```
0x7fa
--------
assert(0 == msg.value)
$s2 = c[0x4]
assert(0 == (0xff & (s[0x2] >> 0xa0)))
assert($s2 < s[0x6])
m[0x0] = 0x6
$s11 = (0x2 * $s2) + sha3(0x0, 0x20)
$t = $s11
$s4 = $s11
assert(0xffffffffffffffff & s[0x1 + $s11])
$s13 = $m
$m = 0x100 + $m
m[$s13] = s[$t]
$s14 = s[0x1 + $t]
m[0x20 + $s13] = 0xffffffffffffffff & $s14
$s15 = $s14
$s12 = $s15
m[0x40 + $s13] = 0xffffffffffffffff & ($s14 >> 0x40)
m[0x60 + $s13] = 0xffffffff & ($s15 >> 0x80)
m[0x80 + $s13] = 0xffffffff & ($s12 >> 0xa0)
m[0xa0 + $s13] = 0xffffffff & ($s12 >> 0xc0)
m[0xc0 + $s13] = 0xffff & ($s12 >> 0xe0)
$t = $s13
$s13 = $s12 >> 0xf0
$s12 = $t
m[0xe0 + $t] = 0xffff & $s13
$s14 = 0 == (0xffffffff & m[0xa0 + $t])
$t = $s14
$s14 = 0 == $s14
if (! $t){
  $s11 = (0xffffffffffffffff & m[0x40 + $s12]) <= (0xffffffffffffffff & block.number)
} else {

  $s11 = $s14
  goto 0x1a4a
}
assert($s11)
$s5 = 0xffffffff & (s[0x1 + $s4] >> 0xc0)
assert($s5 < s[0x6])
m[0x0] = 0x6
$s13 = sha3(0x0, 0x20) + (0x2 * $s5)
$t = $s13
$s7 = 0xffff & (s[0x1 + $s4] >> 0xf0)
if (0xffff & (s[0x1 + $s13] >> 0xf0) > $s7){
  $s7 = 0xffff & (s[0x1 + $t] >> 0xf0)
}
$s11 = ad_mask & s[0x10]
m[0x20 + $m] = 0x0
m[$m] = 0xd9f5aed0000000000000000000000000000000000000000000000000000000000
m[0x4 + $m] = s[$s4]
m[0x24 + $m] = s[$t]
m[0x44 + $m] = 0xffffffffffffffff & ((0xffffffffffffffff & (s[0x1 + $s4] >> 0x40)) - 0x1)
assert(extcodesize($s11))
assert(call(msg.gas - 0x2c6, $s11, 0x0, $m, (0x64 + $m) - $m, $m, 0x20))
$s12 = m[$m]
m[0x0] = $s2
m[0x20] = 0x7
$s11 = intcall2(ad_mask & s[sha3(0x0, 0x40)], $s12, 0xffff & (0x1 + $s7), 0xffffffff & (s[0x1 + $s4] >> 0xc0), $s2, 0x1bb7)
$s12 = 0x1 + $s4
s[$s12] = 0xffffffff00000000ffffffffffffffffffffffffffffffffffffffffffffffff & s[$s12]
s[0xf] = s[0xf] - 0x1
$s12 = s[0xe]
$s10 = $s11
$s15 = call(0x8fc * (0 == $s12), ad_mask & msg.sender, $s12, $m, 0x0, $m, 0x0)
m[$m] = $s10
return($m, (0x20 + $m) - $m)
```