ALGORITHM 47
ASSOCIATED LEGENDRE FUNCTIONS OF THE FIRST KIND FOR REAL OR IMAGINARY ARGUMENTS

JOHN R. HERNDON
Stanford Research Institute, Menlo Park, California

```
procedure  LEGENDREA (m, n, x, r);  value m, n, x, r;
           integer m, n;  real x, r;
comment  This procedure computes any Pnm(x) or Pnm(ix) for
  n an integer less than 20 and m an integer no larger than n.
  The upper limit of 20 was taken because (42)! is larger than
  10^49. Using a modification of this procedure values up to n=35
  have been calculated. If Pnm(x) is desired, r is set to zero. If
  r is nonzero, Pnm(ix) is computed;
begin
           integer i, j;  array Gamma [1:41];
           real p, z, w, y;
           if n = 0 then
             begin p := 1;
             go to gate end;
           if n < m then
             begin p := 0;
             go to gate end;
           z := 1; w := z;
           if n=m then go to main;
           for i := 1 step 1 until n−m do
             z := x × z;
     main: Gamma [1] := 1;
           for i := 2 step 1 until n+n+1 do
             begin Gamma [i] := w × Gamma [i−1];
             w := w+1 end;
           w := 1;  y := w/(x × x);
           if r=0 then
             begin y := −y;
             w := −w end;
           if x=0 then
             begin i := (n−m)/2;
             if (i+i) ≠ (n−m) then
               begin p := 0;
               go to gate end;
             p := Gamma [m+n+1]/(Gamma [i+1] × Gamma
                  [m+i+1]);
             go to last end;
           j := 3;  p := 0;
           for i := 1 step 1 until 12 do
             begin if (n−m+2)/2 < i then go to last end;
             p := p + Gamma [n+n−i−i+3] × z/(Gamma
               [i] × Gamma [n−i+2] × Gamma [n−i−i−
               m+j]);
             z := z × y end;
     last: z := 1;
           for i := step 1 until n do
             z := z+z;
           p := p/z;
           if r ≠ 0 then
             begin i := n−n/4;
             if 1 < i then
               p := −p end;
           if m = 0 then go to gate;
           j := m/2;  z := abs(w+x × x);
           if m ≠ (j+j) then
             begin z := sqrt (z);
             j := m end;
           for i := step 1 until j do
             p := p × z;
     gate: LEGENDREA := p
end LEGENDREA;
```

CERTIFICATION OF ALGORITHM 47
ASSOCIATED LEGENDRE FUNCTIONS OF THE FIRST KIND FOR REAL OR IMAGINARY ARGUMENTS [John R. Herndon, *Comm. ACM*, Apr. 1961]

RICHARD GEORGE*
Argonne National Laboratory, Argonne, Ill.

* Work supported by United States Atomic Energy Commission.

This procedure was programmed in FORTRAM for the IBM 1620 and was tested with a number of real arguments. A few errors were detected:

1. In the following sequence the *end* must be removed:

   begin if $(n - m + 2)/2 < i$ **then go to** last **end**;

2. In these, the lower bound of 1 is needed:

   for i := **step** 1 **until** n **do**
   for i := **step** 1 **until** j **do**

3. There are four places where integer arithmetic is clearly intended and we must substitute the symbol ÷ for the symbol /.

In addition, it might be mentioned that the statement

if $n = m$ **then go to** main;

could be omitted from the ALGOL program without harm, though the FORTRAN version requires it. Here, and elsewhere in the procedure, one might make an equivalent but more succinct statement. With change in style, the variable j could be eliminated.

CERTIFICATION OF ALGORITHM 47 [S16]
ASSOCIATED LEGENDRE FUNCTIONS OF THE FIRST KIND FOR REAL OR IMAGINARY ARGUMENTS [John R. Herndon, *Comm. ACM 4* (Apr. 1961), 178]

S. M. COBB (Recd. 6 Feb. 1969, 12 May 1969 and 9 July 1969)
The Plessey Co. Ltd., Roke Manor, Romsey, Hants, England

KEYWORDS AND PHRASES: Legendre function, associated Legendre function, real or imaginary arguments
CR CATEGORIES: 5.12

This procedure was tested and run on the I.C.T. Atlas computer.

