# Homework2

## Problem 1 Solution

- First we get the function `forLoopSum`

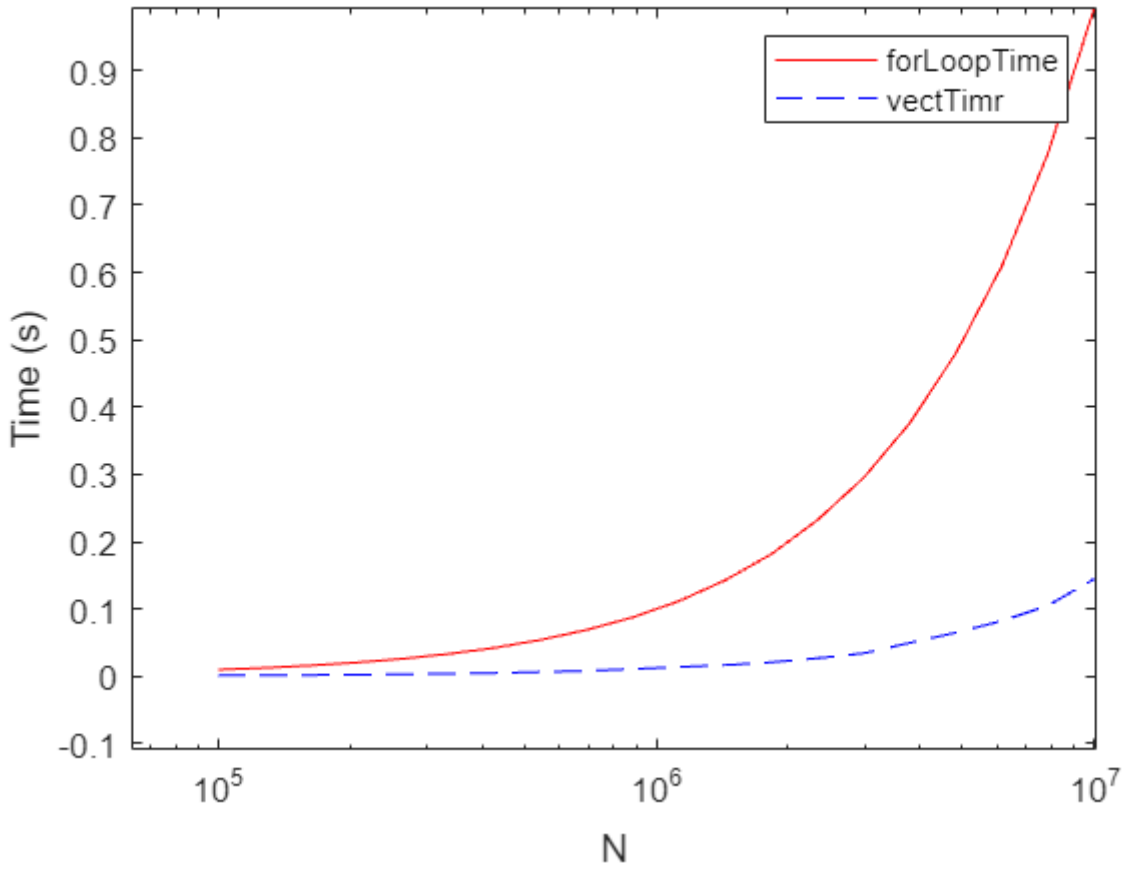
```
function [outSum, computeTime] = forLoopSum(N)
    tic;
    outSum=0;
    for i = 1:N
        outSum = outSum + i^4;
    end
    computeTime = toc;
end
```

- Then we get the function `vectSum`

```
function [outSum, computeTime] = vectSum(N)
    tic;
    outSum = sum((1:N).^4);
    computeTime = toc;
end
```

