CSc 451, Spring 2003
Examination #1 Solutions

Problem 1: (15 points)

Write a program `expand` that reads a spell-checker word list with entries such as these:

```
abbreviate,s,d,\ing,\ion
bar,s,"ed,"ing
calmest
```

and prints all forms of each word.

```
link split
procedure main()
    while ws := split(read(), ',') do {
        write(base := get(ws))
        every suf := !ws do
            if suf[1] == "\\" then
                write(base[1:-1]||suf[2:0])
            else if suf[1] == "\"" then
                write(base||base[-1]||suf[2:0])
            else
                write(base||suf)
        }
end
```

Problem 2: (15 points)

Write a procedure `eval(s)` that evaluates string representations of expressions consisting of integer values and the binary operators `+`, `-`, `*`, and `/`.

```
link split
procedure eval(s)
    ws := split(s,'+*/-',1)
    result := get(ws)
    while result := get(ws)(result, get(ws))
    return result
end
```

Some students used an approach similar to this one by Mr. Rini:

```
procedure eval(s)
    L := split(s,'+*/-',1)
    while *L ~= 1 do
        L := help(L)

    return L[1]
end
procedure help(lst)
    r := lst[2](lst[1],lst[3])
    return [r]|||lst[4:0]
end
```

Problem 3: (10 points)

Write a procedure `Reverse(x)` that reverses either strings or lists. If `x` is a list, the reversal is at the top level only. You may use the built-in function `reverse` in your solution.

```
procedure Reverse(x)
    if type(x) == "string" then return reverse(x)
    R := []
    every push(R, !x)
    return R
end
```

My intention was that `Reverse` should not change its argument but because I did not state that both applicative and non-applicative versions received full credit.

Mr. Kobes, Mr. Lucas, and Mr. Wampler took advantage of polymorphic operations and the swap operator:

```
procedure Reverse(x)
    every i := 1 to *x/2 do
        x[i] :=: x[-i]
    return x
end
```

Problem 4: (8 points)

Write a procedure `altbang(s)` that generates the characters of `s` working in from each end in an alternating manner. If `s` is the null string, the result sequence is empty.

```
procedure altbang(x)
    suspend x[i:=1 to *x & (i|-i)] \ *x
end
```

Mr. Graham produced a unique solution:

```
procedure altbang(s)
    temp := s
    suspend |{c := temp[1] & temp := reverse(temp[2:0]) & c}
end
```

Several solutions took this form:

```
procedure altbang(s)
    every i := 1 to *s/2 do {
        suspend s[i]
        suspend s[-i]
        }

    if *s % 2 = 1 then
        suspend s[i+1]  # Another way: suspend s[(*s+1)/2]
end
```

Mr. Leslie used the approach of alternately popping and pulling from a list of the characters.

Problem 5: (20 points)

Write a program `exttotal` that reads "`ls -s`" output and prints a table of file extensions and the total number of blocks used by files of that type in the current directory.

```
link split
procedure main()
    t := table(0)
    f := open("ls -s", "rp")

    read(f)

    while ws := split(read(f)) do {
        blocks := ws[1]
        nmp := split(ws[2], '.')
        if *nmp = 1 then
            ext := "(None)"
        else
            ext := nmp[-1]

        t[ext] +:= blocks
        }

    every pair := !sort(t,2) do
        write(left(pair[1],10), " ", right(pair[2],6), " blocks")
end
```

Problem 6: (10 points)

Write a program `lensort` that reads a file named on the command line and prints the lines of the file in order of increasing length.

```
procedure main(a)
    f := open(a[1]) | stop(a[1], ": can't open")
    lines := []
    while line := read(f) do
        put(lines, [*line, line])

    every write((!sortf(lines, 1))[2])
end
```

Mr. Leslie used a table keyed by line length. The value for a given length was a concatenation of all lines having that length.

Problem 7: (1 point each; 5 points total)

Write an expression whose result sequence ...

...is empty: `&fail`, `1 < 0`, and `"a"[2]` are some examples

...is infinite: `|1` (repeated alternation)

...has length 2: `1 | 2`

...has length 10: `!&digits` (Mr. Kobes)

...has length 100: `1 to 100`

Problem 8: (2 points each, 8 points total)

Write expressions that have the following result sequences. You may use built-in functions such as `repl(s,n)` but you may not write any helper procedures.

(a) All capital letters in the string `s`. For example, if `s` is "The Right Way", the result sequence would be {"T", "R", "W"}.

```
!s == !&ucase
```

(b) The character and position of each character in the string `s`. For example, if `s` is "abc", the result sequence would be {"a", 1, "b", 2, "c", 3} — six values altogether.

```
i := 1 to *s & s[i] | i
```

(c) The infinite sequence {1, 1, 2, 1, 2, 3, 1, 2, 3, 4, ...}.

```
i := 0 & |(1 to (i +:= 1))
```

(d) The integers in the list `L`, in descending order. For example, if L is ["x", 5, 3, "y", 10, 5, 4.1], the result sequence would be {10, 5, 5, 3}.

```
L2 := sort(L) & i := *L2 to 1 by -1 &
  type(L2[i]) == "integer" & L2[i];
```

Problem 9: (6 points)

Write a procedure `invert(t)` that returns an inverted copy of the table `t` by swapping keys and values. `invert(t)` fails if the table `t` contains any values that are not unique.

```
procedure invert(t)
    new := table()

    every k := key(t) do
        new[t[k]] := k

    if *new = *t then
        return new
end
```

A number of solutions simply failed upon discovery of a duplicate:

```
procedure invert(t)
    new := table()
    every k := key(t) do {
        if \new[t[k]] then fail
        new[t[k]] := k
        }
    return new
end
```

Problem 10: (3 points)

Show the output of this program:

```
procedure main()
    every write(("+"|"*")(2|3, 4|5))
end
```

Output: 6, 7, 7, 8, 8, 10, 12, 15 (one value per line)

EXTRA CREDIT SECTION (one point each)

(a) *Who was known as "bikmort"?* Tim Korb

(b) *What is the output of the following expression?* Nothing—`every` always fails!

```
every write(every 1 to 10)
```

(c) *Write a procedure `defvalue(t)` that returns the default value of table `t`.*

```
procedure defvalue(t)
    return t[[]]
end
```

(d) *List the last names of ten other students in this class.* Two students came up with seven.

(e) *Write a good one point extra credit question and answer it correctly.*
Mr. Wampler wrote: "What is the shortest Icon program that will successfully compile?"