```
% Shannon Fano Coding
% Created on: 13/02/25
% By Author: Supratit Datta, BT22ECE127

clc;
clear;
close all;

symbols = ['A', 'B', 'C', 'D', 'E'];
probabilities = [0.5, 0.2, 0.2, 0.05, 0.05];

% Ensure probabilities sum to 1
if sum(probabilities) ~= 1
    probabilities = probabilities / sum(probabilities);
end

% Create a cell array with symbols and their probabilities
n = length(symbols);
items = cell(n, 2);
for i = 1:n
    items{i, 1} = symbols(i);
    items{i, 2} = probabilities(i);
end

% Sort items in descending order based on probability
items = sortrows(items, -2);

% Generate Shannon-Fano Codes
[codes, codewords] = generate_code(items, '');

% Display the Shannon-Fano Codes
fprintf('Shannon-Fano Codes:\n');
for i = 1:length(codes)
    fprintf('Symbol: %s, Code: %s\n', codes{i}, codewords{i});
end

% Recursive Shannon-Fano Function
function [codes, codewords] = generate_code(items, prefix)
    n = size(items, 1);

    if n == 1
        % Base case: Only one symbol left, assign the prefix as its code
        codes = {items{1, 1}};
        codewords = {prefix};
        return;
    end

    % Compute cumulative probability
    cumulative_prob = cumsum(cell2mat(items(:, 2)));

    % Find the best splitting point where the sum is closest to 50%
    [~, split_idx] = min(abs(cumulative_prob - cumulative_prob(end) / 2));
```

```
    % Split into left and right groups
    left_items = items(1:split_idx, :);
    right_items = items(split_idx+1:end, :);

    % Recursively generate codes for left and right groups
    [left_codes, left_codewords] = generate_code(left_items, [prefix '0']);
    [right_codes, right_codewords] = generate_code(right_items, [prefix '1']);

    % Combine results
    codes = [left_codes, right_codes];
    codewords = [left_codewords, right_codewords];
end
```

```
Shannon-Fano Codes:
Symbol: A, Code: 0
Symbol: B, Code: 10
Symbol: C, Code: 110
Symbol: D, Code: 1110
Symbol: E, Code: 1111
```

Published with MATLAB® R2024b