```
PROBLEM 1
TEST CASES

PROBLEM 1C

# Simprex.freeVarsExpString "(simprec a (b c (+ b (* c d))) e)";;
- : Simprex.S.elt list = ["a"; "d"; "e"]
# Simprex.substString "(simprec a (b c (+ b (* c (/ a d)))) d)"
"((a (+ b c)) (d (- b c)))";;
  (simprec (+ b c) (b.4 c.5 (+ b.4 (* c.5 (/ (+ b c) (- b c))))) (- b c))
- : unit = ()
# SimprexEnvInterp.runString "(simprex (x) (simprec 0 (n sum (+ n sum)) x))" [10];;
- : int = 55
# SimprexEnvInterp.runFile "fact.spx" [5];;
- : int = 120
# SimprexEnvInterp.runFile "expt.spx" [3;4];;
- : int = 81
# SimprexEnvInterp.runFile "sos.spx" [4];;
- : int = 30

PROBLEM 1D
(We are using the simplification that lo = 1)

simprex> (sigma i 1 2 i);;
3

simprex> (sigma i 1 7 i);;
28

simprex> (sigma i 1 2 (* i i));;
5


PROBLEM 2
TEST CASES FOR PARTIAL EVALUATION

# test();;
Add-2-3 -- OK!
Simple1 -- OK!
Years -- OK!
Residuals1 -- OK!
Residuals2 -- OK!
Residuals3 -- OK!
- : unit = ()

PROBLEM 3
TEST CASES FOR PRIMITIVE OPERATORS

valex> (abs -17)
17

valex> (abs 42)
42

valex> (sqrt 25)
5

valex> (sqrt 35)
5

valex> (sqrt 36)
6

valex> (between 1 20)
(list 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20)

valex> (between 3 7)
```

```
(list 3 4 5 6 7)

valex> (between 7 3)
#e

valex> (reverse (list 1 2 3))
(list 3 2 1)

valex> (reverse (between 3 7))
(list 7 6 5 4 3)

valex> (reverse (list))
#e


PROBLEM 4
TEST CASES FOR CLASSIFY

# testEnvInterp();;

classify1 95 = 'A' OK!
classify1 85 = 'B' OK!
classify1 75 = 'C' OK!
classify1 65 = 'D' OK!
classify1 55 = 'F' OK!
classify1 45 = 'F' OK!
classify2 10 20 3 4 = 30 OK!
classify2 10 20 3 6 = 120 OK!
classify2 10 20 3 5 = 2 OK!
classify3 0 = 0 OK!
classify3 5 = 5 OK!
classify3 10 = 11 OK!
classify3 20 = 21 OK!
classify3 25 = 5 OK!
classify3 30 = 4 OK!

============================================================
TESTING SUMMARY:
Passed 151 of 151 test cases.
- : unit = ()
```