**Class AnnotationArray** – a set of phrases, their concept annotations, and documents that they come from

String [] **getNextData** - returns an array with document number as the first position, concept in the last position, and phrase in between with one word per position. Returns null if there is no more data.

void **reset**() – Resets the array so that the next call to getNextData() returns the first item in the object.

**Class AnnotationSet**  - instantiated objects from this class store a set of annotated phrases, the document the phrase comes from, and the concept used to annotate the phrase. Also stores the longest phrase (in terms of number of tokens or words) among all phrases in the set.

int **maxPhraseLength** – Longest phrase among all annotated phrases in set

AnnotationArray **annotations** – object to store all the phrases along with their associated document numbers and concepts (annotations)

String[] **getNextData**() – returns an array with document number as the first position, concept in the last position, and phrase at the indexes between the first and last with one word per index position. Returns null if there is no more data.

CrossValObject **generateCrossValObject**(int numCross, longI randSeed) – returns a CrossValObject where numCross indicates the cross-fold validation. numCross must be between 2 and the number of annotated phrases in the annotation set.

int **getMaxPhraseLength**() – Returns the maxPhraseLength attribute

**Class CrossValObject**  - an object from this class stores a set of annotated phrases, their document numbers, and the concepts used to annotate each phrase. In addition, it contains an indicator of the number of cross-fold validations that will be used on the set

int **numCross** – number of cross-fold validations to be run. The number should be a minimum of 2 and a maximum of the number of annotated phrases in the annotation set

storageObject? **Annotations -** object to store all the phrases along with their associated document numbers and concepts (annotations)

Other structures???

AnnotationArray **currentTestingSet** – Holds 1/numCross documents used for training the machine learning system. So, if there are 200 documents and numCross is 40, this should return 1/5 or 0.2 of the documents. The documents chosen should be random. Except that each call to getNextSet() should generate a new currentTrainingSet (and new currentTrainingSet) containing none of the documents previously contained within currentTrainingSet.

AnnotationArray **currentTrainingSet –** Holds all documents not included in the currentTrainingSet.

public void **createNextSet()** - Generates a new currentTestingSet and currentTrainingSet. Each time this is called, the currentTestingSet should not contain any documents previously within currentTestingSet. It should not be possible to call this more than numcCross times without a call to reset(). If a call is made to getNextSet() without calling reset(), null should be returned.

All documents should be returned after calling getNextTrainingSet() numCross times. So, a set with 10 documents would be divided into sets of 3, 3, and 4 documents. Whether the call returns 3 or 4 documents should be random. (Note that for current project, we should keep the “randomness” the same every time, so we should always use the same random seed for determining which set of documents to return and also which set, if needed, has a different number of documents).

public AnnotationSet getCurrentTraningSet() – returns currentTrainingSet

public AnnotationSet getCurrentTestingSet() – returns currentTestingSet