Operating Systems

$\frac{\text{Software}}{\text{Hardware}}$ OS

- provide/manage access to shared resources.
- provides services to make working w/ resources easier

Project 12.

Sys.jack

halt - inf. loop.

error - just print ERR #.

init - • calls init functions of all other OS classes
- • call Main.main
- • call halt.

Array - just call Memory.alloc/deAlloc.

String

- store char array, cur len, max len in fields.
- intValue (String → int)
- setInt (int → String).

int Value

val = 0
loop through chars as long as they are digits.

'0' .... '9'

48 57

is digit : $(47 < c) \& (c < 58)$

$val = 10 * val + (c - 48)$

set Int

```
while n > 9:
    get last digit with % 10
    append digit to string.
    chop off last digit with / 10.
```

Problems:

① mod? $a - (a/d) * d$

② reverse?

a) build reversed String afterwards.

b) use recursion.

③ negative #'s? deal w/ @ outermost layer; inner code only works for pos.

Keyboard.

keyPressed — use Memory.peek.

readChar — 2 loops + Output.printChar

readLine — readChar in a loop,
append to String (or erase if
backspace).

Math

- naive is OK!
- Handle negatives separately.

More efficient multiplication:

$$\begin{array}{rll}
362 & & \\
\times\ 43 & & \\ \hline
1086 & \leftarrow & 3\times 362 \\
\text{- - -}\ 0 & & 40\times 362 \\
+\ \text{- - -} & & \\ \hline
\end{array}$$

```
      10010
  x    1101
  ----------
      10010   <- 1 x 10010
    10010     <- 100 x 10010
+ 10010          1000 x  ...
  ----------
```

```
mul(x, y):
    res = 0
    for i in 0..15:
        if bit i of y = 1:
            res += x
        x = 2x        <-  x = x + x.
    return res.
```

To look at bit i of y:

```
   101101b0000
&  00000010000
  -------------
   000...b0000
         ↑
```

check if $(y \& 2^i) = 0$