**Anhang 8**

zum Gutachten DESY ZFitter_GFitter vom 17.03.2014

Lübeck, den 17. März 2014

U. Obermöller

*Dr. Ulrich Obermöller*
*-Sachverständiger-*

Von der Handelskammer Hamburg
Dr.
Ulrich Obermöller
Sachverständiger für
Systeme und Anwendungen
der Informationsverarbeitung,
insbesondere Individualsoftware
und
Software-Qualitätssicherung
öffentlich bestellt und vereidigt

**Hinweis:**

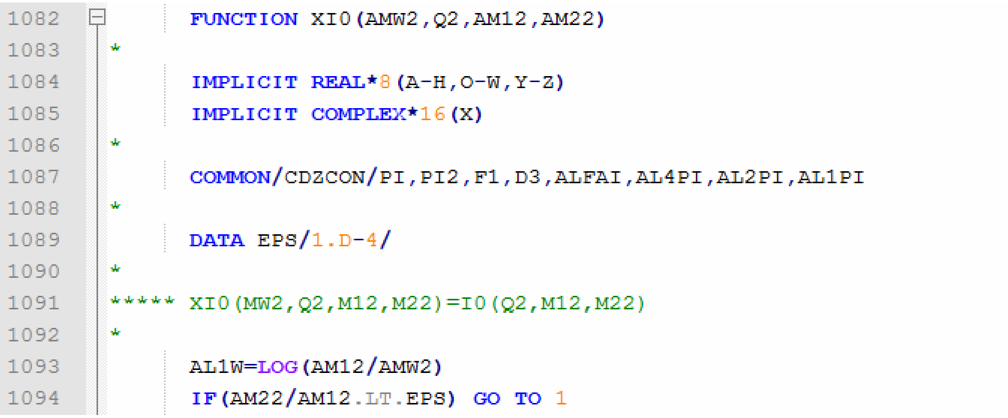
Kommentare mit Hinweisen auf ZFitter sind grün markiert

Übereinstimmungen sind gelb markiert.

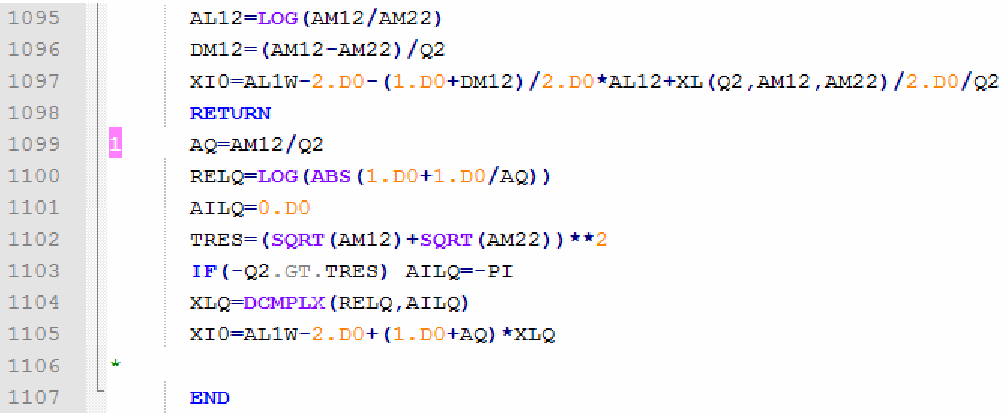
Auffällige Stellen bzw. Anmerkungen sind violett markiert

```
/*************************
 * Package: GSM
 * Class  : ZMath                                       *
 *                                                  *
 * Description:                                         *
 *      Auxillary Functions, for ZFITTER option                  *
 *                                                  *
 * Sources:                                            *
 *       - Z. Phys. C44, 493-502 (1989)                    *
 *      - Bardin et al., ZFitter package dizet6_42.f              *
 *                                                  *
 * This class also contains code lines ported to C++ from the Fortran package     *
 * ZFITTER                                              *
 *                                                  *
 *******************************************************************************/

#include "TMath.h"
#include <iostream>

#include "GSM/ZMath.h"

using namespace std;

// reference Z. Phys. C44, 493-502 (1989)
// to all functions were added extra features
// to avoid divergences

// eq. (D.1) Z. Phys. C44, 493-502 (1989)
// see dizet6_42.f line 1082-1107
std::complex<Double_t> GSM::ZMath::I0( const Double_t& MW2, const Double_t& Q2, const Double_t& m12, const Double_t& m22 )
{
  std::complex<Double_t> i0(0,0);


  if (m22/m12 < 1e-4) {
    Double_t RELQ = TMath::Log(TMath::Abs(1.0 + Q2/m12));
    if (-Q2 > Gfitter::GMath::IPow(sqrt(m12)+sqrt(m22),2)) {
      std::complex<Double_t> LQ (RELQ,-TMath::Pi());
      i0 = TMath::Log(m12/MW2) - 2.0 + (1.0 + m12/Q2)*LQ;
    }
    else i0 = TMath::Log(m12/MW2) - 2.0 + (1.0 + m12/Q2)*RELQ;
  }
```

Match 1



```
1082        FUNCTION XI0(AMW2,Q2,AM12,AM22)
1083  *
1084        IMPLICIT REAL*8(A-H,O-W,Y-Z)
1085        IMPLICIT COMPLEX*16(X)
1086  *
1087        COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1088  *
1089        DATA EPS/1.D-4/
1090  *
1091  ***** XI0(MW2,Q2,M12,M22)=I0(Q2,M12,M22)
1092  *
1093        AL1W=LOG(AM12/AMW2)
1094        IF(AM22/AM12.LT.EPS) GO TO 1
```



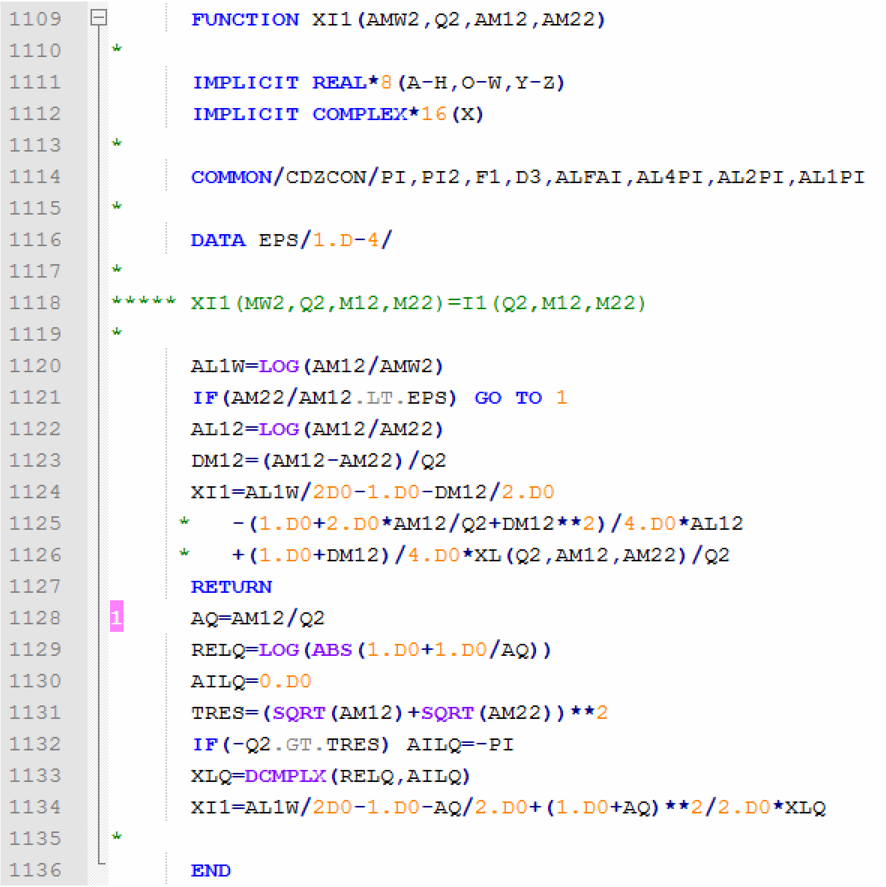
```
1095        AL12=LOG(AM12/AM22)
1096        DM12=(AM12-AM22)/Q2
1097        XI0=AL1W-2.D0-(1.D0+DM12)/2.D0*AL12+XL(Q2,AM12,AM22)/2.D0/Q2
1098        RETURN
1099  1     AQ=AM12/Q2
1100        RELQ=LOG(ABS(1.D0+1.D0/AQ))
1101        AILQ=0.D0
1102        TRES=(SQRT(AM12)+SQRT(AM22))**2
1103        IF(-Q2.GT.TRES) AILQ=-PI
1104        XLQ=DCMPLX(RELQ,AILQ)
1105        XI0=AL1W-2.D0+(1.D0+AQ)*XLQ
1106  *
1107        END
```

```
  else {
    i0 = TMath::Log(m12/MW2) - 2.0 - (1.0/2.0 + (m12-m22)/(2.0*Q2))*TMath::Log(m12/m22) + 1.0/(2.0*Q2)*L(Q2,m12,m22);
  }
  return i0;
}
```

Ende Match 1

```
// eq. (D.2) Z. Phys. C44, 493-502 (1989)
// see dizet6_42.f line 1109-1136
std::complex<Double_t> GSM::ZMath::I1( const Double_t& MW2, const Double_t& Q2, const Double_t& m12, const Double_t& m22 )
{
  std::complex<Double_t> i1(0,0);

  if (m22/m12 < 1e-4) {

    Double_t RELQ = TMath::Log(TMath::Abs(1.0 + Q2/m12));
    if (-Q2 > Gfitter::GMath::IPow(sqrt(m12)+sqrt(m22),2)) {
      std::complex<Double_t> LQ (RELQ,-TMath::Pi());
      i1 = 0.5*TMath::Log(m12/MW2) - 1.0 - m12/(2.0*Q2) + (1.0 + m12/Q2)*1/2.0*LQ;
    }
    else i1 = 0.5*TMath::Log(m12/MW2) - 1.0 - m12/(2.0*Q2) + (1.0 + m12/Q2)*1/2.0*RELQ;
  }
  else {
    i1 = 0.5*TMath::Log(m12/MW2) - 1.0 - (m12-m22)/(2.0*Q2)
      - (1.0 + 2.0*m12/Q2 + Gfitter::GMath::IPow((m12-m22),2)/(Q2*Q2))*1/4.0*TMath::Log(m12/m22)
      + (1.0 + (m12-m22)/Q2)*1.0/(4.0*Q2)*L(Q2,m12,m22);
  }
  return i1;
}
```

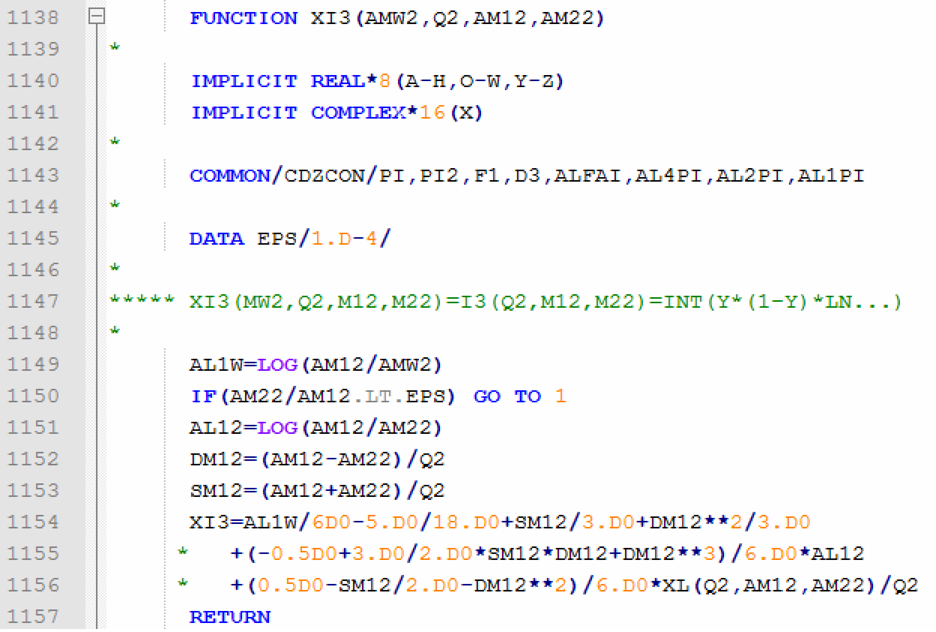
Match 2



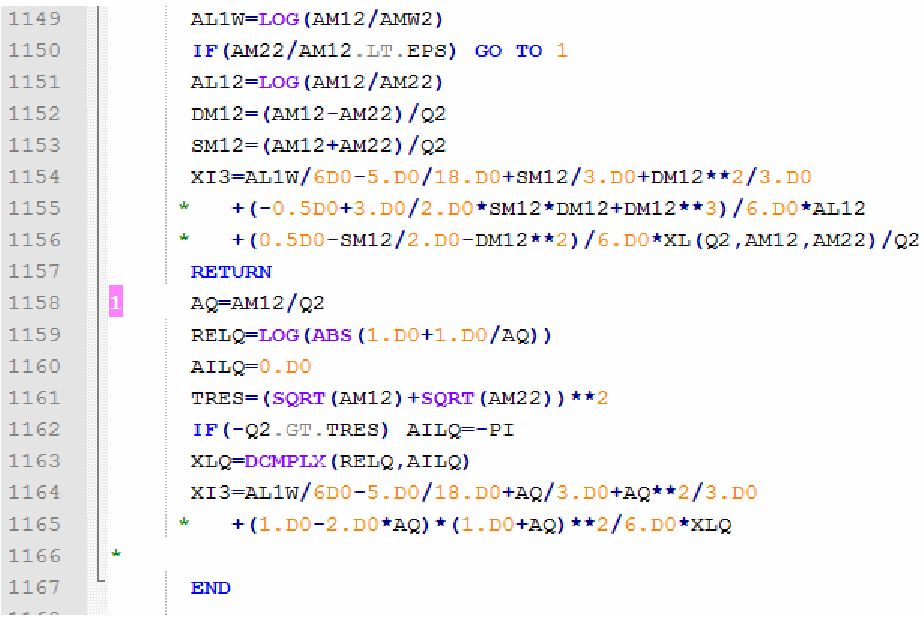
```
1109       FUNCTION XI1(AMW2,Q2,AM12,AM22)
1110 *
1111       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1112       IMPLICIT COMPLEX*16(X)
1113 *
1114       COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1115 *
1116       DATA EPS/1.D-4/
1117 *
1118 ***** XI1(MW2,Q2,M12,M22)=I1(Q2,M12,M22)
1119 *
1120       AL1W=LOG(AM12/AMW2)
1121       IF(AM22/AM12.LT.EPS) GO TO 1
1122       AL12=LOG(AM12/AM22)
1123       DM12=(AM12-AM22)/Q2
1124       XI1=AL1W/2D0-1.D0-DM12/2.D0
1125      *   -(1.D0+2.D0*AM12/Q2+DM12**2)/4.D0*AL12
1126      *   +(1.D0+DM12)/4.D0*XL(Q2,AM12,AM22)/Q2
1127       RETURN
1128 1     AQ=AM12/Q2
1129       RELQ=LOG(ABS(1.D0+1.D0/AQ))
1130       AILQ=0.D0
1131       TRES=(SQRT(AM12)+SQRT(AM22))**2
1132       IF(-Q2.GT.TRES) AILQ=-PI
1133       XLQ=DCMPLX(RELQ,AILQ)
1134       XI1=AL1W/2D0-1.D0-AQ/2.D0+(1.D0+AQ)**2/2.D0*XLQ
1135 *
1136       END
```

