Kodowanie CRC

W tym zadaniu, najpierw odczytuję zawartość pliku Z.txt. Następnie, z pomocą biblioteki crc32, obliczam CRC odczytanej wiadomości, a początek uzupełniam o 0, aby cała wiadomość była długości 32 bitów.

Kodowanie odbywa się poprzez dopisanie, stworzonego powyższym sposobem, CRC na początek. Iteruję następnie po 32-bitowej wiadomości, zliczając jedynki – jeśli napotkam ich 5 pod rząd, to dopisuję 0. Finalnie, na początek oraz koniec wiadomości dodaję ciąg 01111110

Tak zakodowaną wiadomość zapisuję w W.txt

Odkodowanie to w zasadzie odwrotność powyżej procedury – usunięcie 01111110 z początku oraz końca, 0 po każdej piątce jedynek, rozdzielenie wiadomości na CRC i właściwą wiadomość, i ponowne obliczenie CRC w celu sprawdzenia integralności danych.

Symulacja CSMA

W mojej implementacji kanał reprezentuje prosta tablica długości 10 (długość można dostosować dowolnie). Urządzenia wysyłające sygnał reprezentowane są przez klasę Sender. Klasa ta przechowuje zmienne dotyczące obecnej sytuacji – ile ma czekać przed wysłaniem wiadomości, czy została ona wysłana, czy doszło do kolizji. Zawiera tez takie metody jak move, odpowiadającą za „poruszanie się” wiadomości po kanale, czy collsion, odpowiadającą za zachowanie urządzenia podczas kolizji (urządzenie czeka losową liczbę sekund)

Funkcja process odpowiada za zarządzanie kanałem. Przechowuje informacje: które urządzenie otrzymało wiadomość, czy doszło do kolizji. Robi to poprzez sprawdzanie, gdzie obecnie na kanale znajduje się wysyłana wiadomość.

Głowny main odczytuje wartości zwrócone przez funkcję process, i zachowuje się odpowiednio do nich – widzi, kto otrzymał wiadomość, a także czy doszło do kolizji.