- Then we get the plot

```
% give the list N
N = round(logspace(5,7,20));

forLoopTime = zeros(size(N));
vectTime = zeros(size(N));
outSum1 = zeros(size(N));
outSum2 = zeros(size(N));
flag=1;

for i = 1:length(N)
    [outSum1(i), forLoopTime(i)] = forLoopSum(N(i));
    [outSum2(i), vectTime(i)] = vectSum(N(i));
end

semilogx(N, forLoopTime, 'r-', N, vectTime, 'b--');
legend('forLoopTime','vectTimr');
disp(outSum1);
disp(outSum2);
```




- Conclusion
  - The efficiency of vectSum is better than forLoopSum
  - the result of the two method is the same

# Problem 2 Solution

### my_gcd

To make it more convenient, I choose `mod` instead of `rem`.

```
function out = my_gcd(a,b)
    while mod(a,b) ~= 0
        t=a;
        a=b;
        b=mod(t,b);
    end
    out = b;
end
```

### my_lcm

In MATLAB's built-in lcm, it will raise a error while a=0 or b=0, So

```
function out = my_lcm(a,b)
    if a==0 || b==0
        out = NaN;
    else
        out = a*b/my_gcd(a,b);
    end
end
```

# Problem 3 Solution

### check

We first write a function to check if it is a perfect number

```
function out = check(N)
    cnt = 0;
    out = 0;
    for i = 1: round(N/2)
        if rem(N,i) == 0
            cnt = cnt + i;
        end
    end
    if cnt == N
        out = 1;
    end
end
```

### perfectNumbers

Then we output the result

```
function out = perfectNumbers(N)
    out = [];
    for i = 2:N
        if check(i) == 1
            out(end+1) = i;
        end
    end
end
```

The output is:

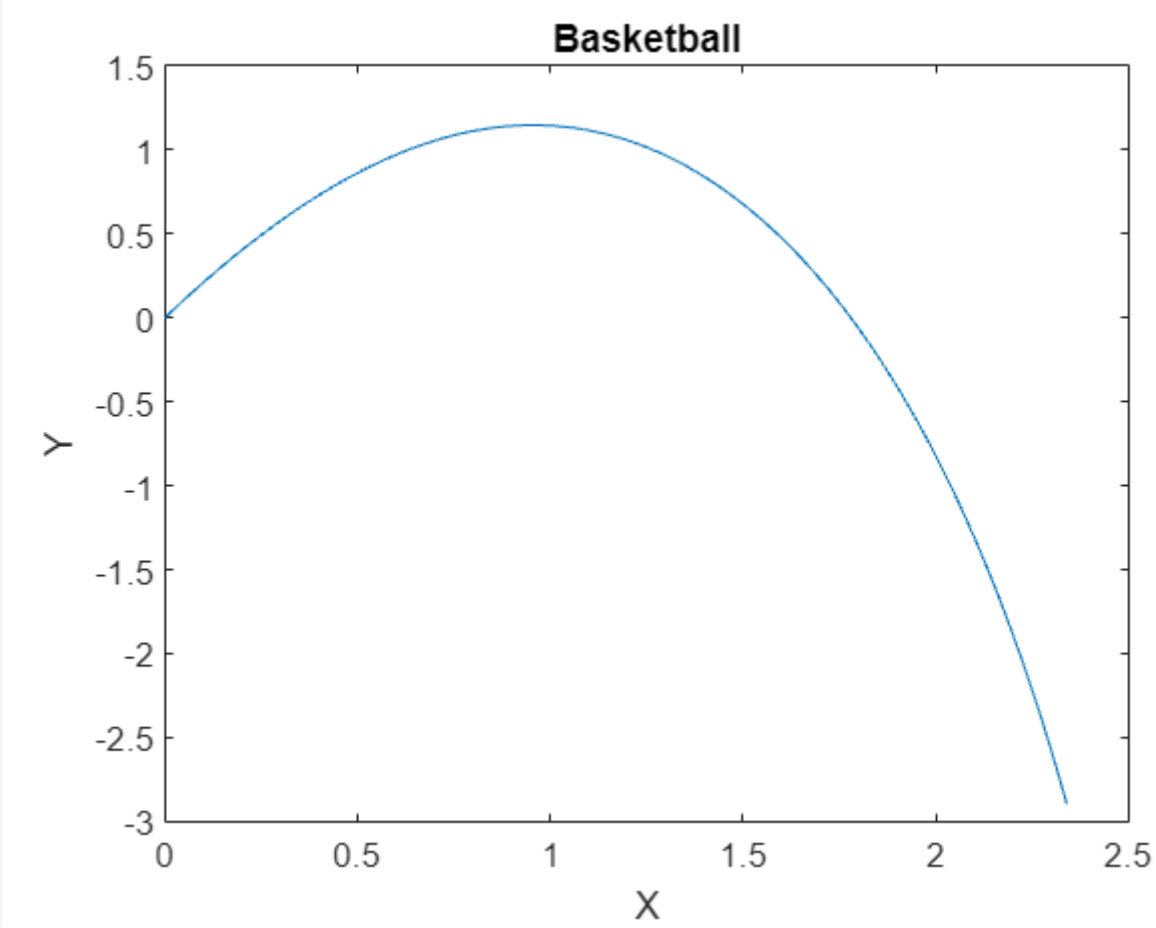
```
>> perfectNumbers(10000)

ans =

           6          28         496        8128
```

