CS100M Section Exercise 13

1 Merge Sort

The code for `mergeSort` and the function header for `merge` are shown below. Trace the execution of the script

```
a= [4 1 6 3 2 9 5 7 6 0];
b= mergeSort(a);
```

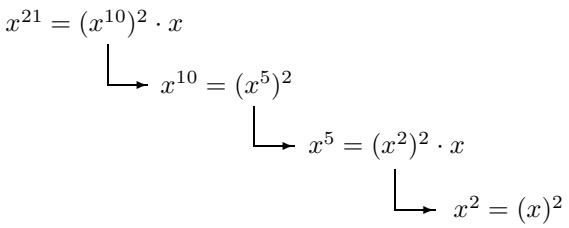
For each call of the `mergeSort` and `merge`, write down the arguments that are passed to the function and the values that are returned. The code below displays the values in vectors `y1` and `y2` and they are the values returned by specific calls to `mergeSort`. Notice that multiple instances of the same function may be open at one time—which function has this behavior, `mergeSort` or `merge`? Ask your section instructor if you have any questions!

```
function y = mergeSort(x)
% x is a vector.
% y is a vector consisting of the values in x sorted from
%   smallest to largest.

n = length(x);
if n==1
    y = x;
else
    m  = floor(n/2);
    % Sort the first half..
    y1 = mergeSort(x(1:m))    % values displayed are the values returned by this call of mergeSort
    % Sort the second half...
    y2 = mergeSort(x(m+1:n))  % values displayed are the values returned by this call of mergeSort
    % Merge...
    y  = merge(y1,y2)         % values displayed are the values returned by this call of merge
end
```

```
function z = merge(x,y)
% x is a row n-vector with x(1) <= x(2) <= ... <= x(n)
% y is a row m-vector with y(1) <= y(2) <= ... <= y(m)
% z is a row (m+n)-vector comprised of all the values in x and
%   y and sorted so that z(1) <= ... <= z(m+n)

n = length(x);  m = length(y);  z = zeros(1,n+m);
ix = 1;   % The index of the next x-value to select.
iy = 1;   % The index of the next y-value to select.
for iz=1:(n+m)
   % Deteremin the iz-th value  for the merged array...
   if ix > n
   % All done with x-values. Select the next y-value.
      z(iz) = y(iy);  iy = iy+1;
   elseif iy>m
   % All done with y-values. Select the next x-value.
      z(iz) = x(ix);  ix = ix + 1;
   elseif x(ix) <= y(iy)
   % The next x-value is less than or equal to the next y-value
      z(iz) = x(ix);  ix = ix + 1;
   else
   % The next y-value is less than the next x-value
      z(iz) = y(iy);  iy = iy + 1;
   end
end
```

2 Efficient calculation of x^n where n is large

If you cannot use MATLAB's power operator ^ how would you calculate x to the n-th power? One way is to use iteration—a loop that executes $n-1$ times. Another strategy is recursion—repeated squaring in this case. The idea is illustrated with the following schematic that shows how to compute x^{21}:



The recursive definition behind the scenes is given by

$$f(x,n) = \begin{cases} 1 & \text{if } n = 0 \\ f(x, n/2) \cdot f(x, n/2) & \text{if } n > 0 \text{ and } n \text{ is even} \\ f(x, (n-1)/2) \cdot f(x, (n-1)/2) \cdot x & \text{if } n > 0 \text{ and } n \text{ is odd} \end{cases} .$$

Write the following function based on the *recursive* strategy. Do not use loops.

```
function y = Power(x, n)
% y = x^n where n is an integer >=0
```