Programming Assignment 2: BabylonianSort

Dr. Sharma Thankachan

COP 3503 – Fall 2018

Due Sunday, October 21st, at 11:59pm

1. Deliverable:

A single source file named *BabylonianSort.java* written in Java.

2. Objective:

The objective of this assignment is to demonstrate an application of radix sort using an exotic number system. It will require knowledge of base conversion, and will touch on the concept of exception handling in Java as well.

3. Description:

Numbers can be represented using many different *bases* besides the base-10 decimal system most frequently used today. A few examples include:

- base-2 (binary): Represents computer bits and Boolean logic.
- base-8 (octal): Used in computers for permissions and settings.
- base-16 (hexadecimal): Frequently used to represent color codes succinctly.

The ancient Babylonians of Mesopotamia employed a *base-60* number system called *sexagesimal* for daily use. Base-60 is superior for dealing with fractions and is "hand-countable" like the decimal system.

For this assignment, we will use a modified version of the Babylonian sexagesimal system. Our adaptation maps Arabic decimal numerals to Latin characters, unlike the original version that represented numbers using cuneiform. The Babylonians also had no way to represent zero, but our adaptation will include zero. The following tables map each Arabic decimal numeral to its Latin character equivalent. The top row of each table is the sexagesimal representation of a numeral, and the bottom row is the decimal representation.

0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	g	h	i	j
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	A	B	C	D
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39

E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59

Notice that this adapted system is case-sensitive. Below are some examples of numbers written in both forms.

Sexagesimal Numeral	Decimal Numeral
9	9
R	53
3X	239
gg	976
gG	1,002
qp	1,585
2aX	7,859
AAAz	7,907,795

4. Requirements:

This assignment must be named *BabylonianSort.java* and contain a public class named *BabylonianSort*. Within this public class you must write these 4 methods:

```
public static void babylonianSort(String[] numbers) throws NumberFormatException
```

This method will take an array of strings as input and must sort this array *in place.* Each string in the array represents a sexagesimal number (see section 5 for specifications).

NOTE: Sorting an array *in place* means that the function returns nothing—instead the original parameter array is changed.

NOTE: You **must** use radix sort to sort the sexagesimal numbers. If you use a different sorting method, you will fail all test cases associated with this function.

NOTE: You may **not** convert the sexagesimal numbers to decimal and sort them that way. The numbers passed will be far too large to store as integers or longs anyway, so don't waste time attempting this.

```
public static String decimalToSexagesimal(long number)
```

Return a string representing the sexagesimal version of the given long decimal number. See section 5 for input specifications for this function.

```
public static long sexagesimalToDecimal(String number) throws NumberFormatException,
                                                              ArithmeticException
```

Return a long integer representing the decimal version of the given sexagesimal number stored in the input string. This function must throw the error *NumberFormatException* if the given sexagesimal number is invalid (*e.g.* it contains characters not enumerated in the description of the sexagesimal number system as described in section 3). This function must throw the error *ArithmeticException* if the given number is too large to be converted to a long.

NOTE: The contents of the error message for both exceptions do not matter, but they should ideally describe the error that has occurred. Here is an example throw of the *NumberFormatException:*

```
throw new NumberFormatException(“This message can say anything”);
```

```
public static boolean isValidSexagesimalNumber(String number)
```

This function will return true if a given sexagesimal number string is a valid sexagesimal number, else return false. The number is only valid if it matches the description of sexagesimal numbers given in the description (section 3).

5. Input Specification:

Each function for this assignment has a slightly different set of input specifications.

babylonianSort(String[] numbers):

This function will receive an array of strings, containing at least one string. Each string will contain at least one character. Each string may contain any valid ASCII character. It is crucial that you verify that each string in the array is a valid sexagesimal number, as defined in section 3, before proceeding with the function.

decimalToSexagesimal(long number):

This function will be passed a long in the range of [0, *Long.MAX_VALUE*].

sexagesimalToDecimal(String number):

A string containing one character or more will be passed to this function. All characters in the string will be valid ASCII characters. This function should throw a *NumberFormatException* if the given string is not a valid sexagesimal number. The string is of arbitrary length, so it's decimal representation could cause an overflow, even for long integers. If the string cannot be represented as a long integer, throw an *ArithmeticException.*

isValidSexagesimalNumber(String number):

This function will receive a string with at least one character. The characters in the string can be any valid ASCII characters. All sexagesimal numbers are non-negative, and this holds for all functions (if you ever find a negative sign in a string, that string is not a valid sexagesimal number). Additionally, strings that contain leading zeros (e.g. "009EE") are not valid sexagesimal numbers.

6. Testing:

To test, you will want to run the provided test script in a Linux environment. Your assignment will be graded on Eustis, and if your assignment does not work on Eustis then it **does not work.** For the script to function, you will need to set up your directory (folder) to look like this:

```
TestSuite/
        BabylonianSort.java
        BabylonianSortTester.java
        Test.sh
        Inputs/
                Input1.txt
                Input2.txt
                ...
```

To run the test script, use this command:

```
$ bash Test.sh
```

NOTE: The "$" character is not part of the command. It is a symbol frequently used to indicate the beginning of a terminal command.

7. Grading Criteria:

There are 36 tests for this assignment and each test is weighed equally. The final assignment score will be out of 100. Here is how the tests are divided:

- ***5/36*** *decimalToSexagesimal()* unit tests
- ***7/36*** *sexagesimalToDecimal()* unit tests
- ***9/36*** *isValidSexagesimalNumber()* unit tests
- ***15/36*** *babylonianSort()* unit tests

8. Final Notes:

Comments will not be graded generally, however I reserve the right to penalize assignments that include no comments, or whose comments are horrible. In the real world, comments are extremely useful and expected from developers, so get used to writing them. I also reserve the right to penalize assignments with atrocious/inconsistent style (or lack thereof). This does not mean that you will lose points for using one particular style over another—it only means that you need to apply your style consistently.

You may work on this assignment with a group of up to 2 people. **Only one member of the group must submit the group's solution, with the names and NIDs of all group members in a comment at the top.** You cannot review or use code from peers outside of your group, but you may have high-level discussions about this assignment with any of your peers.

Examples of **acceptable** peer questions about this assignment look like this:

- How do you convert a number in base-10 to base-60 on paper?
- What data structures are most useful when implementing radix sort?
- Can you show me how to use Horner's Rule to calculate the base-10 version of a base-60 number on paper?

Examples of **unacceptable** peer questions about this assignment look like this:

- What does the code look like to convert a number from base-10 to base-60?
- I have an off-by-one error in my radix sort code, can you help me find the bug?
- Can you show me how to code up Horner's Rule?

You MUST do all of the following:

- Include your name(s) and NID(s) in a comment at the top of your submission.
- Ensure that your submission compiles and runs on Eustis.
- Avoid including your single source file in a package.
- Name your submission *BabylonianSort.java.*
- Submit the assignment before the deadline.
- Write the assignment in Java.

If you do not follow the guidelines above you will receive a 0.