Rax, rcx, rdx, rsi, rdi, r8

Rax = return Pointer

Rcx = gap

Rdx = height

Rdi = pointer

Rsi = width

R8 = Padded Width // only available… can’t use for padding!

Pushq rdp

Movq rsp, rdp

Movq 3 x length, r8

PADDING

Cmp 0x100, r8

jumpEqual r8, NORMAL

inc r8

jump PADDING // r8 now has tmp length

NORMAL

Movq r8, %rax

Mulq height, %rax

Subq %rax, %rdi // move pointer’s position to very start (rax used as temporary)

Movq %rax, %rdi // set rax to very first, and don’t move this.

Loop:

Movq (%rdi), %rsi

Subq (%rax), %rsi

Cmp length, %rsi //compares width and current width

jumpGreater END // if current with longer, end

push pos, height, width

Callq DrawToLeft //if not longer, draw to left

Leave current pos, height, pointer

Save r8 (

(needs pos, height, pointer)

Callq DrawToRight //and draw to right

(needs pos, height, width, pointer)

Addq %gap, %rdi

END

Pop things

Ret

**DrawToLeft**

Push pointer

Movq rsp, pointer

Movq 0, currHeight

Get pos

Get height

Get pointer

Loop 2:

DRAW

Movq length + pad , %rdi // how? If space is available, have one space for pad (on stack)

Dec %rdi

Dec pos

Inc currHeight

Cmp 0, pos

jumpLess – end // if position is left of zero, end

Cmp height, currHeight

Jmpgreater End // if height exceeds max, end

Jmp loop 1

END

Pop things

ret

**DrawToRight:**

Push pointer

Movq rsp, pointer

Movq 0, currHeight

Get pos

Get height

Get pointer

Loop 1:

Movq length + pad, %rdi // how? If space is available, have one space for pad (on stack)

inc %rdi

inc pos

Inc currHeight // 여기선 그리기 전에 일단 옮김.

Cmp width, pos

jumpGreater – end // if position is greater than width, end

Cmp height, currHeight

Jmpgreater End // if height exceeds max, end

DRAW

Jmp loop 1

END

Pop things

ret

#--------------------------------------------------------------

#

# 4190.308 Computer Architecture (Spring 2019)

#

# Project #3: Drawing diagonal lines in an image

#

# April 24, 2019.

#

# Jin-Soo Kim (jinsoo.kim@snu.ac.kr)

# Systems Software & Architecture Laboratory

# Dept. of Computer Science and Engineering

# Seoul National University

#

#--------------------------------------------------------------

.text

.align 4

.globl bmp\_diag

.type bmp\_diag,@function

bmp\_diag:

#------------------------------------------------------------

# Use %rax, %rcx, %rdx, %rsi, %rdi, and %r8 registers only

# imgptr is in %rdi

# width is in %rsi

# height is in %rdx

# gap is in %rcx

# (0,0) is in %rax

# only r8 is available

#------------------------------------------------------------

# --> FILL HERE <--

pushq %rbx

movq %rsp, %rbx

leaq (%rsi), %r8

imulq $0x12, %r8

PADDING:

cmp $0x4, %r8

je NORMAL

inc %r8

jmp PADDING

NORMAL:

movq %r8, %rax

imul (%rdx), %rax

subq (%rax), %rdi

movq %rax, %rdi

pushq %rax

LOOP1:

movq (%rdi), %r8

subq (%rax), %r8

cmp %rsi, %r8

jg END

pushq %rdi

pushq %rcx

call DrawToLeft

call DrawToRight

addq %rcx, %rdi

END:

ret

DrawToLeft:

pushq %rbx

movq %rsp, %rbx

movq (%r8), %rcx

Loop2:

movb $0x00, (%rdi) # blue

movb $0x00, 1(%rdi) # green

movb $0xff, 2(%rdi) # red

movq 3(%rsi), %rax

PAD:

cmp $0x4, %rax

je NORM

inc %rax #rax used to store padded length

jmp PAD

NORM:

movq %rdi, %rax

subq $0x3, %rdi

subq $0x3, %rcx

cmp $0, %rcx

jl END2 #if reach left wall, end

cmp 20(%rbx),%rdi #rdp = DTL. 4rdp = ret address from call. 8rdp = rcx, 12rdp = rdi, 16rdp = 0,0,

#20rdp = start of image

jl END2 #if reach bottom, end

jmp LOOP2

END2:

popq %rcx

popq %rdi

ret

DrawToRight:

pushq %rbx

movq %rsp, %rbx

movq (%r8), %rcx

PAD2:

cmp $0x4, %rax

je NORM2

inc %rax

jmp PAD2

NORM2:

movq %rdi, %rax

addq $0x3, %rdi

addq $0x3, %rcx

cmp $rsi, %rcx

jl END2

cmp 20(%rbx),%rdi

jl END2

movb $0x00, (%rdi)

movb $0x00, 1(%rdi)

movb $0xff, 2(%rdi)

jmp NORM2