```
$ cat print.ass

        .makeexe

        call  main
        halt

// void print_char(char c)
//                        c is at FP+2
print_char:
        push  FP
        load  FP, SP
// { char string[4];
//   int print_control_block[3];
        sub   SP, 4
//                    string is at FP-1
//       print_control_block is at FP-4 to FP-2
//   string[0]=c;
        load  R1, [FP+2]
        store R1, [FP-1]     // it is a one char "string"
//   print_control_block[0] = $termoutc;
        load  R1, $termoutc
        store R1, [FP-4]
//   print_control_block[1] = 1;
        load  R1, 1
        store R1, [FP-3]
//   print_control_block[2] = string;
        load  R1, FP-1
        store R1, [FP-2]
//   peri(&print_control_block); }
        peri  FP-4
// end of function
        load  SP, FP
        pop   FP
        ret
```

```
// void print_digit(int n)
//                           n is at FP+2
print_digit:
        push  FP
        load  FP, SP
// { char x = n;             x is at FP-1
        sub   SP, 1
        load  R1, [FP+2]
        store R1, [FP-1]
//   if (n<10)
        load  R1, [FP+2]
        comp  R1, 10
        jcond geq, L2
//     print_char(x+'0');
        load  R1, [FP-1]
        add   R1, '0'
        push  R1
        call  print_char
        add   SP, 1
        jump  L3
//   else
L2:
//     print_char(x+'A'-10); }
        load  R1, [FP-1]
        add   R1, 'A'
        sub   R1, 10
        push  R1
        call  print_char
        add   SP, 1
L3:
// end of function
        load  SP, FP
        pop   FP
        ret
```

```
// int print_dec(int n, int base)
                     //  n = FP+2
                     //  base = FP+3
print_dec:
        push  fp
        load  fp, sp
// { int last_dig, other_digs, printed;
        sub   sp, 4
                     //  last_dig = FP-1
                     //  other_digs = FP-2
                     //  printed = FP-3
                     //  length_others = FP-4
//   last_dig = n % base;
        load  R1, [FP+2]
        mod   R1, [FP+3]
        store R1, [FP-1]
//   other_digs = n / base;
        load  R1, [FP+2]
        div   R1, [FP+3]
        store R1, [FP-2]
//   printed = 1;
        load  R1, 1
        store R1, [FP-3]
//   if (other_digs != 0)
        load  R1, [FP-2]
        comp  R1, 0
        jcond eql, L1
//   { int length_others = print_dec(other_digs, base);
          push  [FP+3]
          push  [FP-2]
          call  print_dec
          add   SP, 2
          store R0, [FP-4]
//     printed += length_others; }
          load  R1, [FP-3]
          add   R1, [FP-4]
          store R1, [FP-3]
L1:
//   print_digit(last_dig);
        push  [FP-1]
        call  print_digit
        add   SP, 1
//   return printed; }
        load  R0, [FP-3]
        load  SP, FP
        pop   FP
        ret
```

```
// void main()

main:
        push  FP
        load  FP, SP
//                     no locals
// { print_dec(2468, 10);
        push  10
        push  2468
        call  print_dec
        add   sp, 2
//   print_char('\n');
        push  '\n'
        call  print_char
        add   sp, 1
// end of function
        load  SP, FP
        pop   FP
        ret
```