1 SUMMARY

To **evaluate the integral** $\int_a^b f(x)\,dx$ to a specified **relative or absolute accuracy.**

The subroutine uses an adaptive method based on a 3-point Gaussian integration scheme. The stepsizes are reduced locally to match the difficulty of the integrand in such a way that all function values are used economically. Also the accuracy criterion is modified at each stage to obtain a local relative accuracy which is such that the global accuracy will tend to be relative also. See I.G. Robinson, 'An Adaptive Gaussian Integration', Australian Comp. Journal, Vol. 3, no. 3. The subroutine returns an error estimate.

The user must provide a subroutine to evaluate $f(x)$ at any point in the interval $a<x<b$. Note however that as a consequence of using a Gauss scheme, values of $f(x)$ will not be required at the limit points a and b and that this avoids complications which might have arisen should f(x) have had an integrable singularity at either limit.

ATTRIBUTES — Version: 1.0.0. **Remark:** Shorter than `QA05` but less efficient on some difficult problems, but well worth trying first on most. **Types:** `QA04A`; `QA04AD`. Both use double precision arithmetic. **Original date:** May 1972. **Origin:** I.G.Robinson, Univ. of Melbourne, implemented for Harwell by A.B. Smith, later modified by A.R.Curtis.

2 HOW TO USE THE PACKAGE

2.1 The argument list

The single precision version

```
CALL QA04A(Q,A,B,EPS,MAXIT,F)
```

The double precision version

```
CALL QA04AD(Q,A,B,EPS,MAXIT,F)
```

`Q` is a `REAL` (`DOUBLE PRECISION` in the D version) variable which is set by the subroutine to the value of the integral.

`A` is a `REAL` (`DOUBLE PRECISION` in the D version) variable which must be set by the user to a the lower limit of integration. It is not altered by the subroutine.

`B` is a `REAL` (`DOUBLE PRECISION` in the D version) variable which must be set by the user to b the upper limit of integration. It is not altered by the subroutine.

`EPS` is a `REAL` (`DOUBLE PRECISION` in the D version) variable which must be set by the user to the accuracy required in the final value of the integral. The subroutine attempts to obtain the integral to a relative accuracy.

`MAXIT` is an `INTEGER` variable which must be set by the user to the maximum number of levels of subdivision to be allowed. **Restriction:** `MAXIT` ≤ 30. On return, `MAXIT` is set negative if the subroutine has failed to achieve the requested accuracy in `MAXIT` levels. Otherwise, it is set to the maximum number of levels actually used.

`F` is the name of the `FUNCTION` subprogram which must be provided by the user to evaluate $f(x)$, see section 2.2.

2.2 The evaluation of f(x)

The user must provide a `FUNCTION` subprogram to evaluate the integrand $f(x)$. Any name may be chosen but it should be declared `EXTERNAL` in the calling program. The `FUNCTION` has just one argument. It must be double precision for `QA04AD`.

```
FUNCTION name(X)

 statements to set name=f(x)

RETURN
END
```

2.3 The common area

The integration subroutine makes reference to a common block.

The single precision version

```
COMMON/QA04B/ DIV,LP,NF,EST
```

The double precision version

```
COMMON/QA04BD/ DIV,LP,NF,EST
```

`DIV` is a `REAL` (`DOUBLE PRECISION` in the D version) variable and is an adaptive divisor used to modify the accuracy criterion at each level of interval subdivision. It is set to 1.4 a value found empirically to be reliable.

`LP` is an `INTEGER` variable and is the Fortran unit number used to print diagnostic messages. Normally set to 6 but may be reset by the user.

`NF` is an `INTEGER` variable which on return from the subroutine will give the number of function evaluations that were required to obtain the value of the integral.

`EST` is a `REAL` (`DOUBLE PRECISION` in the D version) variable and on return will contain an approximate estimate of the relative error attained. Note that it is only necessary to include the common statement in the calling program when its variables are to be modified or used.

3 GENERAL INFORMATION

Use of common: uses common areas `QA04B/BD`, see section 2.3.

Workspace: there are 13 arrays private to `QA04A/AD` each of length 30 words.

Other routines called directly: requires a `FUNCTION` subroutine of the form `F(X)` to evaluate the integrand $f(x)$, see section 2.2.

Input/output: diagnostic messages are output on unit 6 normally, see section 2.3.

Restrictions: no. of levels of interval subdivision must be ≤ 30, see `MAXIT`.

4 METHOD

An adaptive scheme is used in which the 3-point Gauss formula is applied to each interval, the first interval being the range of integration $a \le x \le b$. At each stage the accuracy criterion is tested and if not satisfied the interval is decreased by using the left-hand Gauss point as the mid-point of another 3-point Gauss integration. At each reduction of the interval the accuracy criterion is modified by a factor `1/DIV`. The process continues until an interval is found which integrates to the required accuracy. When this happens the integration moves on by taking the next free point to the right into the scheme. The method is described in, Robinson, I.G. 'An Adaptive Gaussian Integration', Australian Computer Journal, Vol.3, No.3, Aug. 1972. If at any point the interval has been subdivided `MAXIT` times and still the error requirement cannot be met, an error diagnostic is given before return, and `MAXIT` is set negative. The diagnostic states the number of points in the range at which this happened. The result `Q` is set to the best estimate available, and the relative error estimate found is returned in `EST`.

5 EXAMPLE OF USE

Suppose we require to evaluate

$$Q = \int_0^1 \sqrt{x} \log x \; dx$$

to an accuracy of 10^{-5}. The code might go as follows

```
C      DECLARE FUN TO BE EXTERNAL
       EXTERNAL FUN
C      SET INTEGRATION LIMITS
       A=0
       B=1
C      SET ACCURACY REQUIREMENT
       EPS=1E-5
C      SET INTERVAL REDUCTION LIMIT
       MAXIT=30
C      EVALUATE THE INTEGRAL
       CALL QA04A(Q,A,B,EPS,MAXIT,FUN)
C      PRINT THE RESULT
       WRITE(6,10) Q
  10   FORMAT('INTEGRAL  VALUE =' ,E13.6)
       STOP
       END
```

A `FUNCTION` subprogram called `FUN` will also be required.

```
       FUNCTION  FUN(X)
       FUN=SQRT(X)*ALOG(X)
       RETURN
       END
```