```
.data
Request: .asciiz "Using values between 1 and 255 please do the
following: \n"
input1: .asciiz "\nEnter a value to be multiplied: "
input2: .asciiz "Enter a value to multiply with: "
endl: .asciiz "\n"
results: .asciiz "\nResult: "
.text
main:
addi
                 $sp, $sp, −4
li
                 $s0, 256
                 $s1, 1
li
reinput:
li
                 $v0, 4
                                          ## code to print string
                                          ## loading string to print
la
                 $a0, Request
syscall
li
                 $v0, 4
                                          ## code to print string
                 $a0, input1
la
                                          ## loading string to print
syscall
li
                 $v0, 5
svscall
                 $t1, $v0
move
li
                 $v0, 4
                                          ## code to print string
                 $a0, input2
                                          ## loading string to print
la
syscall
li
                 $v0, 5
syscall
                 $t2, $v0
move
                 $t1, $s0, reinput
bge
                 $t2, $s0, reinput
bge
                 $t1, $0, reinput
ble
                 $t2, $0, reinput
ble
                 $t1, 4($sp)
SW
                 $t2, 8($sp)
SW
jal rmult
li
                 $v0, 4
                                          ## code to print string
                 $a0, results
                                          ## loading string to print
la
syscall
```

```
li
                 $v0, 1
lw
                 $a0, 12($sp)
syscall
                 $v0, 10
li
syscall
rmult:
                                           ## beginning loop 1 (build
stacks)
### construct stack
                 $t1, 4($sp)
                                           ## loads multiplicand
        lw
        lw
                 $t2, 8($sp)
                                           ## loads multiplier
                 $sp, $sp, 16
                                           ## allocates new space
        addi
                 $ra, 0($sp)
                                           ## Return address
        SW
                                           ## multiplicand
                 $t1, 4($sp)
        SW
                 $t0, $t2, $s1
                                           ## decriments multiplier
        sub
                 $t0, 8($sp)
                                           ## multiplier
        SW
### Function Controller
        bne
                 $t2, $s1, deconstruct
                                           ## kill switch
        b exit
deconstruct:
                                           ## deconstruct stack
        jal rmult
                                           ## Loop control (end loop1/
begin loop2)
### End Function Controller
### deconstruct stack
                 $ra, 0($sp)
                                           ## loads return address
        lw
        lw
                 $t1, 4($sp)
                                           ## loads multiplicand
                 $t2, 12($sp)
        lw
                                           ## loads result
        add
                 $t1, $t1, $t2
                                           ## adds result and
multiplicand
exit:
        sub
                 $sp, $sp, 16
                                           ## Deconstructs stack
                 $t1, 12($sp)
                                           ## store result
        SW
                                           ## returns to main or
        jr
                 $ra
deconstruct
```