HW12: Please write 7 Octave scripts for the following problems and name these scripts as HW12-1.m, HW12-2.m, ... HW12-7.m, respectively, and zip them into a zip file and upload to BlackBoard's HW 12 submission locker.

HW12-1: Problem Description

Compose a function isOdd which accepts a number n as input and returns true if the number is odd or false is the number is even.

HW12-2: Problem Description

Compose a function isRight which accepts three positive numbers a, b, c (where c is the largest number). isRight should return true if the triangle with sides a, b and c is right-angled. Otherwise, it should return false. You may use any valid mathematical means to determine if the triangle is right-angled.

HW12-3: Problem Description

Compose a function everyOther which accepts a vector and returns a vector containing only every other element of the argument vector. That is, it returns the all odd-numbered elements, starting with the first.

Examples

Input

```
x = [1 3 2 4 3 5]
```
Output

```
y = [1 2 3]
```
Input

```
x = [5 9 3 2 2 0 -1]
```
Output

```
y = [5 3 2 -1]
```

HW12-4: Problem Description

As you no doubt have drilled into perfection at this point, times tables are memorized charts of basic number multiplications.

Compose a function timestable which accepts a number n and returns an n x n times table.

Example

Input

```
    n = 5;
    T = timestable( n );
```
Output

```
    T = [ 1  2  3  4  5
          2  4  6  8 10
          3  6  9 12 15
          4  8 12 16 20
          5 10 15 20 25 ]
```

HW12-5: Problem Description

Given an input array of strings (characters) s, pick out the second column of data and convert it into a column vector of data. Missing data will be indicated by the number 9999. If you encounter missing data, you should set it to the average of the neighboring values.

If the input array is

```
 s = [ 'A' '0096' ;
       'B' '0114' ;
       'C' '9999' ;
       'D' '0105' ;
       'E' '0112'  ];
```
then the output variable `t` is the following column vector.

```
 t = [96 114 109.5 105 112]';
```
Compose a function read_and_interp which accepts an array in the above format and returns the data converted as specified. You may find the conversion function `str2num` to be useful.

HW12-6: Problem Description

Recall that DNA sequences may be written as text strings consisting of the letters `A`, `C`, `T`, and `G`. We would like to count the number of times that letter combinations of a certain length occur in a string. This is useful because, for instance, a three-letter RNA codon could begin at any point in the sequence.

For instance, given the string 'ACTAGG', we can identify the following five 2-grams: 'AC', 'CT', 'TA', 'AG', 'GG' or the following four 3-grams: 'ACT', 'CTA', 'TAG', 'AGG'. Given the string 'AACAAT', we identify the following four 2-grams: 'AA', 'AC', 'CA', AT', and the first 2-gram 'AA' has a corresponding frequency of two instead of one.

Compose a script which assumes at the beginning a string `dna` and a number of n-grams to count `n`. This script then counts the number of n-grams which occur of that length. n-grams should be listed as rows in an array `ngrams`, while the corresponding frequencies should be stored in a column vector `freqs`. You'll thus need to:

loop over the string and find n-letter strings.
check if the n-gram is in the array `ngrams` already; if so, increment its frequency in `freqs`.

else add it to the array `ngrams` and to `freqs` as 1.
after looping, identify the highest-frequency n-gram as `higram`. If two match, select either one.
Your script should thus have three key output variables: `ngrams`, `freqs` (in corresponding order), and `higram`. The strings `dna` and integer `n` should not be defined in the script. The order of the n-grams in `ngrams` should be in the order they first occur in the string `dna`.

HW12-7: Problem Description

Provide a function which calculates the so-called R^2 value, an assessment of how well a model fits data points.

Compose a function r_sqr which accepts an array of polynomial coefficients and an array of data points and returns the R^2 value for the model and data set.

Several intermediate values are necessary to calculate R^2:

`resid`, an array containing the difference between the calculated values at the data x-points and the actual values (resid = ymodel-y)
`svar`, an array containing the variation from the mean of the data (svar = y-mean(y))
`sstot = svar' * svar`, the sum of variations squared
`sserr = resid' * resid`, the sum of errors squared
`R2 = 1 - sserr/sstot`, the R^2 value