Main function User Interface () {

Get user input for input country, output country, output file name, output file location, input file name and input file location

Create input file and output file by concatenating corresponding file location and file name.

Scan input file

Create an array list called file lines

While input file has a new line {

Add line into file lines array list

}

While file lines array list has another value {

Add all the file line values to one string input text while spacing them apart with a common delimiter

}

Assign returned values to the return of the function Localise (input country, output country, input text)

Split returned values into array called Localised values using the common delimiter

Find the number of dates value

Add all the date values into an array called date values

Scan through the input text with a date regular expression

While regular expression finds a match {

Replace all old values with new values from date values array

}

Add the tail to the end of the string buffer

Convert string buffer to new input text string

Find the number of currencies value

Add all the currency values into an array called currency values

Scan through the input text with a currency regular expression

While regular expression finds a match {

Replace all old values with new values from currency values array

}

Add the tail to the end of the string buffer

Convert string buffer to final text

Split final text using common delimiter into an array called output text

Create file using output file variable

While output text has a value {

Write each line into output file

}

}

Function Localise (input country, output country and input text) {

Scan through the input text with a date regular expression

While regular expression finds a match {

Find the date and set as a string called date text

Assign local date to the return value of the function Localise date (input country, output country, date text)

Increment number of dates by one

Add local date to localised date value spaced with common delimiter

}

Add number of dates to front of localised date value spaced with common delimiter

Scan through the input text with a currency regular expression

While regular expression finds a match {

Find the currency and set as a string called currency text

Remove any characters to make a double value

Assign local currency to the return value of the function Localise currency (input country, output country, currency text)

Add one to number of currencies

Add local currency to localised currency value spaced with common delimiter

}

Add number of currencies to front of localised currency value spaced with common delimiter

Create localised values by concatenating both localised date and localised currency spaced by a common delimiter

Return localised values

}

Function Localise Date (input format, output format, input text) {

Find delimiter of the input text

Split the input text using the delimiter into an array called date values

Load date format into a map

Retrieve input and output country date format using the load date format map

Find the delimiter of input format fields and split input format fields into an array called input format fields using the delimiters

Find the delimiter of output format fields and split output format fields into an array called output format fields using the delimiters

Create a new map called conversion and assign the corresponding values from date values array to the input format fields

If (output format [year value] = input format [year value]) {}

Else If input format [year value] == “yy” {

Add “20” to the begin of the year value

}

Else {

Substring the year value for the last 2 digits

}

Concatenate all the fields together using the output format fields order as the key calling from the conversion map while putting the correct delimiter between the values.

Return localised date

}

Function Localise Currency (input format, output format, input text) {

Load exchange rate into a map

Load location of symbol into a map

Load type of symbol into a map

Create a key by concatenating input format + “-“ + output format

Get exchange rate from map using key

Assign final value to exchange rate \* input text

Round final value to 2 decimal places

Convert final value to a string

Add in the comma or dots in the right locations by inserting them into the final value string

Add the correct symbol in the correct location using the symbol and location maps and the output format to the final value string

Return final value

}