Match 3

```
// eq. (D.3) Z. Phys. C44, 493-502 (1989)
// see dizet6_42.f line 1138-1167
std::complex<Double_t> GSM::ZMath::I3( const Double_t& MW2, const Double_t& Q2, const Double_t& m12, const Double_t& m22 )
{
  std::complex<Double_t> i3(0,0);
  Double_t AL1W  = TMath::Log(m12/MW2);

  if (m22/m12 < 1e-4) {
    Double_t AQ = m12/Q2;
    Double_t RELQ = TMath::Log(TMath::Abs(1.0 + 1.0/AQ));
    if (-Q2 > Gfitter::GMath::IPow(sqrt(m12)+sqrt(m22),2)) {
      std::complex<Double_t> LQ (RELQ,-TMath::Pi());
      i3 = AL1W/6.0 - 5.0/18.0 + AQ/3.0 + AQ*AQ/3.0 + (1.0 - 2.0*AQ)*(1.0 + AQ)*(1.0 + AQ)*1.0/6.0*LQ;
    }
    else {
      i3 = ( 1.0/6.0*TMath::Log(m12/MW2) - 5.0/18.0 + m12/(3.0*Q2) + m12*m12/(3.0*Q2*Q2)
             + (1.0 + 2.0*m12/Q2)*Gfitter::GMath::IPow(1.0 + m12/Q2,2)*1.0/6.0*RELQ );
    }
  }
  else {
    i3 = ( 1.0/6.0*TMath::Log(m12/MW2) - 5.0/18.0 + (m12+m22)/(3.0*Q2) + 1/3.0*Gfitter::GMath::IPow((m12-m22)/Q2,2)
           + (-0.5 + 3.0/2.0*(m12+m22)*(m12-m22)/(Q2*Q2) + Gfitter::GMath::IPow((m12-m22)/Q2,3))*1/6.0*TMath::Log(m12/m22)
           + (0.5 - (m12+m22)/(2.0*Q2) - Gfitter::GMath::IPow((m12-m22)/Q2,2))*1.0/(6.0*Q2)*L(Q2,m12,m22) );
  }

  return i3;
}
```



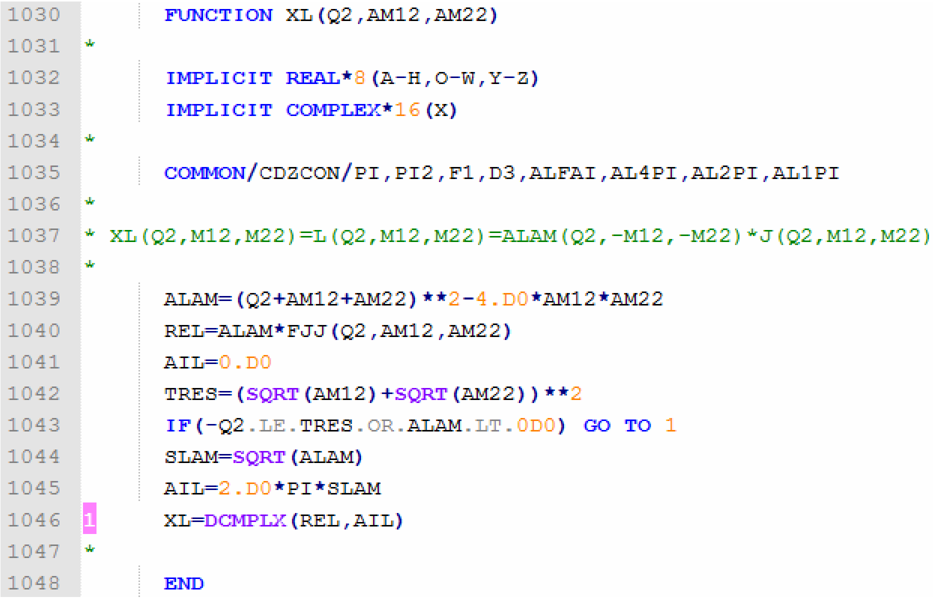
```
1138       FUNCTION XI3(AMW2,Q2,AM12,AM22)
1139 *
1140       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1141       IMPLICIT COMPLEX*16(X)
1142 *
1143       COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1144 *
1145       DATA EPS/1.D-4/
1146 *
1147 ***** XI3(MW2,Q2,M12,M22)=I3(Q2,M12,M22)=INT(Y*(1-Y)*LN...)
1148 *
1149       AL1W=LOG(AM12/AMW2)
1150       IF(AM22/AM12.LT.EPS) GO TO 1
1151       AL12=LOG(AM12/AM22)
1152       DM12=(AM12-AM22)/Q2
1153       SM12=(AM12+AM22)/Q2
1154       XI3=AL1W/6D0-5.D0/18.D0+SM12/3.D0+DM12**2/3.D0
1155      *   +(-0.5D0+3.D0/2.D0*SM12*DM12+DM12**3)/6.D0*AL12
1156      *   +(0.5D0-SM12/2.D0-DM12**2)/6.D0*XL(Q2,AM12,AM22)/Q2
1157       RETURN
```



```
1149       AL1W=LOG(AM12/AMW2)
1150       IF(AM22/AM12.LT.EPS) GO TO 1
1151       AL12=LOG(AM12/AM22)
1152       DM12=(AM12-AM22)/Q2
1153       SM12=(AM12+AM22)/Q2
1154       XI3=AL1W/6D0-5.D0/18.D0+SM12/3.D0+DM12**2/3.D0
1155      *   +(-0.5D0+3.D0/2.D0*SM12*DM12+DM12**3)/6.D0*AL12
1156      *   +(0.5D0-SM12/2.D0-DM12**2)/6.D0*XL(Q2,AM12,AM22)/Q2
1157       RETURN
1158  1    AQ=AM12/Q2
1159       RELQ=LOG(ABS(1.D0+1.D0/AQ))
1160       AILQ=0.D0
1161       TRES=(SQRT(AM12)+SQRT(AM22))**2
1162       IF(-Q2.GT.TRES) AILQ=-PI
1163       XLQ=DCMPLX(RELQ,AILQ)
1164       XI3=AL1W/6D0-5.D0/18.D0+AQ/3.D0+AQ**2/3.D0
1165      *   +(1.D0-2.D0*AQ)*(1.D0+AQ)**2/6.D0*XLQ
1166 *
1167       END
```

Match 4

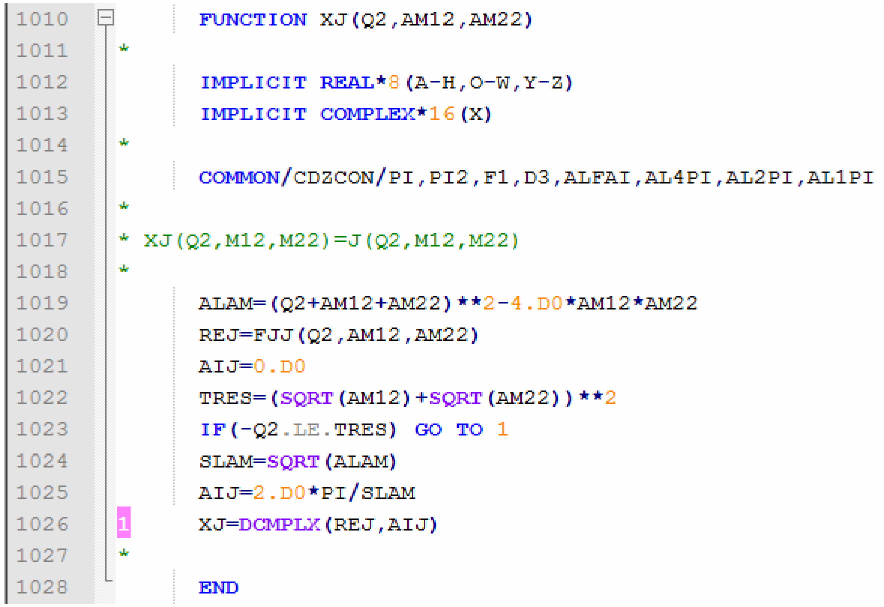
```
// eq. (A.8) Z. Phys. C44, 493-502 (1989)
// see dizet6_42.f line 1030-1048
std::complex<Double_t> GSM::ZMath::L( const Double_t& Q2, const Double_t& m12, const Double_t& m22)
{
  Double_t LAM = (Q2 + m12 + m22)*(Q2 + m12 + m22) - 4.0*m12*m22;
  std::complex<Double_t> REL = LAM*JJ(Q2,m12,m22);
  Double_t IL  = 0;
  if (-Q2 > Gfitter::GMath::IPow(sqrt(m12)+sqrt(m22),2) && LAM >= 0) IL = 2.0*TMath::Pi()*sqrt(LAM);
  std::complex<Double_t> l(real(REL),IL);

  return l;
}
```



```
1030       FUNCTION XL(Q2,AM12,AM22)
1031 *
1032       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1033       IMPLICIT COMPLEX*16(X)
1034 *
1035       COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1036 *
1037 * XL(Q2,M12,M22)=L(Q2,M12,M22)=ALAM(Q2,-M12,-M22)*J(Q2,M12,M22)
1038 *
1039       ALAM=(Q2+AM12+AM22)**2-4.D0*AM12*AM22
1040       REL=ALAM*FJJ(Q2,AM12,AM22)
1041       AIL=0.D0
1042       TRES=(SQRT(AM12)+SQRT(AM22))**2
1043       IF(-Q2.LE.TRES.OR.ALAM.LT.0D0) GO TO 1
1044       SLAM=SQRT(ALAM)
1045       AIL=2.D0*PI*SLAM
1046       XL=DCMPLX(REL,AIL)
1047 *
1048       END
```

Match 5

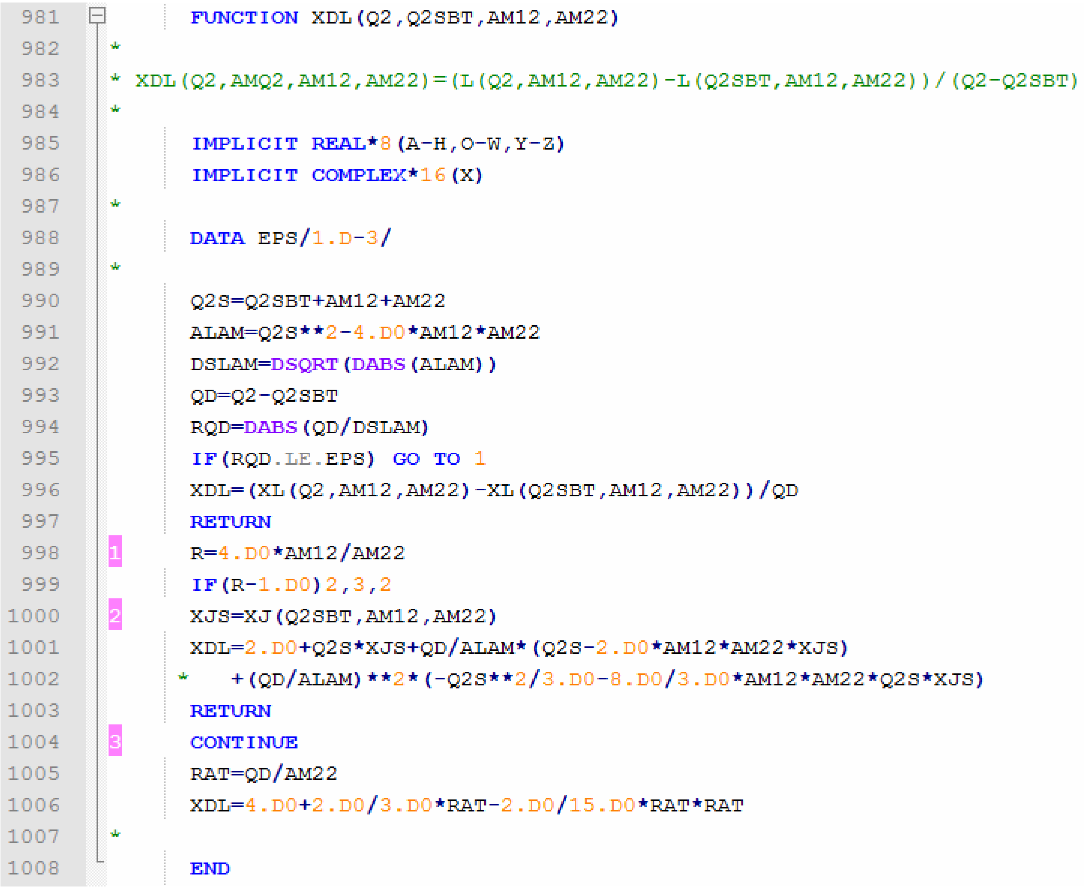
```
// eq. (C.6) Z. Phys. C44, 493-502 (1989)
// see dizet6_42.f line 1010-1028
std::complex<Double_t> GSM::ZMath::J( const Double_t& Q2, const Double_t& m12, const Double_t& m22 )
{
  Double_t LAM = (Q2 + m12 + m22)*(Q2 + m12 + m22) - 4.0*m12*m22;
  std::complex<Double_t> REJ = JJ(Q2,m12,m22);
  Double_t IJ  = 0;
  if (-Q2 > Gfitter::GMath::IPow(sqrt(m12)+sqrt(m22),2)) IJ = 2.0*TMath::Pi()/sqrt(LAM);
  std::complex<Double_t> j (real(REJ),IJ);

  return j;
}
```



```
1010       FUNCTION XJ(Q2,AM12,AM22)
1011 *
1012       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1013       IMPLICIT COMPLEX*16(X)
1014 *
1015       COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1016 *
1017 * XJ(Q2,M12,M22)=J(Q2,M12,M22)
1018 *
1019       ALAM=(Q2+AM12+AM22)**2-4.D0*AM12*AM22
1020       REJ=FJJ(Q2,AM12,AM22)
1021       AIJ=0.D0
1022       TRES=(SQRT(AM12)+SQRT(AM22))**2
1023       IF(-Q2.LE.TRES) GO TO 1
1024       SLAM=SQRT(ALAM)
1025       AIJ=2.D0*PI/SLAM
1026       XJ=DCMPLX(REJ,AIJ)
1027 *
1028       END
```

