```
% Filename: ass1_210100118.m


function ass1_210100118()

    clc;
    ns = [5, 10, 20, 50, 100];
    lu_times = zeros(size(ns));
    inv_times = zeros(size(ns));

    for i = 1:length(ns)
        n = ns(i);

        % Load the NPY file
        path = sprintf('random/matrix_%d.npy', n);
        A = readNPY(path);

        A = double(A);

        % Time the LU decomposition
        tic;
        [L, U, P] = lu(A);
        lu_times(i) = toc * 1e6; % Convert seconds to microseconds

        % Time the matrix inversion
        tic;
        A_inv = inv(A);
        inv_times(i) = toc * 1e6; % Convert seconds to microseconds
    end

    lu_times
    inv_times

    % Save the timing results
    save("lu_times.mat", "lu_times");
    save("inv_times.mat", "inv_times");

end


function [arrayShape, dataType, fortranOrder, littleEndian, totalHeaderLength,
 npyVersion] = readNPYheader(filename)
fid = fopen(filename);

% verify that the file exists
if (fid == -1)
    if ~isempty(dir(filename))
        error('Permission denied: %s', filename);
    else
        error('File not found: %s', filename);
    end
```

```
end

try

    dtypesMatlab =
 {'uint8','uint16','uint32','uint64','int8','int16','int32','int64','single','double', 'lo
    dtypesNPY =
 {'u1', 'u2', 'u4', 'u8', 'i1', 'i2', 'i4', 'i8', 'f4', 'f8', 'b1'};


    magicString = fread(fid, [1 6], 'uint8=>uint8');

    if ~all(magicString == [147,78,85,77,80,89])
        error('readNPY:NotNUMPYFile', 'Error: This file does not appear to be
 NUMPY format based on the header.');
    end

    majorVersion = fread(fid, [1 1], 'uint8=>uint8');
    minorVersion = fread(fid, [1 1], 'uint8=>uint8');

    npyVersion = [majorVersion minorVersion];

    headerLength = fread(fid, [1 1], 'uint16=>uint16');

    totalHeaderLength = 10+headerLength;

    arrayFormat = fread(fid, [1 headerLength], 'char=>char');

    % to interpret the array format info, we make some fairly strict
    % assumptions about its format...

    r = regexp(arrayFormat, '''descr''\s*:\s*''(.*?)''', 'tokens');
    if isempty(r)
        error('Couldn''t parse array format: "%s"', arrayFormat);
    end
    dtNPY = r{1}{1};

    littleEndian = ~strcmp(dtNPY(1), '>');

    dataType = dtypesMatlab{strcmp(dtNPY(2:3), dtypesNPY)};

    r = regexp(arrayFormat, '''fortran_order''\s*:\s*(\w+)', 'tokens');
    fortranOrder = strcmp(r{1}{1}, 'True');

    r = regexp(arrayFormat, '''shape''\s*:\s*\((.*?)\)', 'tokens');
    shapeStr = r{1}{1};
    arrayShape = str2num(shapeStr(shapeStr~='L'));


    fclose(fid);

catch me
    fclose(fid);
    rethrow(me);
```

```
end
end



function data = readNPY(filename)
% Function to read NPY files into matlab.
% *** Only reads a subset of all possible NPY files, specifically N-D arrays
 of certain data types.
% See https://github.com/kwikteam/npy-matlab/blob/master/tests/npy.ipynb for
% more.
%

[shape, dataType, fortranOrder, littleEndian, totalHeaderLength, ~] =
 readNPYheader(filename);

if littleEndian
    fid = fopen(filename, 'r', 'l');
else
    fid = fopen(filename, 'r', 'b');
end

try

    [~] = fread(fid, totalHeaderLength, 'uint8');

    % read the data
    data = fread(fid, prod(shape), [dataType '=>' dataType]);

    if length(shape)>1 && ~fortranOrder
        data = reshape(data, shape(end:-1:1));
        data = permute(data, [length(shape):-1:1]);
    elseif length(shape)>1
        data = reshape(data, shape);
    end

    fclose(fid);

catch me
    fclose(fid);
    rethrow(me);
end
end
```

```
lu_times =

  200.6000  213.0000   28.6000   41.3000  131.4000


inv_times =

  192.5000  114.1000   20.3000   51.7000  360.5000
```

Published with MATLAB® R2022a