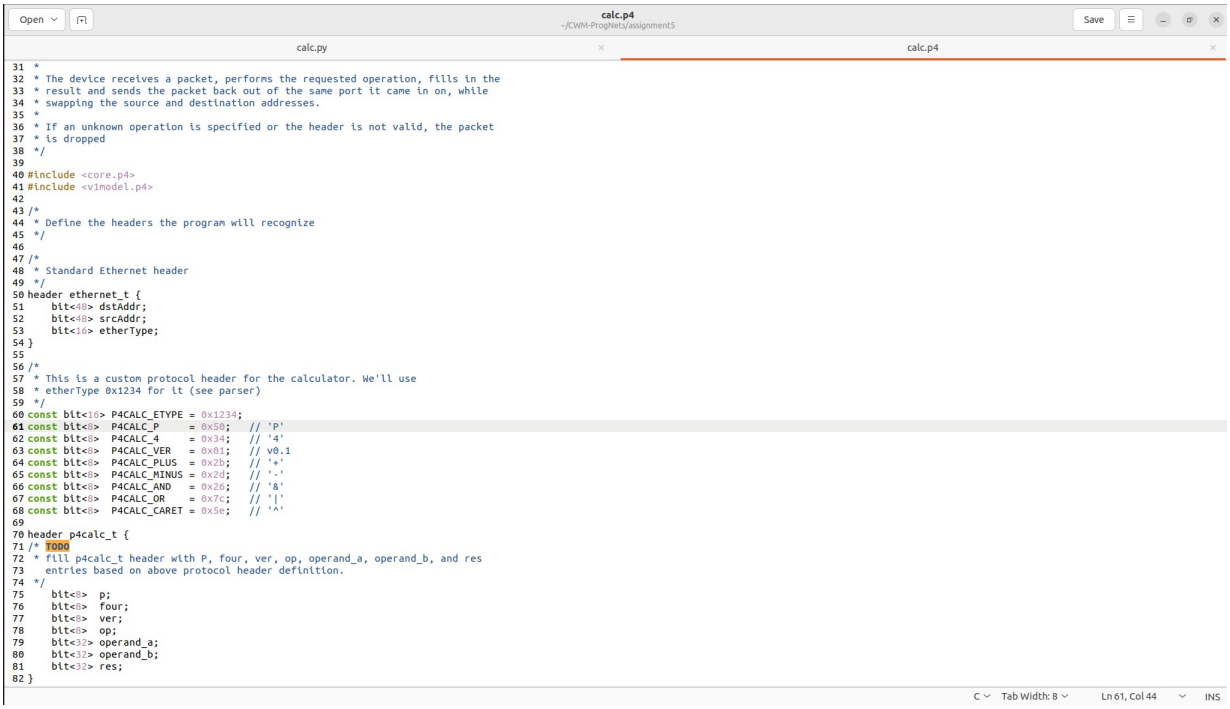
Open
calc.p4
~/CWM-ProgNets/assignment5
Save
calc.py
calc.p4
31 *
32 * The device receives a packet, performs the requested operation, fills in the
33 * result and sends the packet back out of the same port it came in on, while
34 * swapping the source and destination addresses.
35 *
36 * If an unknown operation is specified or the header is not valid, the packet
37 * is dropped
38 */
39
40 #include <core.p4>
41 #include <v1model.p4>
42
43 /*
44 * Define the headers the program will recognize
45 */
46
47 /*
48 * Standard Ethernet header
49 */
50 header ethernet_t {
51 bit<48> dstAddr;
52 bit<48> srcAddr;
53 bit<16> etherType;
54 }
55
56 /*
57 * This is a custom protocol header for the calculator. We'll use
58 * etherType 0x1234 for it (see parser)
59 */
60 const bit<16> P4CALC_ETYPE = 0x1234;
61 const bit<8> P4CALC_P = 0x50; // 'P'
62 const bit<8> P4CALC_4 = 0x34; // '4'
63 const bit<8> P4CALC_VER = 0x01; // v0.1
64 const bit<8> P4CALC_PLUS = 0x2b; // '+'
65 const bit<8> P4CALC_MINUS = 0x2d; // '-'
66 const bit<8> P4CALC_AND = 0x26; // '&'
67 const bit<8> P4CALC_OR = 0x7c; // '|'
68 const bit<8> P4CALC_CARET = 0x5e; // '^'
69
70 header p4calc_t {
71 /* TODO
72 * fill p4calc_t header with P, four, ver, op, operand_a, operand_b, and res
73 entries based on above protocol header definition.
74 */
75 bit<8> p;
76 bit<8> four;
77 bit<8> ver;
78 bit<8> op;
79 bit<32> operand_a;
80 bit<32> operand_b;
81 bit<32> res;
82 }
C
Tab Width: 8
Ln 61, Col 44
INS