Match 6

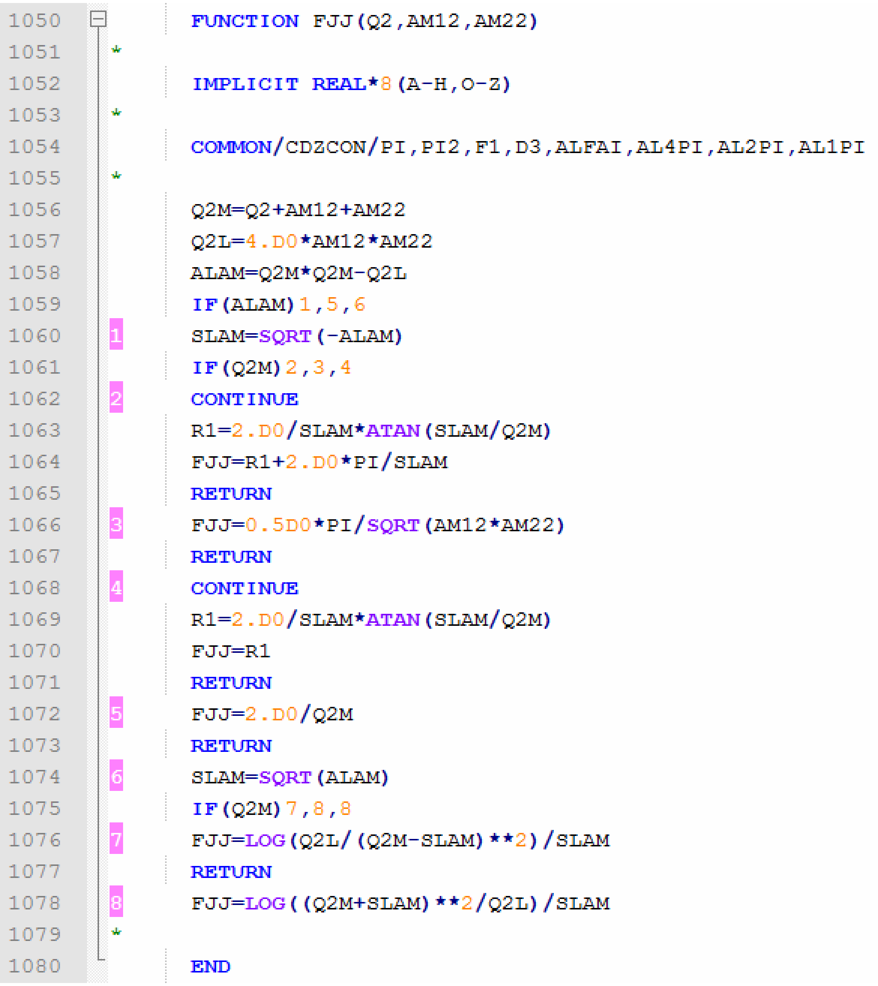
```cpp
// Delta L
// see dizet6_42.f line 981-1008
std::complex<Double_t> GSM::ZMath::DL( const Double_t& Q2, const Double_t& Q2SBT, const Double_t& m12, const Double_t& m22 )
{
   std::complex<Double_t> dl(0,0);
   Double_t Q2S   = Q2SBT + m12 + m22;
   Double_t LAM   = Q2S*Q2S - 4.0*m12*m22;
   Double_t SLAM  = TMath::Sqrt(TMath::Abs(LAM));
   Double_t QD    = Q2 -Q2SBT;
   Double_t RQD   = TMath::Abs(QD/SLAM);
   if (RQD < 1e-3) {
      Double_t R = 4.0*m12/m22;
      if (R-1.0 != 0) {
         std::complex<Double_t> JS = J(Q2SBT,m12,m22);
         dl = ( 2.0 + Q2S*JS + QD/LAM*(Q2S - 2.0*m12*m22*JS)
                 + (QD/LAM)*(QD/LAM)*(-Q2S*Q2S/3.0 - 8.0/3.0*m12*m22*Q2S*JS) );
      }
      else dl = 4.0 + 2.0/3.0*QD/m22 - 2.0/15.0*QD*QD/(m22*m22);
   }
   else dl = (L(Q2,m12,m22) - L(Q2SBT,m12,m22))/QD;

   return dl;
}
```



```fortran
981          FUNCTION XDL(Q2,Q2SBT,AM12,AM22)
982    *
983    * XDL(Q2,AMQ2,AM12,AM22)=(L(Q2,AM12,AM22)-L(Q2SBT,AM12,AM22))/(Q2-Q2SBT)
984    *
985          IMPLICIT REAL*8(A-H,O-W,Y-Z)
986          IMPLICIT COMPLEX*16(X)
987    *
988          DATA EPS/1.D-3/
989    *
990          Q2S=Q2SBT+AM12+AM22
991          ALAM=Q2S**2-4.D0*AM12*AM22
992          DSLAM=DSQRT(DABS(ALAM))
993          QD=Q2-Q2SBT
994          RQD=DABS(QD/DSLAM)
995          IF(RQD.LE.EPS) GO TO 1
996          XDL=(XL(Q2,AM12,AM22)-XL(Q2SBT,AM12,AM22))/QD
997          RETURN
998    1     R=4.D0*AM12/AM22
999          IF(R-1.D0)2,3,2
1000   2     XJS=XJ(Q2SBT,AM12,AM22)
1001         XDL=2.D0+Q2S*XJS+QD/ALAM*(Q2S-2.D0*AM12*AM22*XJS)
1002        *   +(QD/ALAM)**2*(-Q2S**2/3.D0-8.D0/3.D0*AM12*AM22*Q2S*XJS)
1003         RETURN
1004   3     CONTINUE
1005         RAT=QD/AM22
1006         XDL=4.D0+2.D0/3.D0*RAT-2.D0/15.D0*RAT*RAT
1007   *
1008         END
```

Match 7

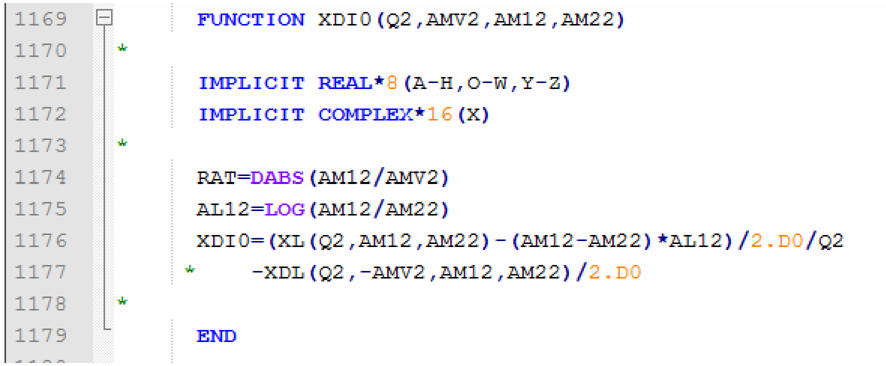
```
// Auxiliary function for JJ
// see dizet6_42.f line 1050-1080
std::complex<Double_t> GSM::ZMath::JJ( const Double_t& Q2, const Double_t& m12, const Double_t& m22 )
{
   std::complex<Double_t> jj (0,0);
   Double_t Q2M  = (Q2 + m12 + m22);
   Double_t Q2L  =  4.0*m12*m22;
   Double_t LAM  = Q2M*Q2M - Q2L;
   Double_t R1    = 0;
   Double_t SLAM = 0;

   if (LAM < 0) {
      SLAM = sqrt(-LAM);
      if (Q2M < 0) {
         R1 = 2.0/SLAM*TMath::ATan(SLAM/Q2M);
         jj = R1 + 2.0*TMath::Pi()/SLAM;
      }
      if (Q2M == 0) {
         jj = 0.5*TMath::Pi()/sqrt(m12*m22);
      }
      if (Q2M > 0) {
         R1 = 2.0/SLAM*TMath::ATan(SLAM/Q2M);
         jj = R1;
      }
   }
   if (LAM ==0) {
      jj = 2.0/Q2M;
   }
   if (LAM > 0) {
      SLAM = sqrt(LAM);
      if (Q2M < 0) {
         jj = TMath::Log(Q2L/Gfitter::GMath::IPow((Q2M-SLAM),2))/SLAM;
      }
      if (Q2M >= 0) {
         jj = TMath::Log(Gfitter::GMath::IPow((Q2M+SLAM),2)/Q2L)/SLAM;
      }
   }
   return jj;
}
```



```
1050        FUNCTION FJJ(Q2,AM12,AM22)
1051  *
1052        IMPLICIT REAL*8(A-H,O-Z)
1053  *
1054        COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
1055  *
1056        Q2M=Q2+AM12+AM22
1057        Q2L=4.D0*AM12*AM22
1058        ALAM=Q2M*Q2M-Q2L
1059        IF(ALAM)1,5,6
1060      1 SLAM=SQRT(-ALAM)
1061        IF(Q2M)2,3,4
1062      2 CONTINUE
1063        R1=2.D0/SLAM*ATAN(SLAM/Q2M)
1064        FJJ=R1+2.D0*PI/SLAM
1065        RETURN
1066      3 FJJ=0.5D0*PI/SQRT(AM12*AM22)
1067        RETURN
1068      4 CONTINUE
1069        R1=2.D0/SLAM*ATAN(SLAM/Q2M)
1070        FJJ=R1
1071        RETURN
1072      5 FJJ=2.D0/Q2M
1073        RETURN
1074      6 SLAM=SQRT(ALAM)
1075        IF(Q2M)7,8,8
1076      7 FJJ=LOG(Q2L/(Q2M-SLAM)**2)/SLAM
1077        RETURN
1078      8 FJJ=LOG((Q2M+SLAM)**2/Q2L)/SLAM
1079  *
1080        END
```

Match 8

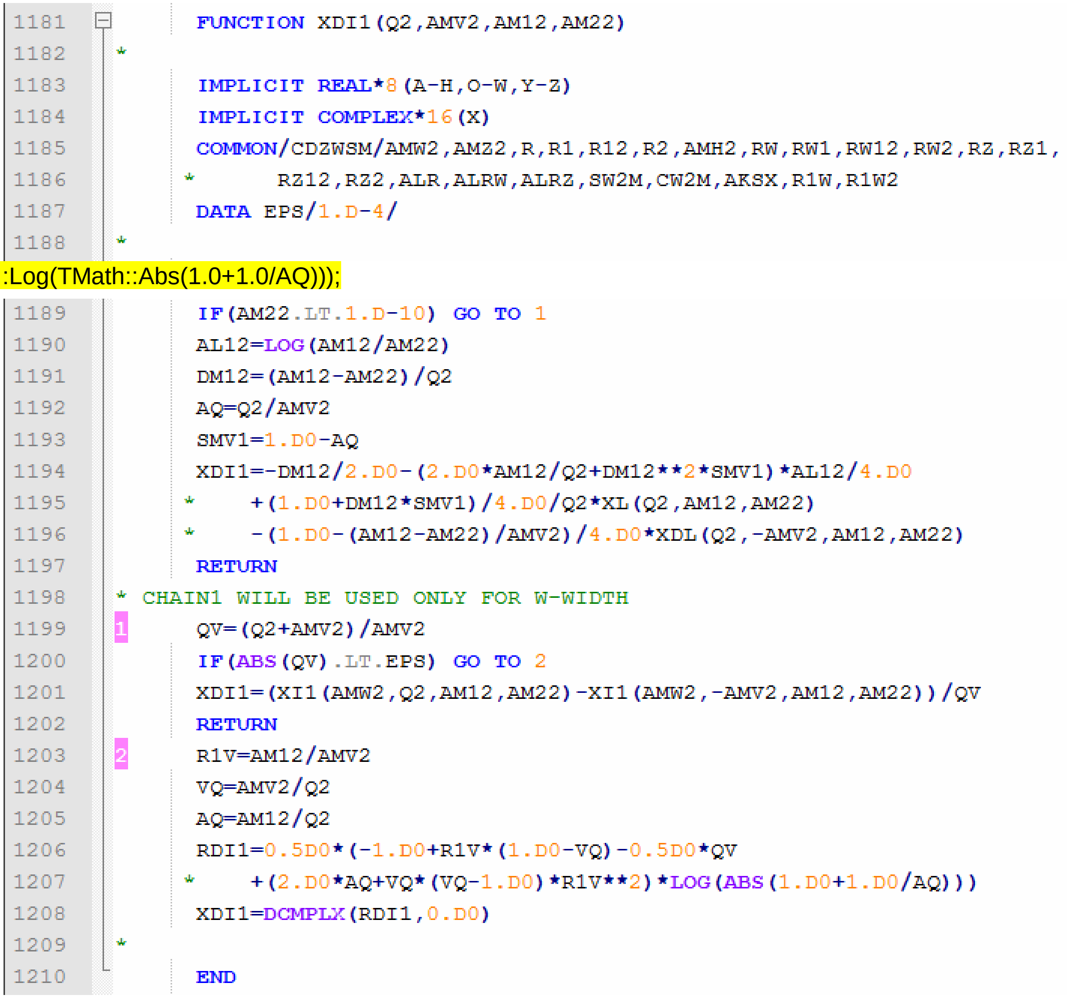
```
// Delta I0
// see dizet6_42.f line 1169-1179
std::complex<Double_t> GSM::ZMath::DI0( const Double_t& Q2, const Double_t& MV2, const Double_t& m12, const Double_t& m22 )
{
   std::complex<Double_t> di0(0,0);
   Double_t L12 = TMath::Log(m12/m22);
   di0 = (L(Q2, m12, m22) - (m12-m22)*L12)/(2.0*Q2) - DL(Q2, -MV2, m12, m22)/2.0;

   return di0;
}
```



```
1169       FUNCTION XDI0(Q2,AMV2,AM12,AM22)
1170 *
1171       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1172       IMPLICIT COMPLEX*16(X)
1173 *
1174       RAT=DABS(AM12/AMV2)
1175       AL12=LOG(AM12/AM22)
1176       XDI0=(XL(Q2,AM12,AM22)-(AM12-AM22)*AL12)/2.D0/Q2
1177      *    -XDL(Q2,-AMV2,AM12,AM22)/2.D0
1178 *
1179       END
```

