Help

```
#include <math.h>
#include "complex.h"
#include "mathtools.h"

/* (C) Copr. 1986-92 Numerical Recipes Software
    A2.>$Y0%9j. */
double Real( fcomplex g )
{
  return g.r;
}

double Imm( fcomplex g )
{
  return g.i;
}

fcomplex Cadd(fcomplex a, fcomplex b)
{
  fcomplex c;
  c.r=a.r+b.r;
  c.i=a.i+b.i;
  return c;
}

fcomplex Csub(fcomplex a, fcomplex b)
{
  fcomplex c;
  c.r=a.r-b.r;
  c.i=a.i-b.i;
  return c;
}

fcomplex Cmul(fcomplex a, fcomplex b)
{
  fcomplex c;
  c.r=a.r*b.r-a.i*b.i;
  c.i=a.i*b.r+a.r*b.i;
  return c;
}
```

```
fcomplex Complex(double re, double im)
{
  fcomplex c;
  c.r=re;
  c.i=im;
  return c;
}

fcomplex Conjg(fcomplex z)
{
  fcomplex c;
  c.r=z.r;
  c.i = -z.i;
  return c;
}

fcomplex Cdiv(fcomplex a, fcomplex b)
{
  fcomplex c;
  double r,den;
  if (fabs(b.r) >= fabs(b.i)) {
  r=b.i/b.r;
  den=b.r+r*b.i;
  c.r=(a.r+r*a.i)/den;
  c.i=(a.i-r*a.r)/den;
  } else {
  r=b.r/b.i;
  den=b.i+r*b.r;
  c.r=(a.r*r+a.i)/den;
  c.i=(a.i*r-a.r)/den;
  }
  return c;
}

double Cabs(fcomplex z)
{
  double x,y,ans,temp;
  x=fabs(z.r);
  y=fabs(z.i);
  if (x == 0.0)
  ans=y;
```

```
  else if (y == 0.0)
  ans=x;
  else if (x > y) {
  temp=y/x;
  ans=x*sqrt(1.0+temp*temp);
  } else {
  temp=x/y;
  ans=y*sqrt(1.0+temp*temp);
  }
  return ans;
}

fcomplex Csqrt(fcomplex z)
{
  fcomplex c;
  double x,y,w,r;
  if ((z.r == 0.0) && (z.i == 0.0)) {
  c.r=0.0;
  c.i=0.0;
  return c;
  } else {
  x=fabs(z.r);
  y=fabs(z.i);
  if (x >= y) {
    r=y/x;
    w=sqrt(x)*sqrt(0.5*(1.0+sqrt(1.0+r*r)));
  } else {
    r=x/y;
    w=sqrt(y)*sqrt(0.5*(r+sqrt(1.0+r*r)));
  }
  if (z.r >= 0.0) {
    c.r=w;
    c.i=z.i/(2.0*w);
  } else {
    c.i=(z.i >= 0) ? w : -w;
    c.r=z.i/(2.0*c.i);
  }
  return c;
  }
}
```

```
fcomplex Clog(fcomplex z)
{
    fcomplex ztmp;
  double pgd2 = 1.5707963267949; /**questo  pig
    reco/2 **/

    if (z.i == 0.0 && z.r > 0.0) {
        ztmp.r = log(z.r);
        ztmp.i = 0.0;
    } else if (z.r == 0.0) {
        if (z.i > 0.0) {
            ztmp.r = log(z.i);
            ztmp.i = pgd2;
        } else {
            ztmp.r = log(-(z.i));
            ztmp.i = - pgd2;
        }
    } else {
        ztmp.r = log(sqrt(z.r*z.r + z.i*z.i));
        ztmp.i = atan2(z.i,z.r);  /**Calculates
    the arctangent of x (atan) or the arctangent of y
    /x (atan2).*/
    }
    return(ztmp);
}
/*
  logaritmo of the Gamma function of a complex
    number
  Si utilizza la formula in Press per la gamma di
  un numero reale. E detto che converge per Re(
    xx)>0
  Attento nell'algoritmo: la formula  per lnG(x
    x+1)
  Utilizzando il fatto che G(xx+1)=xx*G(xx) si
    ha che
  lnG(xx) = lnG(xx+1) - ln(xx)
  Questo spiega la divisione finale per xx
*/

fcomplex Cexp(fcomplex g)
{
```

```
  return Complex(exp(g.r)*cos(g.i), exp(g.r)*si
    n(g.i));
}

fcomplex Cpow(fcomplex z, fcomplex esp)
{
  return Cexp(Cmul(esp,Clog(z)));
}


fcomplex RCmul(double x, fcomplex a)
{
  fcomplex c;
  c.r=x*a.r;
  c.i=x*a.i;
  return c;
}
/* (C) Copr. 1986-92 Numerical Recipes Software
    A2.>$Y0%9j. */

/*Gamma Routines*/

double Carg(fcomplex a)       /* z complex in [-
