1) Rappel du sujet

Le but du TP est de pouvoir calculer et vérifier des messages envoyés avec du code CRC avec un polynome générateur de son choix.

2) Analyse du sujet

Pour calculer un code CRC, on choisit un message, sous forme de nombres binaires. Par exemple : 10010010101

Ensuite, on choisit un polynome de degré 4 : x^4 + x^2 + x, ce qui donne 10110.

Comme le polynome est de degré 4, on ajout 4 zéros à la fin du message. On obtient : 10010010101

Il s'agit là de faire une addition binaire : 1 + 1 = 0, 0 + 0 = 0, 1 + 0 et 0 + 1 = 1.

```
10010010101
10110
---------------
00100010101
  10110
---------------
  001110101
    10110
---------------
    0101101
     10110
---------------
     000001
          10110
```

Dès que la longueur du polynome générateur est supérieure à celle du message restant, alors on a notre CRC. Ici, notre CRC est de 00001, donc 1.

3) Choix techniques utilisés.

Pour cela, nous avons utilisé les StringBuilder en java pour facilement manipuler les 0 et les 1. Toutes les étapes énumérées en haut ont été simplement traduites sous forme d'algorithmes, on fait exactement la même chose.

On a donc une fonction permettant de récupérer le CRC en fonction d'un message sous forme de nombres binaires et d'un polynome générateur (sous forme binaire, aussi)

Une fonction qui permet de générer les étapes sous forme humainement lisibles.

Une fonction qui permet de supprimer les zéros à gauche jusqu'à au premier 1 trouvé, nous permettant ainsi de refaire le calcul correctement à chaque étapes.

Une fonction qui faire un calcul de nombres binaires :
- si opérande de gauche == opérande de droit on retourne 0, sinon 1.

Une fonction qui permet de vérifier un message. Celle-ci prend un message en nombres binaires ainsi que le polynome générateur.

Une dernière fonction qui permet d'encoder un message en fonction d'un string normal passé en paramètres et transformé en nombre binaires.

4) Résultats et tests

```
=== TP2 Réseau | Allan Mercou – Léon Souffes | CRC ===
> [Q]uit: quitte le programme
> [E]ncode: permet d'encoder un message avec le polynome générateur
de son choix.
> [C]heck: permet de vérifier que le message envoyé ne contient pas
d'erreurs.
Entrez votre choix.
e
Entrez le message à encoder :
slt
Entrez le polynome générateur à utiliser :
10110
Message initial : 101001110011001010100


Etapes :
101001110011001010100
10110
-------------------------
   101110011001010100000
   10110
-------------------------
       10011001010100000
       10110
-------------------------
         1010010101000000
         10110
-------------------------
            1010101000000
            10110
-------------------------
               1101000000
               10110
-------------------------
                110000000
                10110
-------------------------
```

```
                 11100000
                 10110
-------------------------
                  1010000
                  10110
-------------------------
                     1000
                     10110
-------------------------
```

CRC trouvé : 1000
Message à envoyer : 10100111001100101010010000

=== TP2 Réseau | Allan Mercou – Léon Souffes | CRC ===
> [Q]uit: quitte le programme
> [E]ncode: permet d'encoder un message avec le polynome générateur de son choix.
> [C]heck: permet de vérifier que le message envoyé ne contient pas d'erreurs.
Entrez votre choix.
c
Entrez le message à vérifier :
10100111001100101010010000
Entrez le polynome générateur à utiliser :
10110

Etapes :

```
10100111001100101010010000
10110
----------------------------
   10111001100101010010000000
   10110
----------------------------
       10011001010100100000000
       10110
----------------------------
         1010010101001000000
         10110
----------------------------
            101010100100000000
            10110
```

```
------------------------------
               11010010000000
               10110
------------------------------
                1100010000000
                10110
------------------------------
                 111010000000
                 10110
------------------------------
                  10110000000
                  10110
------------------------------

                              10110
------------------------------


Le message reçu est valide !
```

5) Conclusion

Il aurait été possible de faire une méthode permettant de repasser d’un nombre binaire à un message humainement lisible. Pour les difficultés rencontrées, il s’agissait de correctement manipuler les strings. Heureusement, la classe StringBuilder est très pratique et faite pour ça.