Match 9

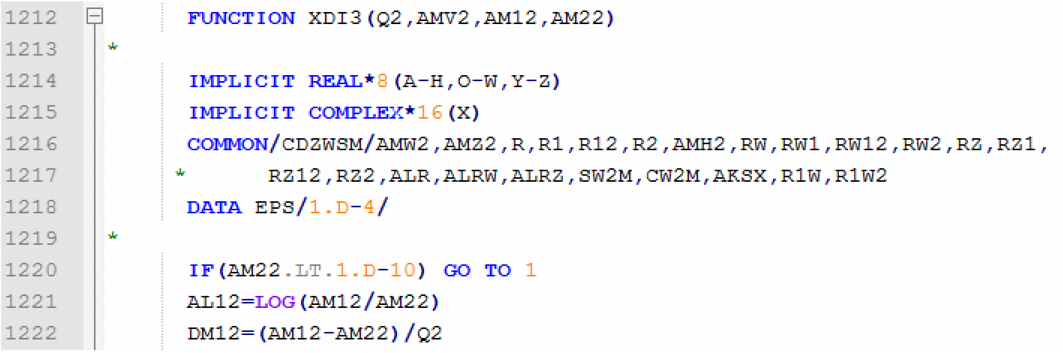
```
// Delta I1
// see dizet6_42.f line 1181-1210
std::complex<Double_t> GSM::ZMath::DI1( const Double_t& Q2, const Double_t& MV2,
                                     const Double_t& m12, const Double_t& m22, const Double_t& MW2 )
{
  std::complex<Double_t> di1(0,0);

  if (m22 < 1e-10) {
    Double_t QV = (Q2 + MV2)/MV2;
    if (TMath::Abs(QV) < 1e-4) {
      Double_t R1V = m12/MV2;
      Double_t VQ  = MV2/Q2;
      Double_t AQ  = m12/Q2;
     di1 = 0.5*(-1.0 + R1V*(1.0-VQ) - 0.5*QV + (2.0*AQ + VQ*(VQ-1.0)*R1V*R1V)*TMath::Log(TMath::Abs(1.0+1.0/AQ)));
    }
    else di1 = (I1(MW2,Q2,m12,m22) - I1(MW2,-MV2,m12,m22))/QV;
  }
  else {
    Double_t AL12 = TMath::Log(m12/m22);
    Double_t Dm12 = (m12-m22)/Q2;
    Double_t AQ   = Q2/MV2;
    Double_t SMV1 = 1.0 -AQ;
    di1 = ( -Dm12/2.0 - (2.0*m12/Q2 + Dm12*Dm12*SMV1)*AL12/4.0
            + (1.0 + Dm12*SMV1)/(4.0*Q2)*L(Q2,m12,m22)
            - (1.0 - (m12-m22)/MV2)/4.0*DL(Q2,-MV2,m12,m22) );
  }
  return di1;
}
```



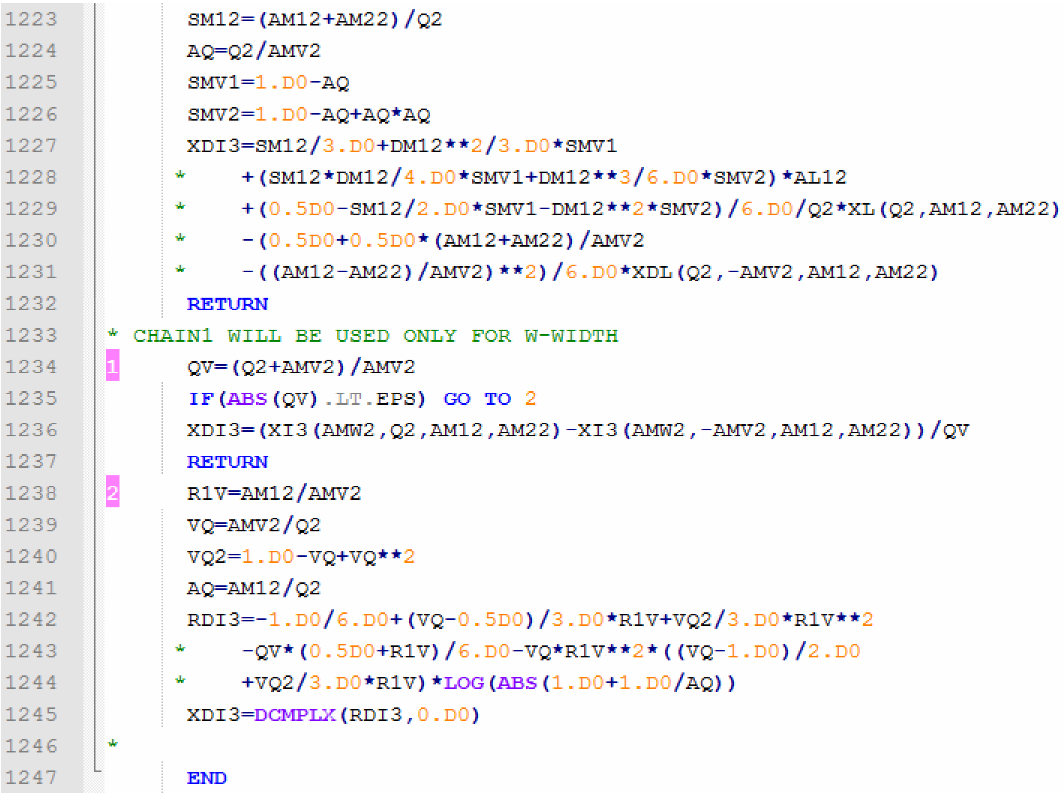
```
1181        FUNCTION XDI1(Q2,AMV2,AM12,AM22)
1182  *
1183        IMPLICIT REAL*8(A-H,O-W,Y-Z)
1184        IMPLICIT COMPLEX*16(X)
1185        COMMON/CDZWSM/AMW2,AMZ2,R,R1,R12,R2,AMH2,RW,RW1,RW12,RW2,RZ,RZ1,
1186       *      RZ12,RZ2,ALR,ALRW,ALRZ,SW2M,CW2M,AKSX,R1W,R1W2
1187        DATA EPS/1.D-4/
1188  *
1189        IF(AM22.LT.1.D-10) GO TO 1
1190        AL12=LOG(AM12/AM22)
1191        DM12=(AM12-AM22)/Q2
1192        AQ=Q2/AMV2
1193        SMV1=1.D0-AQ
1194        XDI1=-DM12/2.D0-(2.D0*AM12/Q2+DM12**2*SMV1)*AL12/4.D0
1195       *    +(1.D0+DM12*SMV1)/4.D0/Q2*XL(Q2,AM12,AM22)
1196       *    -(1.D0-(AM12-AM22)/AMV2)/4.D0*XDL(Q2,-AMV2,AM12,AM22)
1197        RETURN
1198  * CHAIN1 WILL BE USED ONLY FOR W-WIDTH
1199  1     QV=(Q2+AMV2)/AMV2
1200        IF(ABS(QV).LT.EPS) GO TO 2
1201        XDI1=(XI1(AMW2,Q2,AM12,AM22)-XI1(AMW2,-AMV2,AM12,AM22))/QV
1202        RETURN
1203  2     R1V=AM12/AMV2
1204        VQ=AMV2/Q2
1205        AQ=AM12/Q2
1206        RDI1=0.5D0*(-1.D0+R1V*(1.D0-VQ)-0.5D0*QV
1207       *    +(2.D0*AQ+VQ*(VQ-1.D0)*R1V**2)*LOG(ABS(1.D0+1.D0/AQ)))
1208        XDI1=DCMPLX(RDI1,0.D0)
1209  *
1210        END
```

Match 10

```cpp
// Delta I3
// see dizet6_42.f line 1212-1247
std::complex<Double_t> GSM::ZMath::DI3( const Double_t& Q2, const Double_t& MV2, const Double_t& m12,
                                       const Double_t& m22, const Double_t& MW2 )
{
  std::complex<Double_t> di3(0,0);

  if (m22 < 1e-10) {
    Double_t QV = (Q2 + MV2)/MV2;
    if (TMath::Abs(QV) < 1e-4) {
      Double_t R1V = m12/MV2;
      Double_t VQ  = MV2/Q2;
      Double_t VQ2 = 1.0 - VQ + VQ*VQ;
      Double_t AQ  = m12/Q2;
      di3 = ( -1.0/6.0 + (VQ-0.5)/3.0*R1V + VQ2/3.0*R1V*R1V
             -QV*(0.5+R1V)/6.0 - VQ*R1V*R1V*((VQ-1.0)/2.0 + VQ2/3.0*R1V)*TMath::Log(TMath::Abs(1.0+1.0/AQ)) );
    }
    else di3 = (I3(MW2,Q2,m12,m22) - I1(MW2,-MV2,m12,m22))/QV;
  }
  else {
    Double_t AL12 = TMath::Log(m12/m22);
    Double_t Dm12 = (m12-m22)/Q2;
    Double_t Sm12 = (m12+m22)/Q2;
    Double_t AQ   = Q2/MV2;
    Double_t SMV1 = 1.0 - AQ;
    Double_t SMV2 = 1.0 - AQ + AQ*AQ;
    di3 = ( Sm12/3.0 + Dm12*Dm12/3.0*SMV1
           + (Sm12*Dm12/4.0*SMV1 + Gfitter::GMath::IPow(Dm12,3)/6.0*SMV2)*AL12
           + (0.5 - Sm12/2.0*SMV1 - Dm12*Dm12*SMV2)/(6.0*Q2)*L(Q2,m12,m22)
           - (0.5 + 0.5*(m12+m22)/MV2
             - Gfitter::GMath::IPow((m12-m22)/MV2,2))/6.0*DL(Q2,-MV2,m12,m22) );
  }
  return di3;
}
```



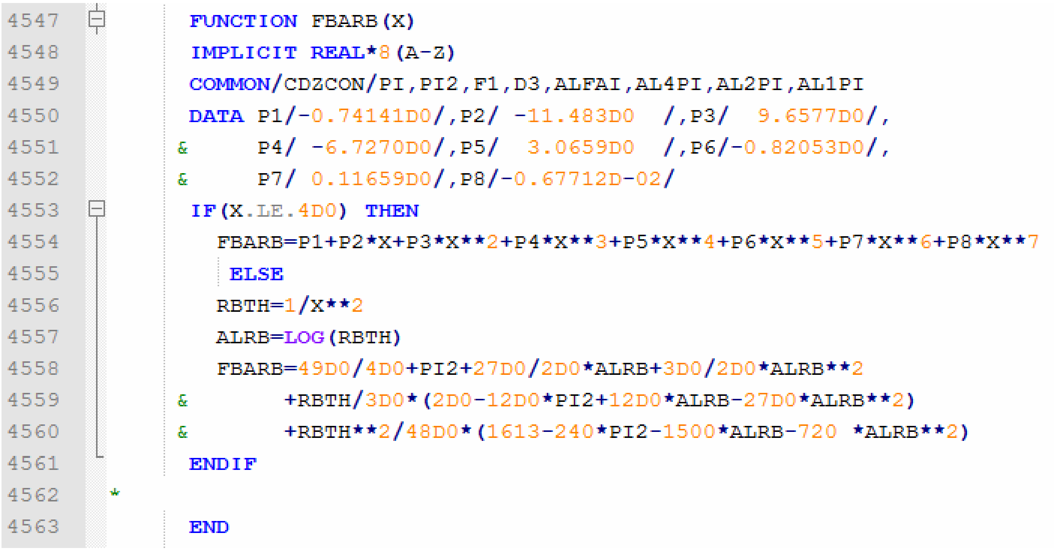
```fortran
1212       FUNCTION XDI3(Q2,AMV2,AM12,AM22)
1213 *
1214       IMPLICIT REAL*8(A-H,O-W,Y-Z)
1215       IMPLICIT COMPLEX*16(X)
1216       COMMON/CDZWSM/AMW2,AMZ2,R,R1,R12,R2,AMH2,RW,RW1,RW12,RW2,RZ,RZ1,
1217      *      RZ12,RZ2,ALR,ALRW,ALRZ,SW2M,CW2M,AKSX,R1W,R1W2
1218       DATA EPS/1.D-4/
1219 *
1220       IF(AM22.LT.1.D-10) GO TO 1
1221       AL12=LOG(AM12/AM22)
1222       DM12=(AM12-AM22)/Q2
```



```fortran
1223       SM12=(AM12+AM22)/Q2
1224       AQ=Q2/AMV2
1225       SMV1=1.D0-AQ
1226       SMV2=1.D0-AQ+AQ*AQ
1227       XDI3=SM12/3.D0+DM12**2/3.D0*SMV1
1228      *    +(SM12*DM12/4.D0*SMV1+DM12**3/6.D0*SMV2)*AL12
1229      *    +(0.5D0-SM12/2.D0*SMV1-DM12**2*SMV2)/6.D0/Q2*XL(Q2,AM12,AM22)
1230      *    -(0.5D0+0.5D0*(AM12+AM22)/AMV2
1231      *    -((AM12-AM22)/AMV2)**2)/6.D0*XDL(Q2,-AMV2,AM12,AM22)
1232       RETURN
1233 * CHAIN1 WILL BE USED ONLY FOR W-WIDTH
1234 1     QV=(Q2+AMV2)/AMV2
1235       IF(ABS(QV).LT.EPS) GO TO 2
1236       XDI3=(XI3(AMW2,Q2,AM12,AM22)-XI3(AMW2,-AMV2,AM12,AM22))/QV
1237       RETURN
1238 2     R1V=AM12/AMV2
1239       VQ=AMV2/Q2
1240       VQ2=1.D0-VQ+VQ**2
1241       AQ=AM12/Q2
1242       RDI3=-1.D0/6.D0+(VQ-0.5D0)/3.D0*R1V+VQ2/3.D0*R1V**2
1243      *    -QV*(0.5D0+R1V)/6.D0-VQ*R1V**2*((VQ-1.D0)/2.D0
1244      *    +VQ2/3.D0*R1V)*LOG(ABS(1.D0+1.D0/AQ))
1245       XDI3=DCMPLX(RDI3,0.D0)
1246 *
1247       END
```

