Introduction to Scientific Computation, Homework #2

Due by 12pm on Thursday 5/12/2021

Problem 1 [20]: If a regular fixed payment P is made n times a year to repay a loan of amount A over a period of k years, where the nominal annual interest rate is r, P is given by

$$\mathrm{P} = \frac{rA(1+r/n)^{nk}}{n[\left(1+\frac{r}{n}\right)^{nk}-1]}$$

You need to generate a table of repayments for a loan of $1000 over 15, 20, and 25 years, at interest rates that vary from 10% to 20% per annum in steps of 1%. (Tip: The table has 11 rows and 3 columns, with rows representing different interest rates and columns representing different years, n = 12). You need to use two different methods, the first is to use nested `for` loops, and the second is to vectorize the outer loop. You might use `repmat()`. The generated table is as follows:

```
10.7461    9.6502    9.0870
11.3660   10.3219    9.8011
12.0017   11.0109   10.5322
12.6524   11.7158   11.2784
13.3174   12.4352   12.0376
13.9959   13.1679   12.8083
14.6870   13.9126   13.5889
15.3900   14.6680   14.3780
16.1042   15.4331   15.1743
16.8288   16.2068   15.9768
17.5630   16.9882   16.7845
```

Problem 2 [20]: Narcissistic number is a three-digit number, the sum of the third power of the digits in each bit of which is equal to itself (for example, 153 = 1^3 + 5^3 + 3^3). You may find more information by searching " Narcissistic number ". You need to write two functions, one for finding all the narcissistic numbers and one for determining whether a three-digit number is a narcissistic number. The second function should be nested within the first function.

Problem 3 [20]: Consider the following 3 x 3 matrix:

```
A =
      1      3      2
      8      4      6
      7      9      5
```

Use array subscripts or MATLAB functions to create the following arrays from A – no decisions, loops, etc. Each array should be created using only a single statement.

i) `B =`

```
     1     3
     7     9
```

ii) `C =`

```
     1     8
     7     3
```

iii) `D =`

```
     1     3     2
     8     4     6
```

iv) Find the maximum value of each row

```
E =
     3
     8
     9
```

v) `F =`

```
     1     0     8     0     7
     0     0     0     0     0
     3     0     4     0     9
     0     0     0     0     0
     2     0     6     0     5
```

Problem 4 [20]: Around 300 B.C. Euclid developed a wonderfully simple algorithm for determining the greatest common divisor of two positive integers. Please search online for "Euclidean Algorithm" to find out the details. Your job is to implement the following two functions:

1) `function out=my_gcd(a,b)`
   which computes the greatest common divisor of two positive integers *a* and *b*. You may find the MATLAB function `rem()` useful.
2) `function out=my_lcm(a,b)`
   which computes the least common multiple of two positive integers a and b. This can be computed in part using `my_gcd`.

Make sure you account for the case where a and/or b is zero in your `my_gcd` function. You can check this behavior by entering `gcd(0,2)`, `gcd(2,0)`, and `gcd(0,0)` into the MATLAB command line. Note that the least common multiple of `a=0` or `b=0` is not defined (only least common multiples of `a` and `b` larger than 0 should be returned).

You can compare your implementations to MATLAB's built-in gcd and lcm functions to determine if everything is working correctly.

Problem 5 [20]: Given a string of ‘1’ and ‘0’, find the maximum number of consecutive times of ‘1’. Write a function to do this. Your function declaration should be as follows.

```
function y = lengthOnes(x)
```

The sample is:

x = ‘110100111’
y = 3