


def sgn(x):
    if x < 0:
        return -1
    if x >= 0:
        return 1


def generate_patterns(p, n):
    # p is the number of patterns to generate
    # n is the number of bits
    result = list()
    for pattern in range(p):
        new_pattern = list()
        for bit in range(n):
            new_pattern.append(choice([-1, 1]))
        result.append(new_pattern)
    return result


def learn_patterns(patterns, n):
    w, h = n, n
    W = [[0 for x in range(w)] for y in range(h)]

    for p in patterns:
        for i in range(n):
            for j in range(i, n):
                if i != j:
                    W[i][j] += 1 / n * p[i] * p[j]
                else:
                    W[i][j] = 0
                W[j][i] = W[i][j]
    return W


def get_p_error_estimation(p):
    n_errors = 0

    for trial in range(100000):

        patterns = generate_patterns(p, N)
        # learn patterns
        W = learn_patterns(patterns, N)

        # choose one random pattern to feed
        v = choice(patterns)
        neuron = choice(range(N))

        previous_state = v[neuron]

        new_state = 0
        for j in range(N):
```

```
                new_state += W[neuron][j] * v[j]

            if sgn(new_state) != previous_state:
                n_errors += 1

        return n_errors / 100000


P_values = [12, 20, 40, 60, 80, 100]

for p in P_values:
    p_error = get_p_error_estimation(p)
    print(str(p) + " : " + str(p_error))
```