Match 11

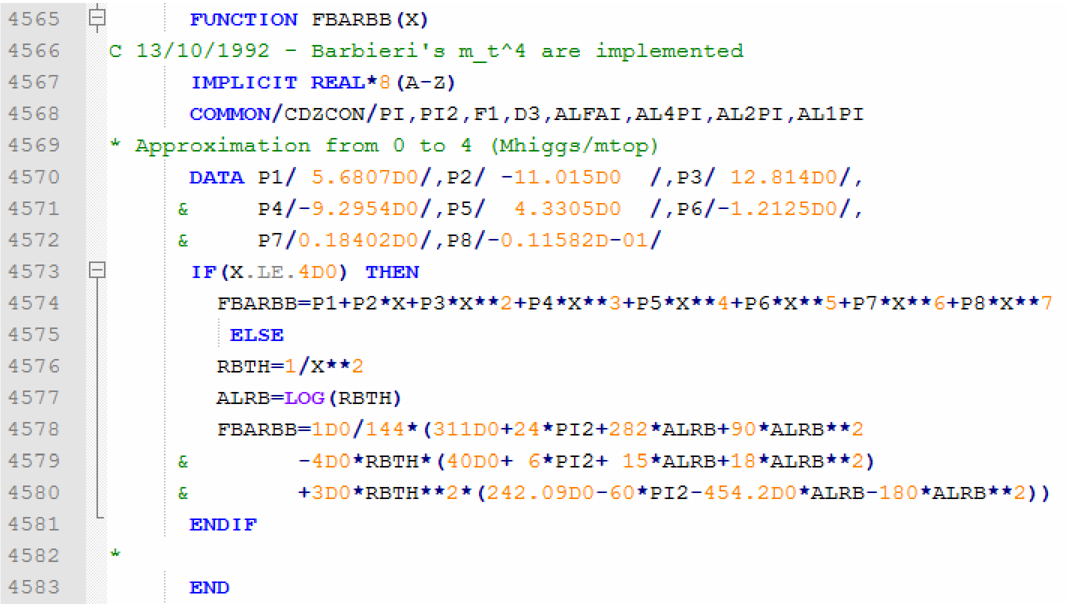
```
// ------- mtop^4 contribution from Barbieri et. al. ---------
// ------- Nucl. Phys. B409(1993) 105-107 --------------------

// see dizet6_42.f line 4547-4563
Double_t GSM::ZMath::BarbMc( const Double_t& x )
{
   Double_t P[8] = {-0.74141, -11.483, 9.6577, -6.727, 3.0659, -0.82053, 0.11659, -0.67712e-2};
   Double_t Barb = 0;
   if (x < 4) {
      for (Int_t i = 0; i < 8; i++) {  Barb += P[i]*Gfitter::GMath::IPow(x,i); }
   }
   else {
      Double_t Pi2      = Gfitter::GMath::IPow(TMath::Pi(),2);
      Double_t Inv2     = 1/(x*x);
      Double_t LogInv2  = TMath::Log(Inv2);
      Barb              = ( 49/4.0 + Pi2 + 27/2.0*LogInv2 + 3/2.0*LogInv2*LogInv2
                          + Inv2/3.0*(2.0 - 12.0*Pi2 + 12.0*LogInv2 -27.0*LogInv2*LogInv2)
                          + Inv2*Inv2/48.0*(1613.0 - 240.0*Pi2 - 1500.0*LogInv2 - 720.0*LogInv2*LogInv2) );
   }
   return Barb;
}
```



```
4547        FUNCTION FBARB(X)
4548        IMPLICIT REAL*8(A-Z)
4549        COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
4550        DATA P1/-0.74141D0/,P2/ -11.483D0  /,P3/  9.6577D0/,
4551       &     P4/ -6.7270D0/,P5/  3.0659D0  /,P6/-0.82053D0/,
4552       &     P7/ 0.11659D0/,P8/-0.67712D-02/
4553        IF(X.LE.4D0) THEN
4554          FBARB=P1+P2*X+P3*X**2+P4*X**3+P5*X**4+P6*X**5+P7*X**6+P8*X**7
4555           ELSE
4556          RBTH=1/X**2
4557          ALRB=LOG(RBTH)
4558          FBARB=49D0/4D0+PI2+27D0/2D0*ALRB+3D0/2D0*ALRB**2
4559       &         +RBTH/3D0*(2D0-12D0*PI2+12D0*ALRB-27D0*ALRB**2)
4560       &         +RBTH**2/48D0*(1613-240*PI2-1500*ALRB-720 *ALRB**2)
4561        ENDIF
4562  *
4563        END
```

Match 12

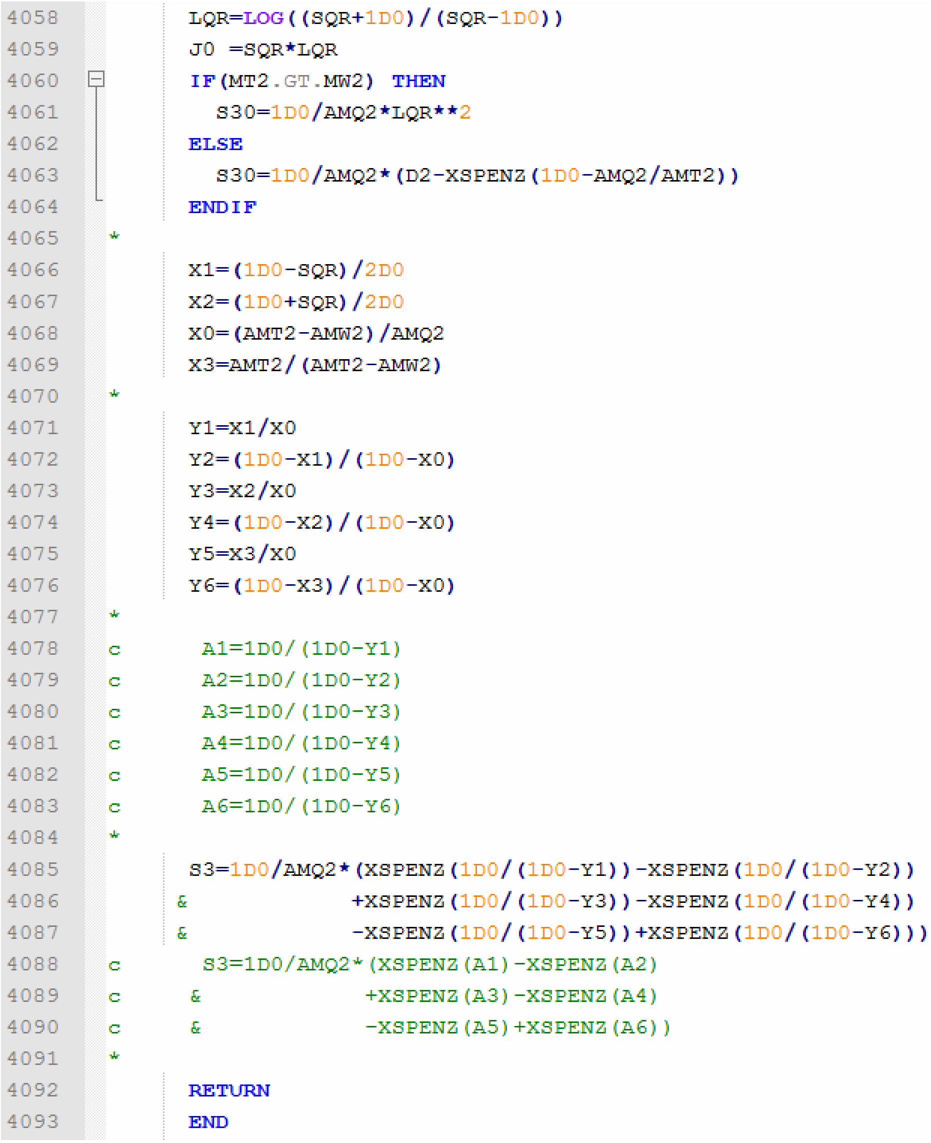
```
// see dizet6_42.f line 4565-4583
Double_t GSM::ZMath::BarbMb( const Double_t& x )
{
  Double_t P[8] = {5.6807, -11.015, 12.814, -9.2954, 4.3305, -1.2125, 0.18402, -0.11582e-1};
  Double_t Barb = 0;
  if (x < 4) {
    for (Int_t i = 0; i < 8; i++) {  Barb += P[i]*Gfitter::GMath::IPow(x,i); }
  }
  else {
    Double_t Pi2      = Gfitter::GMath::IPow(TMath::Pi(),2);
    Double_t Inv2     = 1/(x*x);
    Double_t LogInv2  = TMath::Log(Inv2);
    Barb              = 1/144.0*( 311.0 + 24.0*Pi2 + 282.0*LogInv2 + 90.0*LogInv2*LogInv2
                                  - 4.0*Inv2*(40.0 + 6.0*Pi2 + 15.0*LogInv2 + 18.0*LogInv2*LogInv2)
                                  + 3.0*Inv2*Inv2*(242.09 - 60.0*Pi2 - 454.2*LogInv2 - 180.0*LogInv2*LogInv2) );
  }
  return Barb;
}
```



```
4565        FUNCTION FBARBB(X)
4566  C 13/10/1992 - Barbieri's m_t^4 are implemented
4567        IMPLICIT REAL*8(A-Z)
4568        COMMON/CDZCON/PI,PI2,F1,D3,ALFAI,AL4PI,AL2PI,AL1PI
4569  * Approximation from 0 to 4 (Mhiggs/mtop)
4570        DATA P1/ 5.6807D0/,P2/ -11.015D0  /,P3/ 12.814D0/,
4571       &     P4/-9.2954D0/,P5/  4.3305D0  /,P6/-1.2125D0/,
4572       &     P7/0.18402D0/,P8/-0.11582D-01/
4573        IF(X.LE.4D0) THEN
4574          FBARBB=P1+P2*X+P3*X**2+P4*X**3+P5*X**4+P6*X**5+P7*X**6+P8*X**7
4575           ELSE
4576          RBTH=1/X**2
4577          ALRB=LOG(RBTH)
4578          FBARBB=1D0/144*(311D0+24*PI2+282*ALRB+90*ALRB**2
4579       &        -4D0*RBTH*(40D0+ 6*PI2+ 15*ALRB+18*ALRB**2)
4580       &        +3D0*RBTH**2*(242.09D0-60*PI2-454.2D0*ALRB-180*ALRB**2))
4581        ENDIF
4582  *
4583        END
```

Match 13

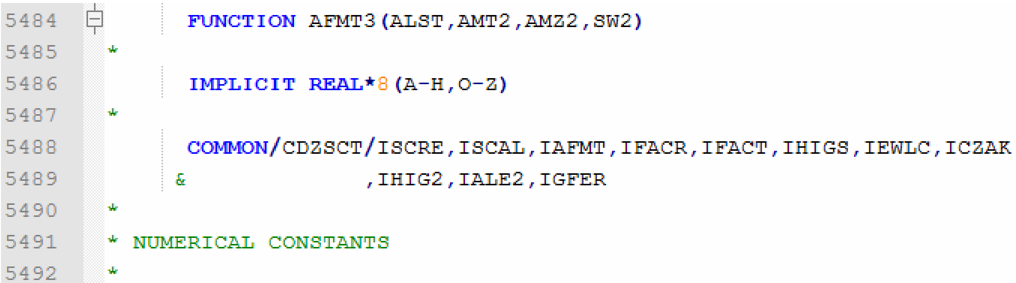
```
// Supplies real parts of J0,S3,S30 (w,t) in analytic presentation
// S3Wana(mt2,MW2,-s,...) supplies `w' indices
// S3Wana(MW2,mt2,-s,...) supplies `t' indices
// see dizet6_42.f line 4063-4093||~ line 4058-4093
void GSM::ZMath::S3Wana ( const Double_t& mt2, const Double_t& MW2, const Double_t& s,
                          Double_t& J0, Double_t& S3, Double_t& S30 )
{
  std::complex<Double_t> Cmt2 (mt2, -1e-10);
  std::complex<Double_t> CMW2 (MW2, -1e-10);
  std::complex<Double_t> Sqrt = std::sqrt( 1.0 + 4.0*CMW2/s );
  std::complex<Double_t> LogS = ZMath::Log( (Sqrt + 1.0)/(Sqrt - 1.0) );

  std::complex<Double_t> x0   = (Cmt2 - CMW2)/s;
  std::complex<Double_t> x1   = 0.5*(1.0 - Sqrt);
  std::complex<Double_t> x2   = 0.5*(1.0 + Sqrt);
  std::complex<Double_t> x3   = Cmt2/(Cmt2 - CMW2);

  std::complex<Double_t> y1   = x1/x0;
  std::complex<Double_t> y2   = (1.0 - x1)/(1.0 - x0);
  std::complex<Double_t> y3   = x2/x0;
  std::complex<Double_t> y4   = (1.0 - x2)/(1.0 - x0);
  std::complex<Double_t> y5   = x3/x0;
  std::complex<Double_t> y6   = (1.0 - x3)/(1.0 - x0);

  // results
  J0                = real( Sqrt*LogS );
  if (mt2 >= MW2) S30 = real( 1.0/s*LogS*LogS );
  else            S30 = real( 1.0/s*(Gfitter::GMath::Zeta2() - ZMath::Li2(1.0 - s/mt2)) );
  S3                = real( 1.0/s*( + ZMath::Li2( 1.0/(1.0-y1) ) - ZMath::Li2( 1.0/(1.0-y2) )
                                    + ZMath::Li2( 1.0/(1.0-y3) ) - ZMath::Li2( 1.0/(1.0-y4) )
                                    - ZMath::Li2( 1.0/(1.0-y5) ) + ZMath::Li2( 1.0/(1.0-y6) ) ) );

}
```



