1. Write a program to enter a string. Write following methods:
- Returning the sum of characters (that are not space character) of the input string
- Print out **the reserve string** of the input string
- If the string's length > 3, print out the third character of the input string. Otherwise, print out to the console the message: "undefined"
- Write an algorithm to accept two strings and check whether the second string exists within the first string. For example, if the first string is "concatenation" and the second string is "cat", the algorithm should display "Substring found at position 4 in the string". However, if the first string is "concatenation" and the second string is "tent", the algorithm should display "Substring not found in the string".
- How many the numeral characters, the vowel characters, the consonant characters and the special characters does the input string have?
- How many words in the input string?
- Print out the lower case letters of input string
- Print out the upper case letters of input string
- Replace letters that are identical and uninterrupted by one (ex: abcbbbcca -> abcbca)
- Create a method that input a string then all remove unnescessary blanks (ex: " I want to   go to   the cinema   " → "I want to go to the cinema")
- Create a method that input a string then count the number of presence of each letter.
- Create a method that input a string and an integer n then output n letters from the right side of that string
- Create a method that convert an input to title case (ex: "java programming class" → "Java Programming Class")
- Create a method that convert an input to the English-name-like string (ex:"Pham Thanh Son" → "Son Pham Thanh")
- Create a method that reverse an input by each word (ex "Pham Thanh Son" → "Son Thanh Pham")

2. Write a method that takes an `ArrayList` of `Strings` as a parameter and that removes all of the strings of even length from the list.
3. Write a method that takes a `Set` of strings as a parameter and that removes all of the strings of even length from the set. For example, if your method is passed a set containing the following elements:

```
["foo", "buzz", "bar", "fork", "bort", "spoon", "!", "dude"]
```

Your method should modify the set to store the following elements (the order of the elements does not matter):

```
["foo", "bar", "spoon", "!"]
```

4. Write a method that takes an `ArrayList` of `String`s as a parameter and that replaces every string with two of that string. For example, if the list stores the values `{"how", "are", "you?"}` before the method is called, it should store the values `{"how", "how", "are", "are", "you?", "you?"}` after the method finishes executing.

5. Write a method that takes as a parameter a sorted `ArrayList` of `String`s and that eliminates any duplicates from the list. For example, suppose that a variable called `list` contains the following values: `{"be", "be", "is", "not", "or", "question", "that", "the", "to", "to"}` After calling `the method` the list should store the following values: `{"be", "is", "not", "or", "question", "that", "the", "to"}`

6. Write a method that accepts a `List` of strings as a parameter and returns `true` if any single string occurs at least 3 times in the list, and `false` otherwise. Use a map as auxiliary storage.