In addition to the errors mentioned in the certification of August 1963 [2] the following points were noted.

1. The requirement that when $n < m$ $p := 0$ must take precedence over $p := 1$ when $n = 0$. Hence the order of the first two **if** statements must be interchanged.

2. Most computers fail on division by zero. Hence the statement beginning **if** $x = 0$ **then** and ending with **go to** *last* **end**; should be inserted between $w := 1$; and $y := w/(x \times x)$.

3. When $x = 0$, if the argument of the Legendre function is to be considered as real p must be multiplied by $(-1)^i$. This is achieved by inserting after the statement beginning $p := Gamma[m+n+1]$ the **if** statement

if r **then** $p := p \times (-1) \uparrow i$;

(For a change in the meaning of r see item 5 below.)

4. After the label *last* in the compound statement beginning **if** $r \neq 0$ the statement $i := n - n \div 4$; is wrong. This should read

$i := n - 4 \times (n \div 4)$;

5. Since r is used only as an indicator it is better that it be declared as **Boolean**. It can then be given the value **true** if the argument of the Legendre function is x and **false** if it is ix. The following program changes are then necessary. The statement beginning

if $r = 0$ **then**

becomes

if r **then**

The statement beginning

if $r \neq 0$ **then**

becomes

if $\neg$ r **then**

6. Computing time can be saved in several ways. First we should declare another integer k and set it equal to $n - m$. The first statement of the procedure is then

$k := n - m$;

The next statement will begin

if $k < 0$ **then**

(This replaces **if** $n < m$ **then** whose position has been changed in accordance with item 1 above.)
$n - m$ is then replaced by k in the lines
for $i := 1$ **step** 1 **until** $n - m$ **do**

and

if $(i+1) \neq (n-m)$ **then**

Removing j as suggested in the previous certification leaves it free to be set to $k \div 2$. This requires the following modification: instead of the unnecessary statement **if** $n = m$ **then go to** *main* put

$j := k \div 2$;

In the statement beginning **if** $x = 0$ **then** replace the line

begin $i := (n-m) \div 2$;

by

begin $i := j$;

In the **for** loop beginning **for** $i := 1$ **step** 1 **until** 12 **do** a further small saving in computer time could be achieved by setting k to $n - i$. The loop thus becomes

```
for i := 1 step 1 until 12 do
begin if j + 1 < i then go to last;
  k := n - i;
  p := p + Gamma[2×k+3] × z/Gamma[i] × Gamma[k+2] ×
    Gamma[k-i-m+3]);
  z := z × y
end
```

For real argument the program was tested as follows.

(i) $x = 0(0.1)1$, $m = 0(1)3$, $n = 0(1)3$
(ii) $x = 1.2(0.2)2.8$, $m = 0(1)2$, $n = 0(1)2$
(iii) $m = 0$, $n = 9$, $x = 0(0.2)1, 2(2)10$.

For imaginary argument we used

$x = 0(0.2)2$, $m = 0(1)2$, $n = 0(1)2$.

Checking for real argument was carried out where possible using [1], agreement being obtained in all cases to the maximum number of figures available, which varied between 6 and 8. For all other cases [3] had to be used, giving only a 5 figure check.

REFERENCES:

1. ABRAMOWITZ, M., AND STEGUN, I. A. Handbook of mathematical functions. AMS 55, Nat. Bur. Stand. US Govt. Printing Off., Washington, D.C., 1964.
2. GEORGE, R. Certification of Algorithm 47. *Comm. ACM 6* (Aug. 1963), 446.
3. MORSE, P. M., AND FESBACH, H. *Methods of Theoretical Physics Pt. II.* McGraw Hill, New York, 1953.