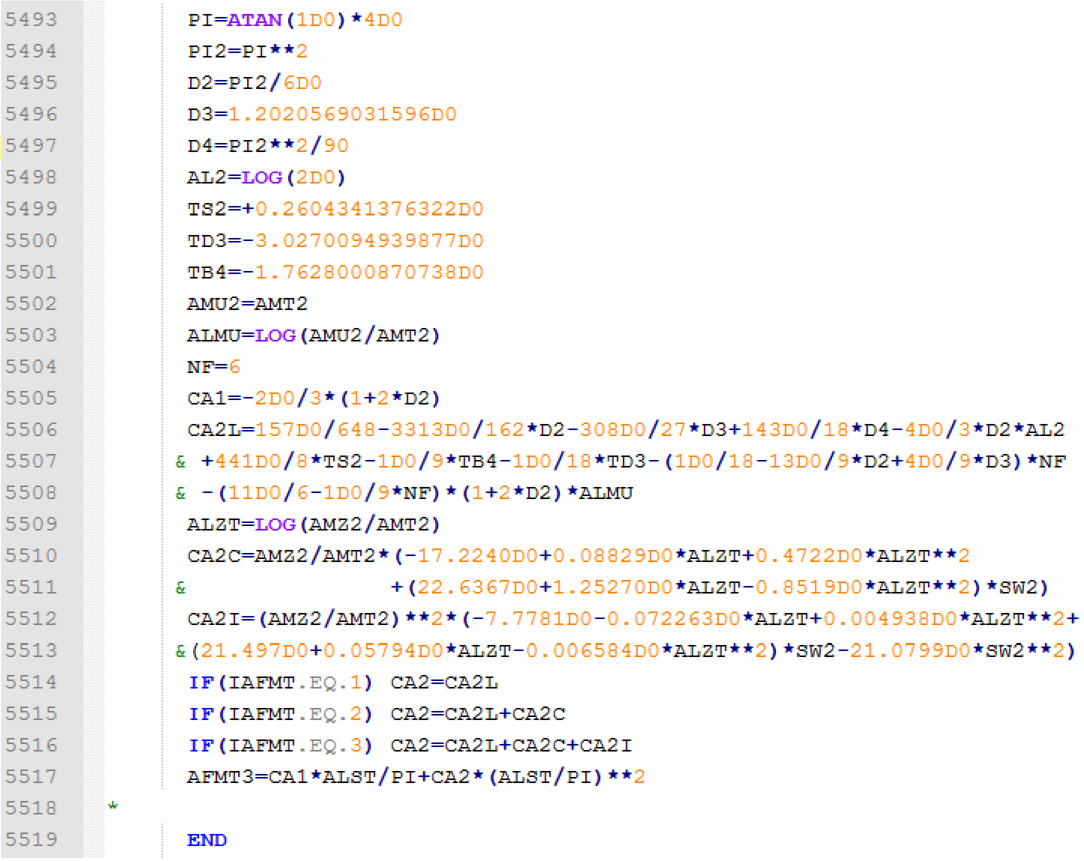
```
4058        LQR=LOG((SQR+1D0)/(SQR-1D0))
4059        J0 =SQR*LQR
4060        IF(MT2.GT.MW2) THEN
4061          S30=1D0/AMQ2*LQR**2
4062        ELSE
4063          S30=1D0/AMQ2*(D2-XSPENZ(1D0-AMQ2/AMT2))
4064        ENDIF
4065  *
4066        X1=(1D0-SQR)/2D0
4067        X2=(1D0+SQR)/2D0
4068        X0=(AMT2-AMW2)/AMQ2
4069        X3=AMT2/(AMT2-AMW2)
4070  *
4071        Y1=X1/X0
4072        Y2=(1D0-X1)/(1D0-X0)
4073        Y3=X2/X0
4074        Y4=(1D0-X2)/(1D0-X0)
4075        Y5=X3/X0
4076        Y6=(1D0-X3)/(1D0-X0)
4077  *
4078  c      A1=1D0/(1D0-Y1)
4079  c      A2=1D0/(1D0-Y2)
4080  c      A3=1D0/(1D0-Y3)
4081  c      A4=1D0/(1D0-Y4)
4082  c      A5=1D0/(1D0-Y5)
4083  c      A6=1D0/(1D0-Y6)
4084  *
4085        S3=1D0/AMQ2*(XSPENZ(1D0/(1D0-Y1))-XSPENZ(1D0/(1D0-Y2))
4086       &            +XSPENZ(1D0/(1D0-Y3))-XSPENZ(1D0/(1D0-Y4))
4087       &            -XSPENZ(1D0/(1D0-Y5))+XSPENZ(1D0/(1D0-Y6)))
4088  c      S3=1D0/AMQ2*(XSPENZ(A1)-XSPENZ(A2)
4089  c     &            +XSPENZ(A3)-XSPENZ(A4)
4090  c     &            -XSPENZ(A5)+XSPENZ(A6))
4091  *
4092        RETURN
4093        END
```

Match 14

```
// see dizet6_42.f line 5484-5519
Double_t GSM::ZMath::Afmt3( const Double_t& alst, const Double_t& mt2, const Double_t& MZ2, const Double_t& sW2 )
{
  Double_t TS2   =  0.2604341376322;
  Double_t TD3   = -3.0270094939877;
  Double_t TB4   = -1.7628000870738;
  Double_t LogZT =  TMath::Log(MZ2/mt2);
  Int_t nf       =  6;

  Double_t CA1   = -2/3.0*(1.0 + 2*Gfitter::GMath::Zeta2());
  Double_t CA2C  = ( MZ2/mt2*(-17.224 + 0.08829*LogZT + 0.4722*LogZT*LogZT
                     + ( 22.6367 + 1.2527*LogZT - 0.8519*LogZT*LogZT)*sW2) );
  Double_t CA2L  = ( 157/648.0 - 3313/162.0*Gfitter::GMath::Zeta2() - 308/27.0*Gfitter::GMath::Zeta3()
              + 143/18.0*Gfitter::GMath::Zeta4() - 4/3.0*Gfitter::GMath::Zeta2()*TMath::Log(2.0)
              + 441/8.0*TS2 - 1/9.0*TB4 - 1/18.0*TD3
              - (1/18.0 - 13/9.0*Gfitter::GMath::Zeta2() + 4/9.0*Gfitter::GMath::Zeta3())*nf );
  // - (11/6.0 - 1/9.0*nf)*(1.0 + 2*Gfitter::GMath::Zeta2())*TMath::Log(mt2/mt2)
  Double_t CA2I  = ( Gfitter::GMath::IPow(MZ2/mt2,2)*
              (-7.7781 - 0.072263*LogZT + 0.004938*LogZT*LogZT
              + (21.497 + 0.05794*LogZT - 0.006584*LogZT*LogZT)*sW2 - 21.0799*sW2*sW2) );

  Double_t CA2   = CA2L + CA2C + CA2I;

  return CA1*alst/TMath::Pi() + CA2*Gfitter::GMath::IPow( alst/TMath::Pi() ,2 );
}
```



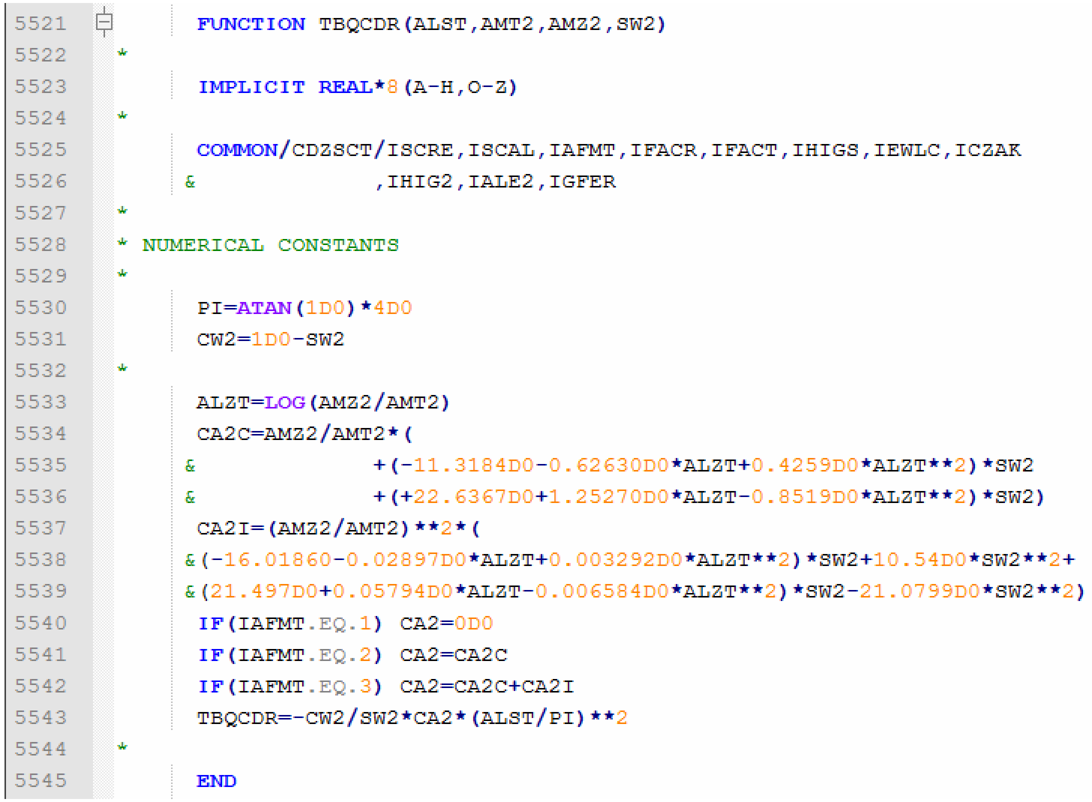
```
5484       FUNCTION AFMT3(ALST,AMT2,AMZ2,SW2)
5485 *
5486       IMPLICIT REAL*8(A-H,O-Z)
5487 *
5488       COMMON/CDZSCT/ISCRE,ISCAL,IAFMT,IFACR,IFACT,IHIGS,IEWLC,ICZAK
5489      &              ,IHIG2,IALE2,IGFER
5490 *
5491 * NUMERICAL CONSTANTS
5492 *
```



```
5493       PI=ATAN(1D0)*4D0
5494       PI2=PI**2
5495       D2=PI2/6D0
5496       D3=1.2020569031596D0
5497       D4=PI2**2/90
5498       AL2=LOG(2D0)
5499       TS2=+0.2604341376322D0
5500       TD3=-3.0270094939877D0
5501       TB4=-1.7628000870738D0
5502       AMU2=AMT2
5503       ALMU=LOG(AMU2/AMT2)
5504       NF=6
5505       CA1=-2D0/3*(1+2*D2)
5506       CA2L=157D0/648-3313D0/162*D2-308D0/27*D3+143D0/18*D4-4D0/3*D2*AL2
5507      & +441D0/8*TS2-1D0/9*TB4-1D0/18*TD3-(1D0/18-13D0/9*D2+4D0/9*D3)*NF
5508      & -(11D0/6-1D0/9*NF)*(1+2*D2)*ALMU
5509       ALZT=LOG(AMZ2/AMT2)
5510       CA2C=AMZ2/AMT2*(-17.2240D0+0.08829D0*ALZT+0.4722D0*ALZT**2
5511      &              +(22.6367D0+1.25270D0*ALZT-0.8519D0*ALZT**2)*SW2)
5512       CA2I=(AMZ2/AMT2)**2*(-7.7781D0-0.072263D0*ALZT+0.004938D0*ALZT**2+
5513      &(21.497D0+0.05794D0*ALZT-0.006584D0*ALZT**2)*SW2-21.0799D0*SW2**2)
5514       IF(IAFMT.EQ.1) CA2=CA2L
5515       IF(IAFMT.EQ.2) CA2=CA2L+CA2C
5516       IF(IAFMT.EQ.3) CA2=CA2L+CA2C+CA2I
5517       AFMT3=CA1*ALST/PI+CA2*(ALST/PI)**2
5518 *
5519       END
```

