1 Serialisation Simple

Question 1 Créé une classe SerializerBuffer qui contient un ByteBuffer et des méthodes pour lire et écrire des entiers et des flottants.
void writeInt(int), void writeFloat(float), int readInt(), float readFloat()

Question 2 Ajouté à la classe précédente des méthodes pour lire et écrire des chaînes de caractère encodé en UTF-8.
void writeString(String), String readString()

Soit l'interface suivante :

```
public interface MySerialisable {
  public void writeToBuff(SerializerBuffer ms);
  public void readFromBuff(SerializerBuffer ms);
}
```

Question 3 Écrire une classe de test (Test1) qui contient un entier et une chaîne de caractère et qui implémente l'interface MySerialisable. La classe doit surcharger toString() pour renvoyer un message informatif.

Question 4 Ajouté un constructeur sans argument à la classe de test. Écrire une méthode main qui fabrique un objet de la classe de test le sérialise et le desérialise. *La déserialisation commence par appeler le constructeur vide puis appel la méthode readFromBuffer().*

Question 5 Écrire une seconde classe de test (Test2) qui contient deux flottants et qui implémente l'interface MySerialisable. Ajouter à la classe Test1 un champ de type Test2. Mettre à jour les fonctions de sérialisation.

Question 6 Que ce passe-t-il,quand le champ Test2 de la classe Test1 == **null**.

2 Gestion des objets null

Soit l'interface

```
public interface Creator<T extends MySerialisable>{
  public abstract T init();
}
```

Question 7 Ajouté un champ **public static final** Creator CREATOR aux classes Test1 et Test2 qui appel le constructeur vide de la classe.

Question 8 Ajouté à SerializerBuffer deux méthodes :

```
  public void writeMySerialisable( MySerialisable );
  public <T extends MySerialisable> T readMySerialisable( Creator<T> );
```

Modifier les classes Test1 et Test2 Afin qu'elle utilise ces méthodes pour la sérialisation.

Question 9 Modifier les deux méthodes la question précédente pour gérer le cas des objets **null**.

3 Gestion des structures récursives circulaires

Question 10 Ajouté à la classe SerializerBuffer les champs suivant :

```
private Map<Integer,Integer> objMap = new HashMap<>();
private ArrayList<MySerialisable> objArray = new ArrayList<>();
```

Question 11 Modifier la méthode writeMySerialisable pour qu'elle enregistre tous les objets qu'elle sérialise dans le tableau objArray. On utilisera de plus System.identityHashCode(Object) pour générer un identifiant unique de chacun des objets que l'on ajoutera à la table objMap avec la position de l'objet dans le tableau.

Question 12 Modifier les méthodes writeMySerialisable() et readMySerialisable() pour gérer les structures récursives circulaires.