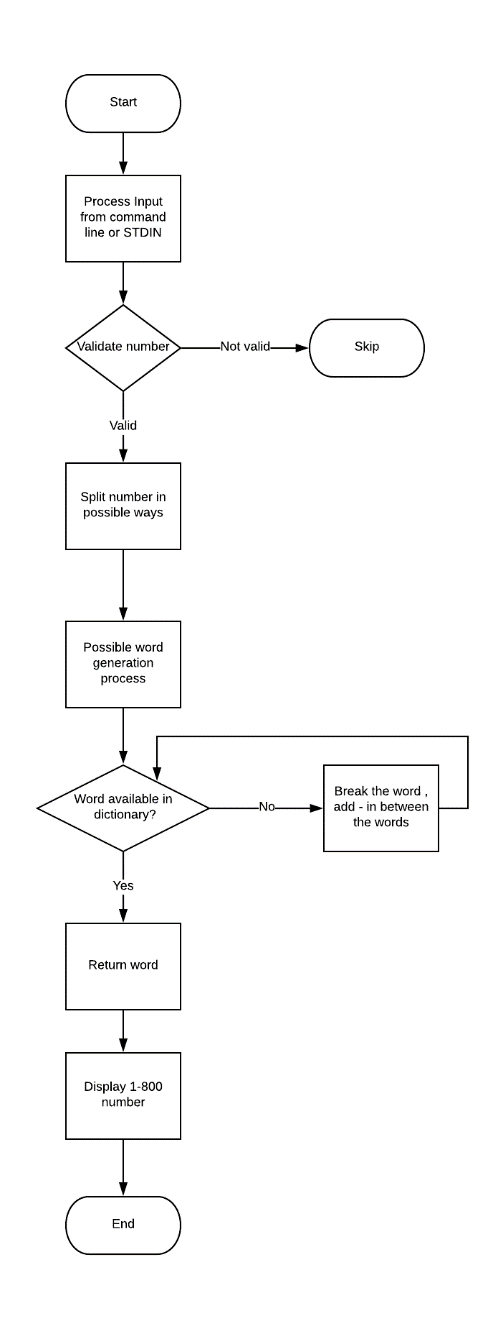
**Why I have selected the 1-800 number coding challenge?**

After reading about both the challenges, I clearly understood the requirement of 1-800 at first go. So, I decided to go with this challenge as expectations were clearly defined to me.

**Design & Approach:**



Steps:

1. Read the input from file or STDIN
2. Split the phone number into all possible ways
3. Generate the words for all the possibility
4. Check in the dictionary for available words
5. Print the 1-800 number

As shown in above flow diagram, program will read an input from STDIN or command line arguments.

Just to keep it simple, I have assumed that if user wants to provide the dictionary file, than it should be the first argument. After that user can add as many phone number files as possible.

If user is not providing the arguments than program will read from STDIN until END is not detected.

Dictionary – This is a static set as it is going to be used throughout the program

Input will be processed using Stream API individually.

I have used OOP based layout, Stream API of Java8 and recursive function for splitting the number in possible ways. I have used Logger to display logs because program should not reply on third party library as per requirement. I have used Junit for testing purpose.

**How to execute the challenge?**

1. Main class - com.aconex.oneeighthundred.*Main*.java
2. Run program with command line arguments
   1. Add first argument with –d for custom dictionary
   2. After that all the possible number files(Each file should have single number)
   3. Program will display all the possible number on console
3. Run the program without any arguments
   1. Enter number in single line
   2. At last write END to stop the input
   3. Program will display all the possible number on console
   4. Default dictionary is available in resources folder