Match 15

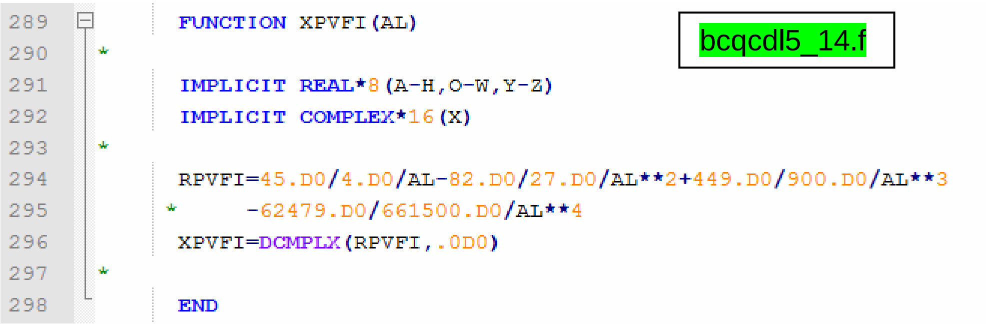
```
// see dizet6_42.f line 5521-5545
Double_t GSM::ZMath::TbQCD( const Double_t& alst, const Double_t& mt2, const Double_t& MZ2, const Double_t& sW2 )
{
   Double_t cW2   = 1.0 - sW2;
   Double_t logzt = TMath::Log( MZ2/mt2);

   Double_t Ca2c  = ( MZ2/mt2*( +(-11.3184 - 0.62630*logzt + 0.4259*logzt*logzt)*sW2
                                +(+22.6367 + 1.25270*logzt - 0.8519*logzt*logzt)*sW2) );
   Double_t Ca2i  = ( GMath::IPow( (MZ2/mt2), 2 )*( (-16.01860 - 0.02897*logzt + 0.003292*logzt*logzt)*sW2
                                             + 10.54*sW2*sW2
                                             + (21.497 + 0.05794*logzt - 0.006584*logzt*logzt)*sW2 - 21.0799*sW2*sW2) );

   return -cW2/sW2*(Ca2c + Ca2i)*GMath::IPow( (alst/TMath::Pi()), 2 );
}
```



```
5521        FUNCTION TBQCDR(ALST,AMT2,AMZ2,SW2)
5522  *
5523        IMPLICIT REAL*8(A-H,O-Z)
5524  *
5525        COMMON/CDZSCT/ISCRE,ISCAL,IAFMT,IFACR,IFACT,IHIGS,IEWLC,ICZAK
5526       &             ,IHIG2,IALE2,IGFER
5527  *
5528  * NUMERICAL CONSTANTS
5529  *
5530        PI=ATAN(1D0)*4D0
5531        CW2=1D0-SW2
5532  *
5533        ALZT=LOG(AMZ2/AMT2)
5534        CA2C=AMZ2/AMT2*(
5535       &             +(-11.3184D0-0.62630D0*ALZT+0.4259D0*ALZT**2)*SW2
5536       &             +(+22.6367D0+1.25270D0*ALZT-0.8519D0*ALZT**2)*SW2)
5537        CA2I=(AMZ2/AMT2)**2*(
5538       &(-16.01860-0.02897D0*ALZT+0.003292D0*ALZT**2)*SW2+10.54D0*SW2**2+
5539       &(21.497D0+0.05794D0*ALZT-0.006584D0*ALZT**2)*SW2-21.0799D0*SW2**2)
5540        IF(IAFMT.EQ.1) CA2=0D0
5541        IF(IAFMT.EQ.2) CA2=CA2C
5542        IF(IAFMT.EQ.3) CA2=CA2C+CA2I
5543        TBQCDR=-CW2/SW2*CA2*(ALST/PI)**2
5544  *
5545        END
```

Match 16

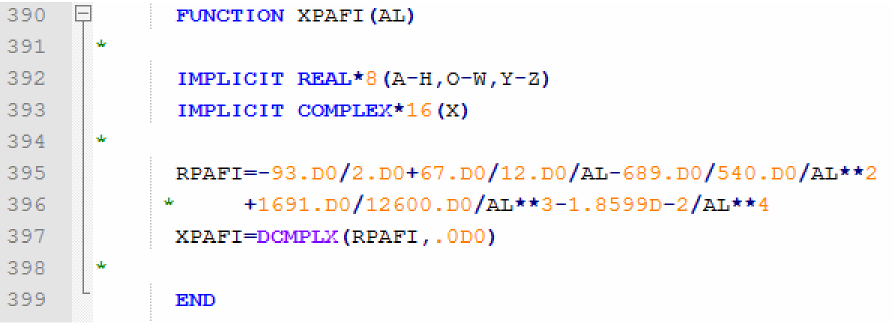
```
// ------------ some polarization operators and their auxiliary functions ----------------

// see bcqcd5_14.f line 289-298
complex<Double_t> GSM::ZMath::PolVfi( const Double_t& al )
{
   return ( 45.0/(4.0*al) - 82.0/(27.0*al*al) + 449.0/(900.0*al*al*al)
            - 62479.0/(661500.0*al*al*al*al) );
}
```



bcqcdl5_14.f

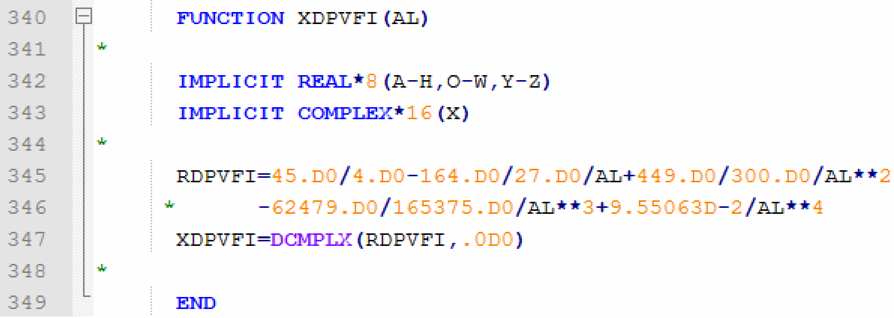
```
289       FUNCTION XPVFI(AL)
290 *
291       IMPLICIT REAL*8(A-H,O-W,Y-Z)
292       IMPLICIT COMPLEX*16(X)
293 *
294       RPVFI=45.D0/4.D0/AL-82.D0/27.D0/AL**2+449.D0/900.D0/AL**3
295      *      -62479.D0/661500.D0/AL**4
296       XPVFI=DCMPLX(RPVFI,.0D0)
297 *
298       END
```

```
// see bcqcd5_14.f line 390-399
complex<Double_t> GSM::ZMath::PolAfi( const Double_t& al )
{
   return ( -93.0/2.0+67.0/12.0/al-689.0/540.0/(al*al)
            +1691.0/12600.0/(al*al*al)-1.8599e-2/(al*al*al*al) );
}
```



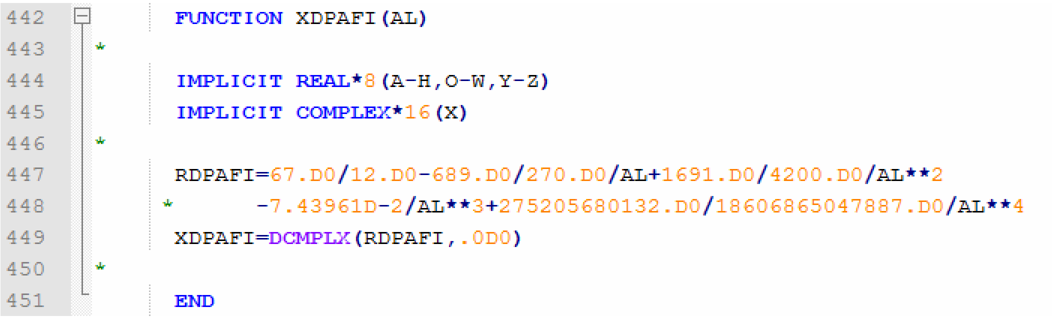
```
390       FUNCTION XPAFI(AL)
391 *
392       IMPLICIT REAL*8(A-H,O-W,Y-Z)
393       IMPLICIT COMPLEX*16(X)
394 *
395       RPAFI=-93.D0/2.D0+67.D0/12.D0/AL-689.D0/540.D0/AL**2
396      *      +1691.D0/12600.D0/AL**3-1.8599D-2/AL**4
397       XPAFI=DCMPLX(RPAFI,.0D0)
398 *
399       END
```

```
// see bcqcd5_14.f line 340-349
complex<Double_t> GSM::ZMath::DPolVfi( const Double_t& al )
{
   return ( 45.0/4.0 - 164.0/(27.0*al) + 449.0/(300.0*al*al)
            - 62479.0/(165375.0*al*al*al) + 9.55063e-2/(al*al*al*al) );
}
```



```
340       FUNCTION XDPVFI(AL)
341 *
342       IMPLICIT REAL*8(A-H,O-W,Y-Z)
343       IMPLICIT COMPLEX*16(X)
344 *
345       RDPVFI=45.D0/4.D0-164.D0/27.D0/AL+449.D0/300.D0/AL**2
346      *       -62479.D0/165375.D0/AL**3+9.55063D-2/AL**4
347       XDPVFI=DCMPLX(RDPVFI,.0D0)
348 *
349       END
```

```
// see bcqcd5_14.f line 442-451
complex<Double_t> GSM::ZMath::DPolAfi( const Double_t& al )
{
   return ( 67.0/12.0-689.0/270.0/al+1691.0/4200.0/(al*al)
            -7.43961e-2/(al*al*al)+275205680132.0/18606865047887.0/(al*al*al*al) );
}
```

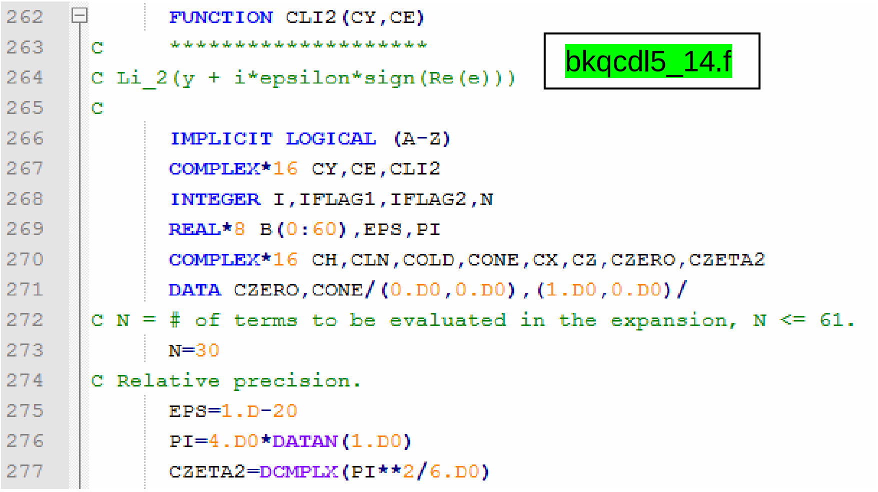


```
442       FUNCTION XDPAFI(AL)
443 *
444       IMPLICIT REAL*8(A-H,O-W,Y-Z)
445       IMPLICIT COMPLEX*16(X)
446 *
447       RDPAFI=67.D0/12.D0-689.D0/270.D0/AL+1691.D0/4200.D0/AL**2
448      *       -7.43961D-2/AL**3+275205680132.D0/18606865047887.D0/AL**4
449       XDPAFI=DCMPLX(RDPAFI,.0D0)
450 *
451       END
```

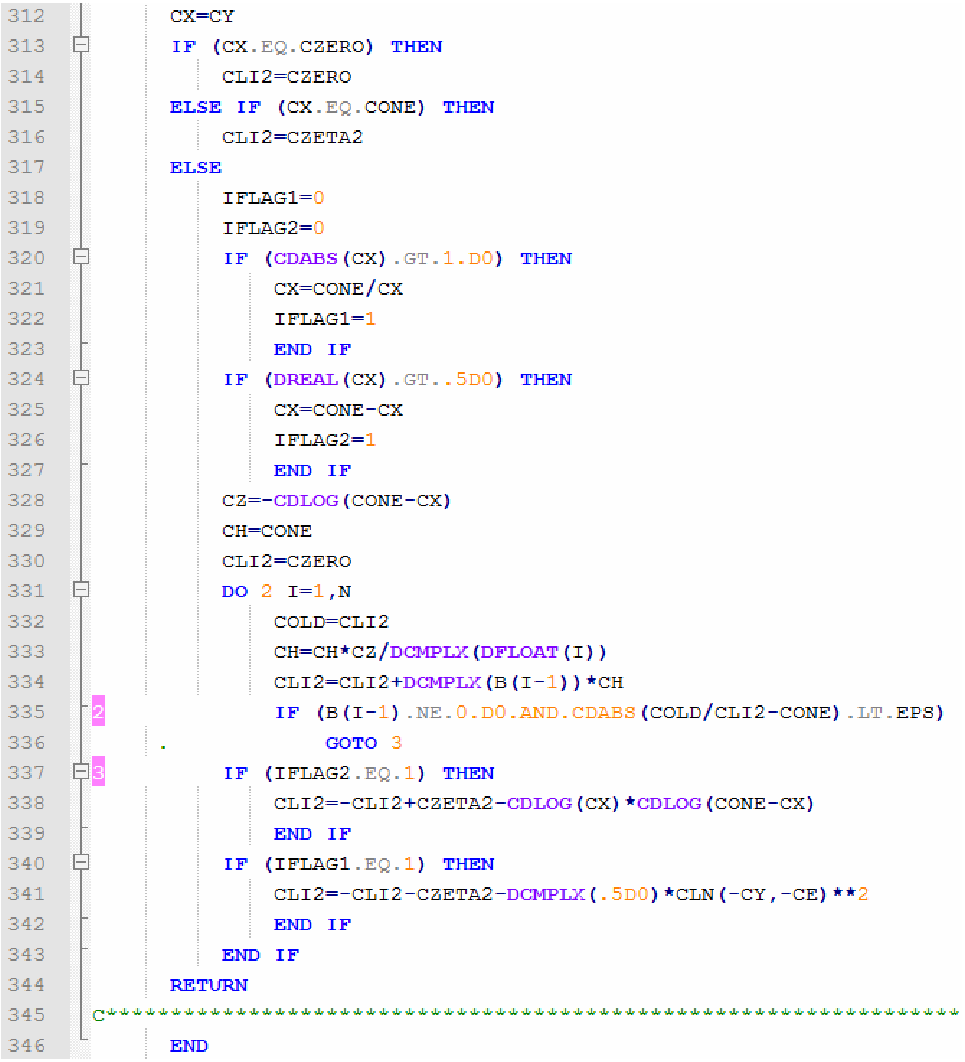
Match 17

```cpp

// Li_2(y + i*epsilon*sign(Re(e))), see for example
// "The Standard Model in the Making" by Bardin and Passarino
std::complex<Double_t> GSM::ZMath::Li2( const std::complex<Double_t>& y, std::complex<Double_t> e )
{
  std::complex<Double_t> li2  (0,0);
  std::complex<Double_t> cone (1.0,0);
  std::complex<Double_t> x = y;
  Double_t B[59];
  ZMath::GetCoefficients( B );

  const Int_t N = 30;

  Int_t flag1 = 0;
  Int_t flag2 = 0;

  if      (x == li2)  li2 = 0.0;
  else if (x == cone) li2 = GMath::Zeta2();
  else {
    if (abs(x) > 1.0) {
      x = cone/x;
      flag1 = 1;}
    if (real(x) > 0.5) {
      x = cone - x;
      flag2 = 1;}
    std::complex<Double_t> z = -ZMath::Log(cone-x);
    std::complex<Double_t> h = cone;
    std::complex<Double_t> Cold (0.0,0.0);
    for (Int_t i=1; i<=N; i++) {
      Cold = li2;
      h    = h*z/(1.0*i);
      li2  = li2 + B[i-1]*h;
      if (B[i-1] != 0 && abs(Cold/li2-cone) < 1e-20) break;
    }

    if (flag2 == 1) li2 = -li2 + GMath::Zeta2() - ZMath::Log(x)*ZMath::Log(cone-x);
    if (flag1 == 1) li2 = -li2 - GMath::Zeta2() - 0.5*pow(CLog(-y,-e),2);
  }

  return li2;
}

