Farhan Fuad Haque

Roll : 35

Assumptions

- It is assumed that the program will run on some server for an indefinite period of time. Thus the vocabulary is stored in an array (main memory) instead of secondary storage.
- A sentence can contain the same word multiple times.
- A sentence can be one word. (Reference)
- The sentence generator does not necessarily generate a meaningful sentence.
- In SSG, the words are sorted alphabetically before printing
- In OSG, since the vocabulary can be very large, a number of words are randomly selected first and then they are ordered in the sequence they were entered into the vocabulary.
- For all the 3 generators, the number of words in a sentence are decided randomly before each sentence generation.
- Since the number of words in a sentence is decided randomly, there needs to be a max possible number of words pre-defined. The maximum number of words in a sentence is assumed to be 30. The *MAX_NUMBER_OF_WORDS* attribute in the SentenceGeneratorContext class holds this value which the concrete generator classes (RSG, SSG, OSG) use to limit the number of words.
- The Math.random() method in JAVA is assumed to generate evenly distributed random numbers.
- It is assumed that each generator has its own individual vocabulary.
- The words in the vocabulary may not be unique.
- It is assumed that user may enter wrong operation number or wrong input type (ie. string instead of integer). These cases are handled.
- It is assumed that users will insert correct words. Thus words are not validated before being inserted into the vocabulary.
- The vocabulary is initially assumed to be empty. Thus the generators cannot generate any sentence initially. Rather a message is displayed requesting the user to entrer a word first.
- It is assumed that different sentence generator strategies might be implemented in future. Therefore the Strategy Design Pattern is followed. And thus the generators are accessed using an interface they all implement instead of accessing each of them directly.