    pi,pi] */
{
  double x, y, t;

  x=a.r;
  y=a.i;
  t=atan2(y,x);
  return t;
}

fcomplex Cgamma(fcomplex a)   /* Valeur de gamma(
    z) pour Re(z)!=-k en */
{                             /* utilisant l'app
    roximation de LANCZOS*/
  /* qui donne gamma(z+1) et on divise par z*/
  fcomplex Cun, z, z0, z2, z3, z4, z5, z6, z7, z8
    , z9, z10, z11, z12, z13, z14, z15, z16, zzz;
  double gam, theta, rho, p1, p2;
```

```
int i, NN, test;

Cun=Complex(1.0, 0.0);
gam=5.5;

z0=a;
NN=0;
test=0;

if (z0.r > 0.0){
z=z0;
test=0;
} else {
NN=(int)floor(z0.r);
z16=RCmul((double)NN, Cun);
z=Csub(z0, z16);
test=1;
}


z2=Complex(gam, 0.0);
z3=Cadd(z, z2);
theta=Carg(z3);
rho=Cabs(z3);
z4=Complex(0.5, 0.0);
z5=Cadd(z, z4);
p1=z5.r*log(rho)-theta*z5.i;
p2=z5.r*theta+log(rho)*z5.i;
z6=Complex(exp(p1)*cos(p2), exp(p1)*sin(p2));
z7=Complex(exp(-z3.r)*cos(-z3.i), exp(-z3.r)*si
  n(-z3.i));
z8=RCmul(sqrt(2*PI), Cmul(z6, z7));
z9=Cadd(z8, RCmul(0.000000000190015, z8));
z10=Cdiv(RCmul(76.18009172947146, z8), Cadd(z,
  RCmul(1.0,Cun)));
z11=Cdiv(RCmul(-86.50532032941677, z8), Cadd(z,
   RCmul(2.0,Cun)));
z12=Cdiv(RCmul(24.01409824083091, z8), Cadd(z,
  RCmul(3.0,Cun)));
z13=Cdiv(RCmul(-1.231739572450155, z8), Cadd(z,
   RCmul(4.0,Cun)));
```

```
  z14=Cdiv(RCmul(0.01, RCmul(0.1208650973866179,
    z8)), Cadd(z, RCmul(5.0,Cun)));
  z15=Cdiv(RCmul(0.00001,RCmul(-0.5395239384953,
    z8)), Cadd(z, RCmul(6.0,Cun)));
  z9=Cadd(z9, z10);
  z9=Cadd(z9, z11);
  z9=Cadd(z9, z12);
  z9=Cadd(z9, z13);
  z9=Cadd(z9, z14);
  z9=Cadd(z9, z15);


  if (test==1)
  {
  fcomplex aa, bb;

  bb=Cun;
  aa=z0;
  for(i=1;i<=-NN;i++){
    aa=Cadd(Cun, aa);
    bb=Cmul(bb, aa);
  }
  z9=Cdiv(z9, bb);
  zzz=z9;
  } else {
  zzz=z9;
  }
  zzz=Cdiv(zzz,z0);

  return zzz;
}


double Cnp(int n, int p)    /* Donne C(n,p) avec
    une erreue <=1 pour de */
{                           /*          grandes
    valeurs de n            */

  double z, iter;
  int i;
  z=0.0;
```

```
  if ((n-p<= -1) || (n<0) || (p<0)){
  return z;
  }
  else{
  if (p==0)
    z=1.0;
  else{
    z=1.0;
    iter=z;
    i=0;
    while(i<=p-1){
    iter=iter*(double)(n-i)/(p-i);
    i=i+1;
    }
    if ((iter-floor(iter))>=0.5)
    z=floor(iter)+1;
    else
    z=floor(iter);
  }
  }

  return z;
}

double fact(int n)       /* Donne factoriel(n) en
     double precision */
{
  int i;
  double z;

  z=1.0;

  if (n<=0)
  {
  z=1.0;
  }
  else
  {
  for(i=1;i<=n;i++)
    z=z*i;
```

```
  }

  return z;
}

fcomplex cgammln(fcomplex xx)
{
  fcomplex x,y,tmp,ser;
  static double cof[6]={76.18009172947146,-86.50
    532032941677,
    24.01409824083091,-1.231739572450155,
    0.1208650973866179e-2,-0.5395239384953e-5};
  int j;
  fcomplex sq2pg = Complex(2.5066282746310005,0)
    ;

  y=x=xx;
  tmp=Cadd(x, Complex(5.5,0));
  tmp = Csub( Cmul(Cadd(x, Complex(0.5,0)), Clog
    (tmp)),tmp);
  ser= Complex(1.000000000190015, 0.0);

  for (j=0;j<=5;j++)
  {
    y=Cadd(y,CUNO);
    ser = Cadd(ser, Cdiv(Complex(cof[j],0),y));
  }

  ser =Cmul(sq2pg, ser);

  return Cadd(Clog(Cdiv(ser,x)),tmp);
}
```

References