## Problem 4 Solution

Just get the function with the help of fomular

```
function [x,y] = basketball(g,c,x0,y0,vx0,vy0,tstep,tmax)
    t = 0:tstep:tmax;
    x = zeros(size(t));
    y = zeros(size(t)); % innitialization

    x(1) = x0;
    y(1) = y0;
    vx = vx0;
    vy = vy0;

    for i = 2:length(t)
        ax = -c * vx * sqrt(vx^2+vy^2);
        ay = -g -c * vy * sqrt(vx^2+vy^2);
        x(i) = x(i-1) + vx * tstep;
        y(i) = y(i-1) + vy * tstep;
        vx = vx + ax * tstep;
        vy = vy + ay * tstep;
    end
    plot(x,y);
    xlabel("X");
    ylabel("Y");
    title("Basketball");
end
```

The plot is:




# Problem 5 Solution

## Method a

It is not difficult so I show the function directly

```
function out = my_arctan(x,N)
    out = 0;
    for i = 1:10^N
        out = out + (-1)^(i+1)* x^(2*i-1) / (2*i-1);
    end
end
```

The result is:

```
>> pi = vpa(my_arctan(1,3) * 4 , 8)

pi =

3.1405927

>> pi = vpa(my_arctan(1,4) * 4 , 8)

pi =

3.1414927

>> pi = vpa(my_arctan(1,5) * 4 , 8)

pi =

3.1415827

>> pi = vpa(my_arctan(1,6) * 4 , 8)

pi =

3.1415917

>> pi = vpa(my_arctan(1,7) * 4 , 8)

pi =

3.1415926
```

### Method b

Still easy

```
function out = my_pi(N)
    out = 0;
    for i = 1:10^N
        out = out + 1/(4*i-3)/(4*i-1);
    end
    out = out * 8;
end
```

The result is:

```
>> pi = vpa(my_pi(4),8)

pi =

3.1415427

>> pi = vpa(my_pi(5),8)

pi =

3.1415877

>> pi = vpa(my_pi(6),8)

pi =

3.1415922

>> pi = vpa(my_pi(7),8)

pi =

3.1415926
```

We find that we need fewer terms to reach the same level of accuracy that we got in (a)

### Calculate $\pi$

Call the function we write in (a):

```
>> pi = (6*my_arctan(1/8,7)+2*my_arctan(1/57,7)+my_arctan(1/239,7))*4;
>> vpa(pi,8)

ans =

3.1415927
```

So we get the answer.

