**Muzaffar Sharapov**

**PART A –** OOP Class Design Guidelines, **15 points**

## 1. Cohesion

**Since enums in java cannot be abstract. I've followed standard Java naming conventions. I may use a little bit long naming; however, it's easier for me to understand when I come back to the future.**

## Encapsulation

## I used getters for Enum. Here is the code example:

```
public String getKeyWordFromEnum() {
    return this.keyWordFromEnum.toLowerCase();
}

public String getPartOfSpeechFromEnum() {
    return this.partOfSpeechFromEnum;
}

public String getValueDefinitionFromEnum() {
    return this.valueDefinitionFromEnum;
}
```

1. **Program Analysis to Program Design**, **15 points.** *Please think about Interviews.*

In at least 1 full page, please explain the following **in detail**:

Your analysis of the provided information and the provided complete sample output. *Please think about Clients and Sales.*

▪ What problem you are solving. Please explain it clearly then define it concisely. *Please think about Problem Solving and Interviews.*

*Right now, I'm thinking of a better way to implement an interactive dictionary. I thought if the order didn't matter, I could just use sethash to get rid of duplicates for case "word distinct". I thought about what's faster to get, containsKey vs sorting. And I realized it's faster to use get, containsKey inside the loop to check the duplicates than setting into SetHash and then sorting.*

*Here is the toDistinct method:*

```
public static ArrayList<DictionaryEnum> toDistinct(ArrayList<DictionaryEnum>
dictionaryArr) {


Set<DictionaryEnum> hashSet = new HashSet<>(dictionaryArr);


ArrayList<DictionaryEnum> distinctArrListFromHashSet = new ArrayList<>();

dictionaryArr.clear(); dictionaryArr.addAll(hashSet);

  Collections.sort(dictionaryArr);

HashSet<String, DictionaryEnum> hashMapOfValueAndArrDict = new HashSet<String,
DictionaryEnum>();


ArrayList<DictionaryEnum> hashMapOfValuesToSortedArrList = new

ArrayList<HashSet>(hashMapOfValueAndArrDict); if

(!hashMapOfValueAndArrDict.toString().equalsIgnoreCase("reverse")) {

    Collections.sort(hashMapOfValuesToSortedArrList);

}

  Collections.sort(distinctArrListFromHashSet);


System.out.println("distinctArrListFromHashSet.toString().equalsIgnoreCase:" +

distinctArrListFromHashSet.toString().equalsIgnoreCase("reverse"));

System.out.println("hashSet " + hashSet); if

(!distinctArrListFromHashSet.toString().equalsIgnoreCase("reverse")) {

    Collections.reverse(distinctArrListFromHashSet);

} distinct =

false;

        return dictionaryArr;

}
```

▪ How you store data in enum objects. And why. *Please think about Data Structures and Data Design.*

*I stored multiple values in one enum key that way I could have separate key word, POS, value definition.*

2. **Program Implementation**, **70 points.** *Please think about Interviews.*

After 20 inputs on Search[20] reverse reverse, or distinct distinct doesn't work, but if we wait a little then it works without rebuilding or rerunning. I suspect it's because it takes some time for garbage collector to do its job.

There is always room to improve the program. One thing I was trying to do is by getting read of naming (book1 book2 ...) Instead, the better approach would be to use In order to make the program more readable; I would make the dictionary program more object-oriented design. But we use static methods, which cannot be used in abstract methods.