```
;; Calculate binomial probabilities.
;; Coded in lisp by Roger Kaiser from multiply algorithm in
;; Catherine Loader's paper, "Fast and Accurate Computation of Binomial Probabilities"

(defun dbinom-mult (x n p)
  (if (> (+ x x) n)
      (dbinom-mult (- n x) n (- 1 p))
      (handler-case
          (do ((j0 0) (j1 0) (j2 0) (f 1))
              ((and (>= j0 x) (>= j1 x) (>= j2 (- n x))) f)
            (if (and (< j0 x) (< f 1))
                (setf j0 (1+ j0)
                      f (* f (/ (+ (- n x) j0) j0)))
                (if (< j1 x)
                    (setf j1 (1+ j1)
                          f (* f p))
                    (setf j2 (1+ j2)
                          f (* f (- 1 p))))))
        (floating-point-underflow () 'Underflow)
        (floating-point-overflow () 'Overflow))))

(defun test-dbinom-mult(n p filename)
  (with-open-file (str filename :direction :output)
    (format str "k      dbinom(~a,k,~a)~%" n p)
    (do ((i 0 (+ i 1)))
        ((> i n) nil)
      (let ((f (dbinom-mult i n p)))
        (format str "~a ~a~%" i f)))))

(defun test-rat ()
  (test-dbinom-mult 16 1/2 "test.txt"))

(defun deaths-single-float ()
  (test-dbinom-mult 2000 0.00146S0 "deaths-single-float.txt"))

(defun deaths-double-float ()
  (test-dbinom-mult 2000 0.00146D0 "deaths-double-float.txt"))

(defun deaths-long-float ()
  (setf (ext:long-float-digits) 64)
  (test-dbinom-mult 2000 0.00146L0 "deaths-long-float.txt"))

(defun deaths-rat ()
  (test-dbinom-mult 2000 146/100000 "deaths-rational.txt"))
```