## Problem 6 Solution

So silly

```
A=[1 2 3;4 5 6;7 8 9];
B = [A(:)]'
C = A(:,1:2)
D = A(1:2,:)
E = [A(1,1) 0 A(1,2) 0 A(1,3);
     0 0 0 0 0;
```

```
      A(2,1) 0 A(2,2) 0 A(2,3);
      0 0 0 0 0;
      A(3,1) 0 A(3,2) 0 A(3,3)]'
F = [A(3,3) 0 A(3,2) 0 A(3,1);
      0 0 0 0 0;
      A(2,3) 0 A(2,2) 0 A(2,1);
      0 0 0 0 0;
      A(1,3) 0 A(1,2) 0 A(1,1)]
```

The result:

```
B =

     1     4     7     2     5     8     3     6     9


C =

     1     2
     4     5
     7     8


D =

     1     2     3
     4     5     6


E =

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


F =

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

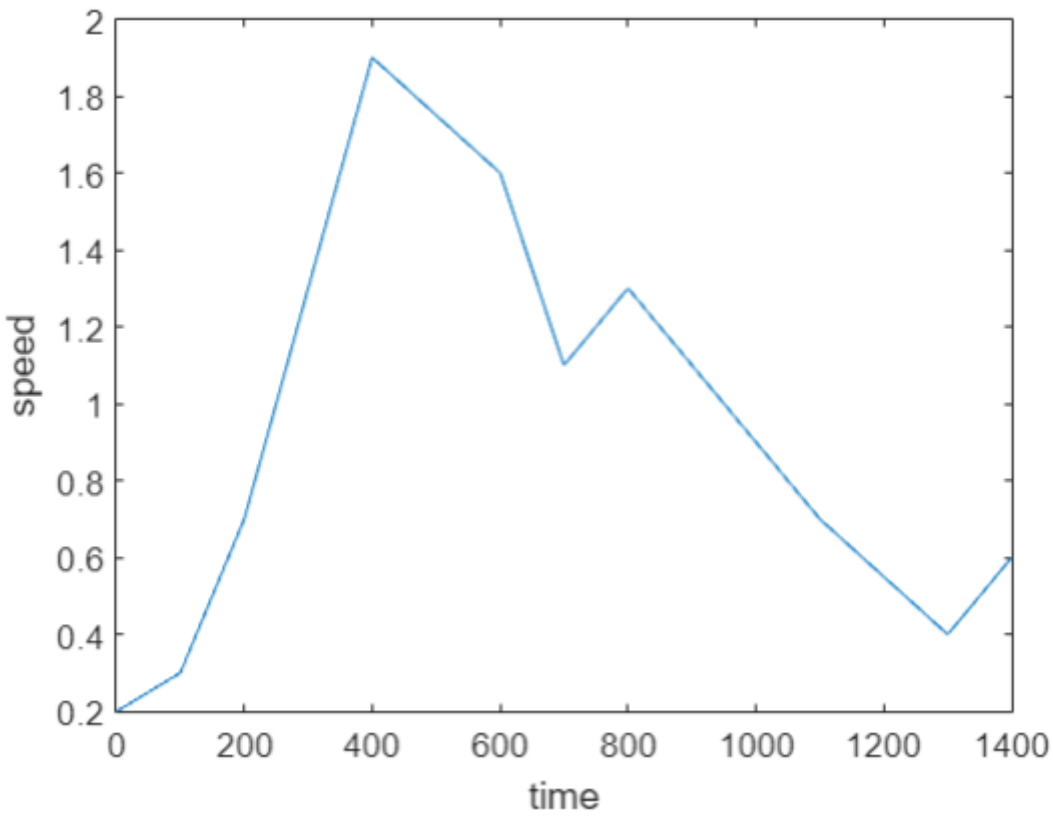
# Problem 7 Solution

It seems that no need to explain too much

