

Université d'Ottawa  
Faculté de génie

École de science informatique  
et de génie électrique



University of Ottawa  
Faculty of Engineering

School of Electrical Engineering  
and Computer Science

---

## CEG3585 : Introduction à la communication de données et au réseautage Printemps/été 2022

Professeur: Mohamed Ali Ibrahim, ing., Ph.D.  
Bureau: STE 0110B

Courriel: [mibrahi3@uottawa.ca](mailto:mibrahi3@uottawa.ca)

Assistant d'enseignement: Mohamed Ali Jemmali

Courriel: [mjemm073@uottawa.ca](mailto:mjemm073@uottawa.ca)

### Programmation Socket et B8ZS **Travaillez en équipe de deux** Dû le 27 juin 2022

#### Objectifs :

Le but de ce laboratoire est d'implémenter deux programmes en utilisant les sockets pour transmettre B8ZS à partir du programme d'encodage (CLIENT) vers le programme de décodage (SERVEUR). Le programme de décodage décode le flux B8ZS reçu au format original.

#### Développement :

1. L'entrée du programme de codage (client) est une chaîne de 0 et 1, entrée à partir du clavier.
2. La chaîne binaire d'entrée est encodée par le programme d'encodage en B8ZS. On suppose toujours que la polarité du premier 1 est positive.
3. Le programme d'encodage doit envoyer un message « demande d'envoi » au programme de décodage et attendre la réponse « prêt à recevoir » du programme de décodage, avant de l'envoyer.
4. Le programme de décodage (serveur) envoie un message « prêt à recevoir » au programme de codage après avoir reçu une « demande à envoyer » du programme de codage.
5. Le flux B8ZS est alors transmis au programme de décodage via le socket.
6. Après réception du flux B8ZS, le programme de décodage accusera réception au programme d'encodage.
7. Ensuite, le programme de décodage décode le flux B8ZS dans son format d'origine et l'imprime à l'écran.
8. Le flux encodé de B8ZS est représenté par la séquence de trois caractères, « + », « - » et « 0 », ce qui signifie respectivement l'impulsion positive, l'impulsion négative, et aucun signal.
9. Vous devez soumettre dans un fichier zip votre code et un fichier README expliquant comment exécuter vos programmes et ce qui est attendu en entrée / sortie.

#### Instructions additionnelles:

**Vous devez implémenter deux programmes**, un client et un serveur et deux fonctions (méthodes), pour encoder (côté client) et pour décoder (côté serveur).

##### ○ Le client:

- Lit les données de l'utilisateur (à partir du clavier)
- **Encode les données en B8ZS**
- Envoie une « demande d'envoi » au serveur via le socket, et attend
- Si le serveur envoie un « prêt à recevoir », alors  
Envoie le flux codé au serveur via socket

##### ○ Le serveur :

- Confirme que les données du client ont été reçues
- Décode le flux de données et l'imprime à l'écran

**Exemple.**

1. L'utilisateur saisit du clavier : 1100000000110000010
2. Client encode les données : + -000- + 0 + - - - + 00000 -
3. Serveur reçoit le flux codé et le décode à : 1100000000110000010
4. Le serveur affiche à l'écran 1100000000110000010

**Vous pouvez coder les deux programmes en utilisant le langage de programmation de votre choix :**  
C/C++, Java, Python, C# ou autres langages

Pour plus d'informations :

Ce qui suit sera schéma utilisé par le marquage TA :

Composante	Points
1. Le client communique à un serveur via un socket	20 points
2. L'encodage fonctionne correctement	10 points
3. Le décodage fonctionne correctement	10 points
4. README décrivant comment exécuter le code	5 points
5. Du côté client, les données sont lues à partir du clavier	5 points
<b>Total</b>	<b>50 points</b>

Les fonctions de codage et de décodage seront testées en utilisant les cas de tests suivants :

Entrée	Sortie (encodage)
100000000	+ 000 + -0- +
100000000000100	+000+-0-+000-00
1100000000110000010	+ -000-+0+--00000+0
1101001	+ -0 + 00-

**Le fonctionnement 50 % et le rapport 50 %.**