Sample Design Document

# Purpose of the software

This software package, named numberSpellingTools.py, allows a user to generate the English spelling of any whole number from 0 up to 1,000,000,000.

For example, “23607” 🡪 “Twenty three thousand six hundred seven”

# Instructions for using the software

1. Open a new Python document and save it in the same directory as numberSpellingTools.py.
2. In the new Python document:
   1. Import the package numberSpellingTools
   2. Call the function spellAnyNumber on any number up to a billion.   
      The number must be in quotes and without commas. For example,

spellAnyNumber( “2000601” ) returns “two million six hundred and one”

# Functions and procedures

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Arguments** | **What it returns or outputs** | **Team member responsible for coding it** |
| **spellOneDigitNumber** | oneDigitNum | Returns the spelling of the one-digit number, oneDigitNum | Hal |
| **spellTensDigit** | tensDigit | Returns the spelling of the 2-digit number that begins with tens digit tensDigit | Hal |
| **spellTeenNumber** | onesDigit | Returns the spelling of the number between  10-19 that has ones digit onesDigit | Maria |
| **spellTwoDigitNumber** | twoDigitNum | Returns the spelling of the 2-digit number, twoDigitNum.   It uses spellOneDigitNumber, spellTensDigit, and spellTeenNumber. | Maria |
| **spellThreeDigitNumber** | threeDigitNum | Returns the spelling of the 3-digit number, threeDigitNum.   It uses spellOneDigitNumber and spellTwoDigitNumber | Richie |
| **spellAnyNumber** | num | Returns the spelling of any number.  It uses spellThreeDigitNumber. | Tamara |

# Test cases

To ensure that the software works correctly, we will test it on these examples:

|  |  |  |
| --- | --- | --- |
| **Input** | **Expected output** | **Working yet?** |
| “7” | “seven” | Yes |
| “47” | “forty seven” | Yes |
| “17” | “seventeen” | Yes |
| “243” | “two hundred forty three” | Yes |
| “207” | “two hundred seven” | Yes |
| “300” | “three hundred” | No – it’s printing  *“*three hundred*”* but then gives an error |
| “3,450” | “three thousand four hundred fifty | Not yet done |
| “17,000” | “seventeen thousand” | Not yet done |
| “459,006” | “four hundred fifty nine thousand six” | Not yet done |
| “5,000,000” | “five million” | Not yet done |
| 511,050,300” | “five hundred eleven million, fifty thousand three hundred” | Not yet done |
| “1,000,000,000” | “one billion” | Not yet done |