```
speed = [0.2, 0.3, 0.7, 1.3, 1.9, 255, 1.6, 1.1, 1.3, 255, 0.9, 0.7, 255, 0.4,
0.6];
time = [0, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200, 1300,
1400];

lessThan05 = numel(speed(speed<0.5))
lessTime = time(speed<0.5)
from350To1050 = speed(350<=time & time<=1050)
tt = logical(speed==255);
speed(tt) = NaN;

% flit the NaN
time = time(~isnan(speed));
speed = speed(~isnan(speed));
plot(time,speed);
xlabel("time");
ylabel("speed");
```

The result is:

```
lessThan05 =

     3


lessTime =

           0         100        1300


from350To1050 =

    1.9000  255.0000    1.6000    1.1000    1.3000  255.0000    0.9000
```




I flit the erroneous measurements to make the plot more clear.

## Problem 8 Solution

With the rule, we get:

```
use = [2,5,7,10,15]*100;
cost = zeros(1,5);

tt = use<=500;
cost(tt) = use(tt) * 2;

tt = use>500 & use<=1000;
cost(tt) = 1000 + (use(tt)-500) * 5;
```

```
tt = use>1000;
cost(tt) = 3500 + (use(tt)-1000) * 10;

cost = cost + 500;

useAndCost = [use' cost']
```

The result is:

```
useAndCost =

         200         900
         500        1500
         700        2500
        1000        4000
        1500        9000
```

# Problem 9 Solution

### flipOddCol

```
function B = flipOddCol(A)
    B = A;
    B(:,1:2:end) = flipud(A(:,1:2:end));
end
```

An example:

```
A =

     1     2     3
     4     5     6
     7     8     9

>>  B = flipOddCol(A)

B =

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

### diagEye

```
function B = diagEye(A)
    di = diag(A);
    di(:) = sum(di);
    B = diag(di);
end
```

An example:

```
A =

     1     2     3
     4     5     6

>> B = diagEye(A)

B =

     6     0
     0     6
```

### colXchange

```
function B = colXchange(A)
    B = A;
    B(:,1:2:end-1) = A(:,2:2:end);
    B(:,2:2:end) = A(:,1:2:end-1);
end
```

An example:

```
A =

     1     2     3     4     5
     6     7     8     9    10

>> B = colXchange(A)

B =

     2     1     4     3     5
     7     6     9     8    10
```

### stripes

```
function B = stripes(n)
    B = zeros(n,n);
    B(2:2:end,:) = 1;
end
```

An example:

```
>> B = stripes(10)

B =

     0     0     0     0     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
```

### threshold

```
function B = threshold(A,lob,hib)
    B = A;
    B(A<lob) = lob;
    B(A>hib) = hib;
end
```

An example:

```
A =

     1     2     3
     4     5     6
     7     8     9

>> B = threshold(A, 3, 7)

B =

     3     3     3
     4     5     6
     7     7     7
```

## Problem 10 Solution

palindrome

```
function res = palindrome(str)
    str = str(isletter(str));
    str = lower(str);
    strr = str(end:-1:1);
    res = all(str == strr);
end
```

Some examples:

```
>> palindrome('Doc, note I dissent: a fast never prevents a fatness. I diet on cod')

ans =

     1

>> palindrome('Never odd or even')

ans =

     1

>> palindrome('A man a plan a canal Panama')

ans =

     1

>> palindrome('I am Iron Man')

ans =

     0
```

## longest palindrome

> Are we not pure? "No, sir!" Panama's moody Noriega brags. "It is garbage!" Irony dooms a man—a prisoner up to new era.

check it:

```
str =

Are we not pure? "No, sir!" Panama's moody Noriega brags. "It is garbage!" Irony dooms a man—a prisoner up to new era.

>> palindrome(str)

ans =

     1
```

So it is a palindrome.