```
1:  # Program to calculate Ackermann's function such that A(m,n)
2:  # terms "m" and "n" are >= 0, specified by user input (integers).
3:  # prompt user input, call ackman function, print out resulting value
4:
5:  # Written by Kollen G
6:
7:
8:         .data
9:         .align  2
10: prmptM:.asciiz  "\nEnter m value to calculate A(m,n): "
11: prmptN:.asciiz  "\nEnter n value to calculate A(m,n): "
12: result:.asciiz  "\nAckermann's function with M and N = "
13:
14: #-------------------------------
15:         .text
16:         .globl  main
17:
18: main:
19:     move    $s0, $0     # s0 : computed A(m,n) value
20:
21:     # get user input M
22:     la  $a0, prmptM #load prmpt for M
23:     li  $v0, 4      #code to print string
24:     syscall         #print
25:
26:     li  $v0, 5      #take int input
27:     syscall
28:     move    $s1, $v0    # s1 = user input "m"
29:     # get user input N
30:     la  $a0, prmptN #load prmpt for N
31:     li  $v0, 4      #code to print string
32:     syscall         #print
33:
34:     li  $v0, 5      #take int input
35:     syscall
36:     move    $s2, $v0    # s2 = user input "n"
37:     # call Ackman func
38:     move    $a0, $s1
39:     move    $a1, $s2
40:     jal ackman
41:     move    $s0, $v0    # s0 = result from Ackman func
42:
43: #------ Display results and exit ---------------------------------
44:     la  $a0, result #load display string
45:     li  $v0, 4      #code to print string
46:     syscall         #print
47:
48:     li  $v0, 1      #code to print int
49:     move    $a0, $s0    #load computed A(m,n)
50:     syscall         #print
51:
52: #----------------- Exit ---------------------
53:     li  $v0, 10
54:     syscall
55:
```

```
56:
57:
58: #*****************************************************************
59:     # ackman function
60:     #
61:     # a0 - user input "m"
62:     # a1 - user input "n"
63:         #
64:         # v0 - computed A(m,n)
65: ackman:
66: #--------------- Usual stuff at function beginning  ---------------
67:         addi    $sp, $sp, -24
68:         sw  $ra, 20($sp)
69:         sw  $s0, 16($sp)
70:         sw  $s1, 12($sp)
71:         sw  $s2, 8($sp)
72:         sw  $s3, 4($sp)
73:         sw  $s4, 0($sp)
74: #-------------------------- function body -------------------------
75:     move    $s0, $a0        # s0 : "M"
76:     move    $s1, $a1    # s1 : "N"
77:     move    $s2, $0     # s2 : computed A(m,n)
78:
79:         # base case if m = 0
80:         bne     $s0, 0, cont1   # if (M == 0)
81:         addi    $s2, $s1, 1 # s2 = (n+1)
82:
83:         # else if m > 0 and n = 0
84: cont1:  ble $s0, 0, cont2   # if (M > 0)
85:         bne $s1, 0, cont2   # if (N == 0)
86:         addi    $a0, $s0, -1    # a0 : M = (m-1)
87:         addi    $a1, $0, 1  # a1 : N = 1
88:         jal ackman      # compute
89:         move    $s2, $v0    # s2 = A(m-1,1)
90:
91:         # else if m > 0 and n > 0 ----- A(m-1, A(m,n-1))
92: cont2:  ble $s0, 0, done    # if (M > 0)
93:         ble $s1, 0, done    # if (N > 0)
94:         #inner
95:         move    $a0, $s0    # a0 : M = m
96:         addi    $a1, $s1, -1    # a1 : N = (n-1)
97:         jal ackman
98:         move    $a1, $v0    # a1 : N = A(m,n-1)
99:         #outer
100:         addi   $a0, $s0, -1    # a0 : M = (m-1)
101:         jal    ackman      # compute
102:         move   $s2, $v0    # s2 = A(m-1, A(m,n-1))
103:
104: done:  move    $v0, $s2
105:
106: #-------------------- Usual stuff at function end -----------------
107:         lw     $ra, 20($sp)
108:         lw $s0, 16($sp)
109:         lw $s1, 12($sp)
110:         lw $s2, 8($sp)
```

```
111:          lw $s3, 4($sp)
112:          lw $s4, 0($sp)
113:          addi   $sp, $sp, 24
114:          jr      $ra
115:
116:
```