J

j0() — Extended function (libm)

Compute Bessel function
#include <xmath.h>
double j0(double *z*);

j0 computes the Bessel function of the first kind for order 0, for its argument *z*.

Example

This example, called **bessel.c**, demonstrates the Bessel functions **j0**, **j1**, and **jn**. Compile it with the following command line

```
      cc -f bessel.c -lm
```

to include floating-point functions and the mathematics library.

```
#include <math.h>
#include <stdlib.h>
#include <xmath.h>
dodisplay(double value, char *name)
{
      if (errno)
            perror(name);

      else
            printf("%10g %s\n", value, name);
      errno = 0;
}

#define display(x) dodisplay((double)(x), #x)

main()
{
      extern char *gets();
      double x;
      char string[64];

      for(;;) {
            printf("Enter number: ");
            if(gets(string) == 0)
                  break;
            x = atof(string);

            display(x);
            display(j0(x));
            display(j1(x));
            display(jn(0,x));

            display(jn(1,x));
            display(jn(2,x));
            display(jn(3,x));
      }
}
```

See Also

extended mathematics, j1, jn

Notes

j0 is not described in the ANSI Standard. Any program that uses it does not conform strictly to the Standard, and may not be portable to other compilers or environments.

j1() — Extended function (libm)

Compute Bessel function
#include <xmath.h>
double j1(double *z***);**

j1 takes the argument *z* and computes the Bessel function of the first kind for order 1.

Example

For an example of this function, see the entry for **j0**.

See Also

extended mathematics, j0, jn

Notes

j1 is not described in the ANSI Standard. Any program that uses it does not conform strictly to the Standard, and may not be portable to other compilers or environments.

jday_to_time() — Extended function (libc)

Convert Julian date to system time
#include <time.h>
#include <xtime.h>
time_t jday_to_time(jday_t *time***);**

jday_to_time converts Julian time to system time.

time is the Julian time to be converted. It is of type **jday_t**, which is defined in the header **xtime.h**. **jday_t** is a structure that consists of two **unsigned long**s. The first gives the number of the Julian day, which is the number of days since the beginning of the Julian calendar (January 1, 4713 B.C.). The second gives the number of seconds since midnight of the given Julian day.

jday_to_time returns the Julian time as converted to type **time_t**. This type is defined in the header **time.h** as being equivalent to a **long**. **Let's C** defines the current system time as being the number of seconds from January 1, 1970, 0h00m00s GMT, which is equivalent to the Julian day 2,440,587.5.

See Also

extended time, jday_to_tm, time_to_jday, tm_to_jday, xtime.h

Note

This function is of use mainly to astronomers, geographers, and historians.

To conform to the ANSI Standard, this function has been moved from the header **time.h** to the header **xtime.h**. This may require that some code be altered.

jday_to_tm() — Extended function (libc)

Convert Julian date to system calendar format
#include <time.h>
#include <xtime.h>
tm *jday_to_tm(jday_t *time***);**

jday_to_tm converts Julian time to the system calendar format.

time is the Julian time to be converted. It is of type **jday_t**, which is defined in the header **xtime.h**. **jday_t** is a structure that consists of two **unsigned long**s. The first gives the number of the Julian day, which is the number of days since the beginning of the Julian calendar (January 1, 4713 B.C.).

The second gives the number of seconds since midnight of the given Julian day.

jday_to_tm returns a pointer to a copy of the structure **tm**, which is defined in the header file **time.h**. For more information on this structure, see the Lexicon entry for **time**.

See Also

extended time, jday_to_time, time_to_jday, tm_to_jday, xtime.h

Note

This function is of use mainly to astronomers, geographers, and historians.

To conform to the ANSI Standard, this function has been moved from the header **time.h** to the header **xtime.h**. This may require that some code be altered.

jmp_buf — Type

Type used with non-local jumps
#include <setjmp.h>

jmp_buf is a type defined in the header **setjmp.h**. It is the type used to hold the current environment to enable a non-local jump. The usual contents of the **jmp_buf** array will be the contents of registers; however, its contents are defined by the implementation.

Cross-references

Standard, §4.6
The C Programming Language, ed. 2, p. 254

See Also

non-local jumps, setjmp.h

Notes

Because **jmp_buf** usually does not contain anything except the current contents of the registers, one should not expect values of local variables or register variables to restored properly.

Historically, code has been written that calls **setjmp** and **longjmp** with an argument of type **jmp_buf**, but without taking its address. This code works because an array passed as a parameter is automatically converted to a pointer. Because structures can now be passed by value, such arguments are no longer converted to pointers. However, because both **setjmp** and **longjmp** expect a pointer argument, the type of **jmp_buf** is restricted to an array type in order to preserve existing code.

If **jmp_buf** must be a structure of heterogeneous elements, then it could be defined as a one-element array of such structures.

jn() — Extended function (libm)

Compute Bessel function
#include <xmath.h>
double jn(short *n*, **double** *z*);

jn takes an argument *z* and computes the Bessel function of the first kind for order *n*.

Example

For an example of this function, see the entry for **j0**.

See Also

extended mathematics, j0, j1

Notes

jn is not described in the ANSI Standard. Any program that uses it does not conform strictly to the Standard, and may not be portable to other compilers or environments.