Lab5 Riddle

姓名：杨涛
学号：PB20020599

分析

既然是实现函数，那么就需要保存变量，则需要自建一个栈并用R6保存栈指针，之后在主函数调用时对变量进行存储。
于此同时考虑到主函数中有两个运算符（乘法与取模）无法简单实现，则需要额外定义两个函数来进行乘法运算与取模运算。

初号机

由上面的分析很容易得到初始版本，其中JUDGE函数是对原函数的简单翻译，Times和Mod函数是最朴素的乘法与取余算法。

```
.ORIG x3000
LEA R6,STACK
JSR JUDGE
HALT

JUDGE STR R2,R6,#0
ADD R6,R6,#1
STR R7,R6,#0
ADD R6,R6,#1
AND R2,R2,#0
ADD R2,R2,#2;R2=i=2
LOOP0 ADD R1,R2,#0
JSR Times
NOT R1,R1
ADD R1,R1,#1
ADD R1,R0,R1
BRn True
ADD R1,R2,#0
JSR Mod
ADD R1,R1,#0
BRz False
ADD R2,R2,#1
BR LOOP0
True AND R1,R1,#0
ADD R1,R1,#1
BR Finish
False AND R1,R1,#0
Finish ADD R6,R6,#-1
LDR R7,R6,#0
ADD R6,R6,#-1
LDR R2,R6,#0
RET

;input:R1,output:R1=R1^2
Times STR R2,R6,#0
ADD R6,R6,#1
STR R3,R6,#0
ADD R6,R6,#1
ADD R2,R1,#0
ADD R3,R1,#0
AND R1,R1,#0
LOOP1 ADD R1,R1,R2
ADD R3,R3,#-1
BRnp LOOP1
ADD R6,R6,#-1
LDR R3,R6,#0
ADD R6,R6,#-1
LDR R2,R6,#0
RET

;input:R1,R0,output:R1=R0%R1
Mod STR R2,R6,#0
ADD R6,R6,#1
STR R3,R6,#0
```

```
ADD R6,R6,#1
NOT R2,R1
ADD R2,R2,#1
ADD R1,R0,#0
ADD R3,R1,R2
BRn Fin
LOOP2 ADD R1,R1,R2
ADD R3,R1,R2
BRzp LOOP2
Fin ADD R6,R6,#-1
LDR R3,R6,#0
ADD R6,R6,#-1
LDR R2,R6,#0
RET

STACK .BLKW #20
.END
```

量产机

p优化版本：

- Times算法：与lab1的p版本相同，不表
- Mod算法：采用试商法，即小学时代所学习的竖式除法运算，同时因为lc3中无法方便地右移，则需要入栈存储左移过程中的数，出栈作为右移运算。
- 于此同时将部分寄存器改为调用者保存，可以减少部分入栈出栈的指令数。

```
.ORIG x3000
LEA R6,STACK
JSR JUDGE
HALT

JUDGE
;存储所用到的寄存器
JUDGE STR R2,R6,#0
ADD R6,R6,#1
STR R3,R6,#0
ADD R6,R6,#1
STR R4,R6,#0
ADD R6,R6,#1
STR R5,R6,#0
ADD R6,R6,#1
STR R7,R6,#0
ADD R6,R6,#1

;函数主体
AND R3,R3,#0
ADD R3,R3,#2;R3=i=2
LOOP0 ADD R1,R3,#0
ADD R2,R3,#0
JSR Times
NOT R1,R1
ADD R1,R1,#1
ADD R1,R0,R1
BRn True
ADD R1,R3,#0
JSR Mod
ADD R1,R1,#0
BRz False
ADD R3,R3,#1
BR LOOP0
True AND R1,R1,#0
ADD R1,R1,#1
BR Finish
False AND R1,R1,#0

;寄存器归位
Finish ADD R6,R6,#-1
LDR R7,R6,#0
ADD R6,R6,#-1
LDR R5,R6,#0
ADD R6,R6,#-1
LDR R4,R6,#0
ADD R6,R6,#-1
LDR R3,R6,#0
ADD R6,R6,#-1
LDR R2,R6,#0
RET
;JUDGE结束

;Times
```

```
;input:R1,R2,output:R1=R1*R2
;R4，R5为调用者保存
Times STR R3,R6,#0
ADD R6,R6,#1

ADD R3,R1,#0
AND R1,R1,#0
AND R4,R4,#0
ADD R4,R4,x1
LOOP2 AND R5,R2,R4
BRz LOOP1
ADD R1,R1,R3
LOOP1 ADD R3,R3,R3
ADD R4,R4,R4
BRnp LOOP2

ADD R6,R6,#-1
LDR R3,R6,#0
RET
;Times结束

;Mod
;input:R1,R0,output:R1=R0%R1
;R2，R5为调用者保存，R0不变const
Mod NOT R2,R1
ADD R2,R2,#1
AND R1,R1,#0
STR R1,R6,#0;设置结束条件
ADD R6,R6,#1
ADD R1,R0,#0
LOOP3 ADD R5,R2,R0;逐步左移并求差，若出现负数说明已经达到最大
BRn LOOP4
STR R2,R6,#0
ADD R6,R6,#1
ADD R2,R2,R2
BR LOOP3

LOOP4 ADD R6,R6,#-1
LDR R2,R6,#0
BRz Fin
ADD R5,R2,R1
BRn skip
ADD R1,R2,R1
skip BR LOOP4

Fin RET
;Mod结束

STACK .BLKW #40
.END
```

经过webLC3的测算，平均指令数达到原来的五分之一左右，为2681.67。