Lab 1 实验报告

3n+1 Problem

（1）首先实现的是这个函数，代码如下。

```
function y=CollatzConjecture(n)
if mod(n,2) == 1
    y=3*n+1;
else
    y=n/2;
end
```

（2）测试从1到10000是否都会满足经历有限多次的循环之后会得到1.代码如下。

```
for j =1:10e4
    k=j;
    while(k~=1)
        k=CollatzConjecture(k);
    end
end

fprintf('done')
```

基本思路是用for循环遍历1到10000，在for循环中再建立一个while循环，当经历循环后变为1时即可跳出while循环。最后如果所有的数都能最后变为1，则会跳出for循环，最后输出done。

```
done
fx >>
```

可以看出最后输出能输出‘done’，即表示所有的数都能归为1.

（3）为了验证每次输出的最后几位是否是4-2-1，在上述代码中添加了一个空向量，并在每次的while循环中记录。最后输出最后三位。代码如下

```
close;
clear;
clc;

for j =1:10e4
    k=j;
    a=[];
    while(k~=1)
        temp=CollatzConjecture(k);
        a(end+1)=temp;
        k=temp;
    end
    if j>=8
        a(end-2:end)
    end
end
```

最后输出的结果如下。

```
ans =

     4     2     1


ans =

     4     2     1


ans =

     4     2     1


ans =

     4     2     1


ans =

     4     2     1
```

（4）用tic和toc测试了（2）中的代码，得到的结果如下。

```
done
历时 0.740288 秒。
>>
```

优化程序的想法是在一些数中包含很多的质因数2的话可以先一直除以2，直到没有2为止。例如$192 = 2^6 \times 3$，则我们只用计算3是否满足即可。

具体实现的代码如下。

```
close;
clear;
clc;

tic
for j =1:10e4
    k=j;
    a=factor(k);
    b=a>2;
    k=sum(b.*a);
    if k~=0
        while(k~=1)
            k=CollatzConjecture(k);
        end
    end
end

fprintf('done\n')
toc
```

先用factor函数实现因数分解，顺序是从小到大，以192为例，则会生成$[2\ 2\ 2\ 2\ 2\ 2\ 3]$。并生成一个逻辑数组，又以192为例，则是$[0\ 0\ 0\ 0\ 0\ 0\ 1]$，让这两个数组点乘并求和则会得到3。注意到像64这种2的次方的数则最终和是0，所以之后判断是否为0即可。

但最后的程序运行时间不尽人意，没有实现程序的优化。代码运行时间如下。

```
done
历时 1.964924 秒。
>>
```

猜测可能的原因是因为质因数分解所需的时间会大大增加程序所需的时间。