```
% Created by Vipul Pawar
% Shannon-Fano Coding Implementation

clear;
clc;

% Define symbols and corresponding frequencies
symbols = {'0-30', '31-59', '60-63', '64-100', '101-127', '128-150',
'151-200', '201-255'};
frequencies = [2048, 2048, 2048, 2048, 819, 819, 3277, 3277];

% Compute probabilities
probabilities = frequencies / sum(frequencies);

% Sort probabilities in descending order
[probabilities, order] = sort(probabilities, 'descend');
symbols = symbols(order);

% Initialize code storage
codes = cell(size(symbols));
code_lengths = zeros(size(symbols));

% Stack-based implementation of Shannon-Fano algorithm
stack = {{symbols, probabilities, ''}};

while ~isempty(stack)
    % Retrieve the last element from the stack
    data = stack{end};
    stack(end) = [];

    s = data{1};
    p = data{2};
    prefix = data{3};

    % Assign code if only one symbol remains
    if length(s) == 1
        index = strcmp(symbols, s{1});
        codes{index} = prefix;
        code_lengths(index) = length(prefix);
        continue;
    end

    % Determine split point where cumulative probability is approximately half
    split_idx = find(cumsum(p) >= sum(p) / 2, 1);

    % Push the two partitions into the stack
    stack{end+1} = {s(1:split_idx), p(1:split_idx), strcat(prefix, '0')};
    stack{end+1} = {s(split_idx+1:end), p(split_idx+1:end), strcat(prefix,
'1')};
end

% Display results
```

```
fprintf('Symbol\tCode\tLength\n');
for i = 1:length(symbols)
    fprintf('%s\t%s\t%d\n', symbols{i}, codes{i}, code_lengths(i));
end
```

```
Symbol    Code    Length
151-200    000    3
201-255    001    3
0-30    01    2
31-59    100    3
60-63    101    3
64-100    110    3
101-127    1110    4
128-150    1111    4
```

Published with MATLAB® R2024b