```

bkqcdl5_14.f



```fortran
      FUNCTION CLI2(CY,CE)
C     ********************
C Li_2(y + i*epsilon*sign(Re(e)))
C
      IMPLICIT LOGICAL (A-Z)
      COMPLEX*16 CY,CE,CLI2
      INTEGER I,IFLAG1,IFLAG2,N
      REAL*8 B(0:60),EPS,PI
      COMPLEX*16 CH,CLN,COLD,CONE,CX,CZ,CZERO,CZETA2
      DATA CZERO,CONE/(0.D0,0.D0),(1.D0,0.D0)/
C N = # of terms to be evaluated in the expansion, N <= 61.
      N=30
C Relative precision.
      EPS=1.D-20
      PI=4.D0*DATAN(1.D0)
      CZETA2=DCMPLX(PI**2/6.D0)
```

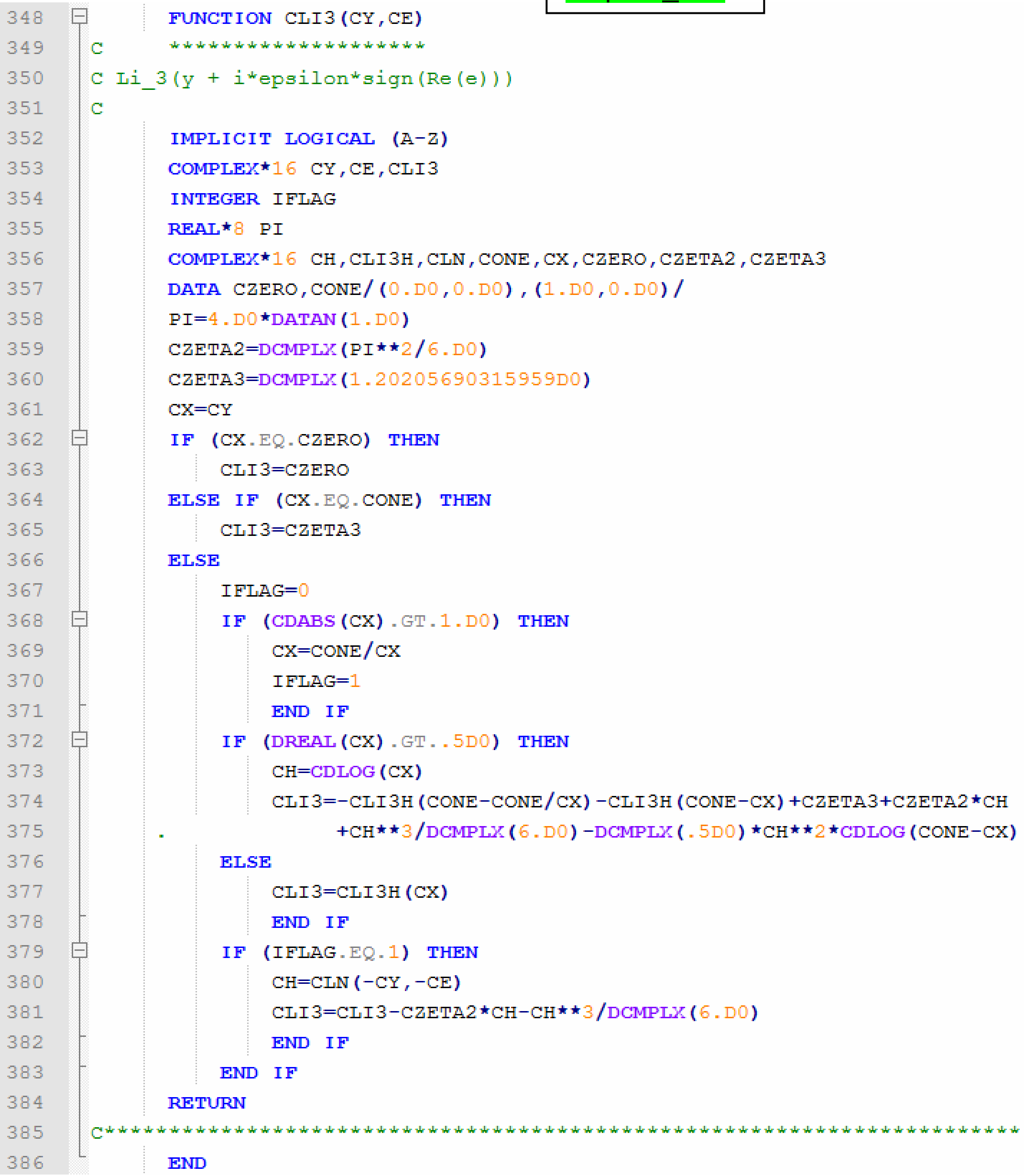


```fortran
      CX=CY
      IF (CX.EQ.CZERO) THEN
        CLI2=CZERO
      ELSE IF (CX.EQ.CONE) THEN
        CLI2=CZETA2
      ELSE
        IFLAG1=0
        IFLAG2=0
        IF (CDABS(CX).GT.1.D0) THEN
          CX=CONE/CX
          IFLAG1=1
        END IF
        IF (DREAL(CX).GT..5D0) THEN
          CX=CONE-CX
          IFLAG2=1
        END IF
        CZ=-CDLOG(CONE-CX)
        CH=CONE
        CLI2=CZERO
        DO 2 I=1,N
          COLD=CLI2
          CH=CH*CZ/DCMPLX(DFLOAT(I))
          CLI2=CLI2+DCMPLX(B(I-1))*CH
2         IF (B(I-1).NE.0.D0.AND.CDABS(COLD/CLI2-CONE).LT.EPS)
     .      GOTO 3
3       IF (IFLAG2.EQ.1) THEN
          CLI2=-CLI2+CZETA2-CDLOG(CX)*CDLOG(CONE-CX)
        END IF
        IF (IFLAG1.EQ.1) THEN
          CLI2=-CLI2-CZETA2-DCMPLX(.5D0)*CLN(-CY,-CE)**2
        END IF
      END IF
      RETURN
C**********************************************************************
      END
```

Match 18

```cpp
// Li_3(y + i*epsilon*sign(Re(e)))
std::complex<Double_t> GSM::ZMath::Li3( const std::complex<Double_t>& y, std::complex<Double_t> e )
{
  std::complex<Double_t> li3  (0,0);
  std::complex<Double_t> cone (1.0,0);
  std::complex<Double_t> x = y;
  std::complex<Double_t> h (0,0);

  Int_t flag = 0;

  if      (x == li3)  li3 = 0.0;
  else if (x == cone) li3 = GMath::Zeta3();
  else {
    if (abs(x) > 1.0) {
      x = cone/x;
      flag = 1;
    }

    if (real(x) > 0.5) {
      h   = ZMath::Log(x);
      li3 = ( - Li3H(cone-cone/x) - Li3H(cone-x) + GMath::Zeta3() + GMath::Zeta2()*h
              + h*h*h/6.0 - 0.5*h*h*ZMath::Log(cone-x) );
    }
    else li3 = Li3H(x);

    if (flag == 1) {
      h = CLog(-y,-e);
      li3 = li3 - GMath::Zeta2()*h - h*h*h/6.0;
    }
  }

  return li3;
}
```

bkqcdl5_14.f

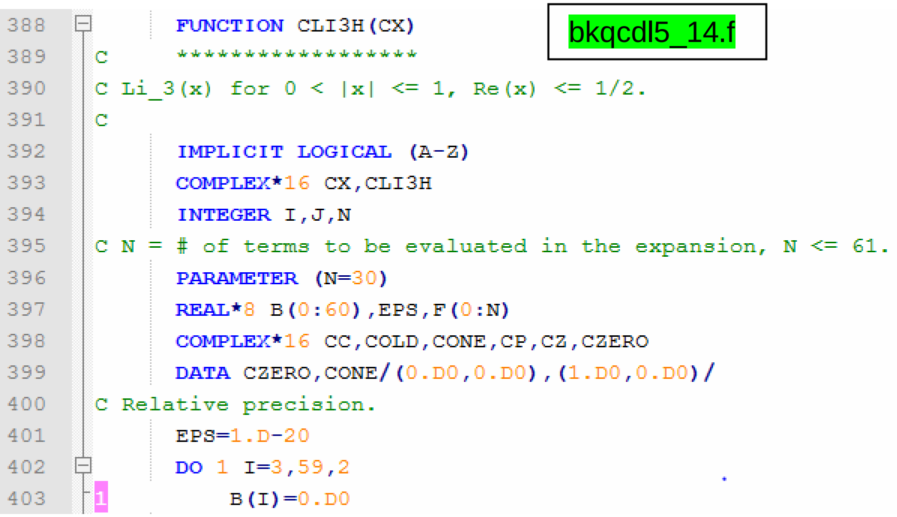


```fortran
      FUNCTION CLI3(CY,CE)
C     ********************
C Li_3(y + i*epsilon*sign(Re(e)))
C
      IMPLICIT LOGICAL (A-Z)
      COMPLEX*16 CY,CE,CLI3
      INTEGER IFLAG
      REAL*8 PI
      COMPLEX*16 CH,CLI3H,CLN,CONE,CX,CZERO,CZETA2,CZETA3
      DATA CZERO,CONE/(0.D0,0.D0),(1.D0,0.D0)/
      PI=4.D0*DATAN(1.D0)
      CZETA2=DCMPLX(PI**2/6.D0)
      CZETA3=DCMPLX(1.20205690315959D0)
      CX=CY
      IF (CX.EQ.CZERO) THEN
        CLI3=CZERO
      ELSE IF (CX.EQ.CONE) THEN
        CLI3=CZETA3
      ELSE
        IFLAG=0
        IF (CDABS(CX).GT.1.D0) THEN
          CX=CONE/CX
          IFLAG=1
          END IF
        IF (DREAL(CX).GT..5D0) THEN
          CH=CDLOG(CX)
          CLI3=-CLI3H(CONE-CONE/CX)-CLI3H(CONE-CX)+CZETA3+CZETA2*CH
     .          +CH**3/DCMPLX(6.D0)-DCMPLX(.5D0)*CH**2*CDLOG(CONE-CX)
        ELSE
          CLI3=CLI3H(CX)
          END IF
        IF (IFLAG.EQ.1) THEN
          CH=CLN(-CY,-CE)
          CLI3=CLI3-CZETA2*CH-CH**3/DCMPLX(6.D0)
          END IF
        END IF
      RETURN
C***********************************************************************
      END
```

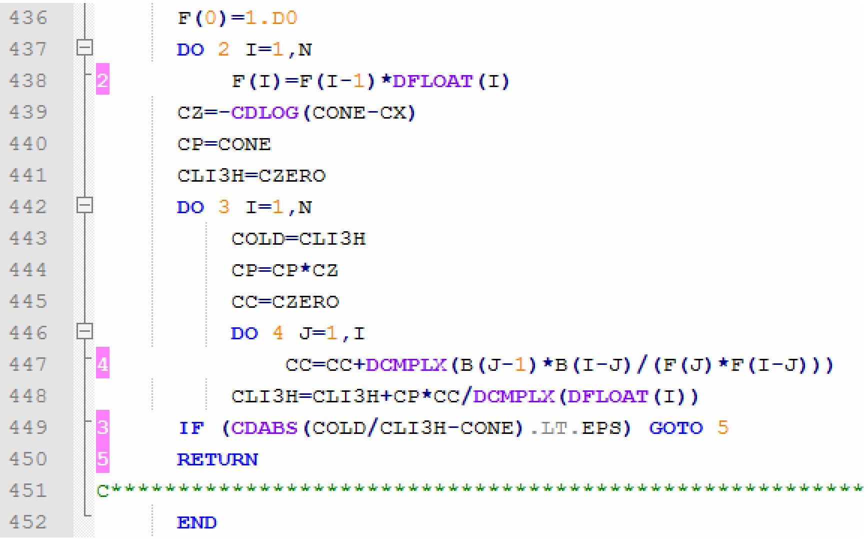
Match 19

```
// Li_3(x) for 0 < |x| <= 1, Re(x) <= 1/2
std::complex<Double_t> GSM::ZMath::Li3H( const std::complex<Double_t>& x )
{
  std::complex<Double_t> li3h (0,0);
  std::complex<Double_t> cone (1.0,0);
  std::complex<Double_t> Cold (0,0);
  std::complex<Double_t> c    (0,0);
  std::complex<Double_t> z = -ZMath::Log(cone-x);
  std::complex<Double_t> p = cone;
  Double_t B[59];
  ZMath::GetCoefficients( B );

  const Int_t N = 30;
  Double_t F[N+1];
  F[0] = 1.0;
  for (Int_t i=1; i <= N ; i++) F[i] = F[i-1]*1.0*i;

  for (Int_t i=1; i <= N ; i++) {
    Cold = li3h;
    p    = p*z;
    c    = 0.0;
    for (Int_t j = 1; j <= i; j++) c = c + B[j-1]*B[i-j]/(F[j]*F[i-j]);

    li3h = li3h + p*c/(1.0*i);
    if (abs(Cold/li3h - cone) < 1e-20) break;
  }

  return li3h;
}
```

bkqcdl5_14.f



```
      FUNCTION CLI3H(CX)
C     ******************
C Li_3(x) for 0 < |x| <= 1, Re(x) <= 1/2.
C
      IMPLICIT LOGICAL (A-Z)
      COMPLEX*16 CX,CLI3H
      INTEGER I,J,N
C N = # of terms to be evaluated in the expansion, N <= 61.
      PARAMETER (N=30)
      REAL*8 B(0:60),EPS,F(0:N)
      COMPLEX*16 CC,COLD,CONE,CP,CZ,CZERO
      DATA CZERO,CONE/(0.D0,0.D0),(1.D0,0.D0)/
C Relative precision.
      EPS=1.D-20
      DO 1 I=3,59,2
1        B(I)=0.D0
```

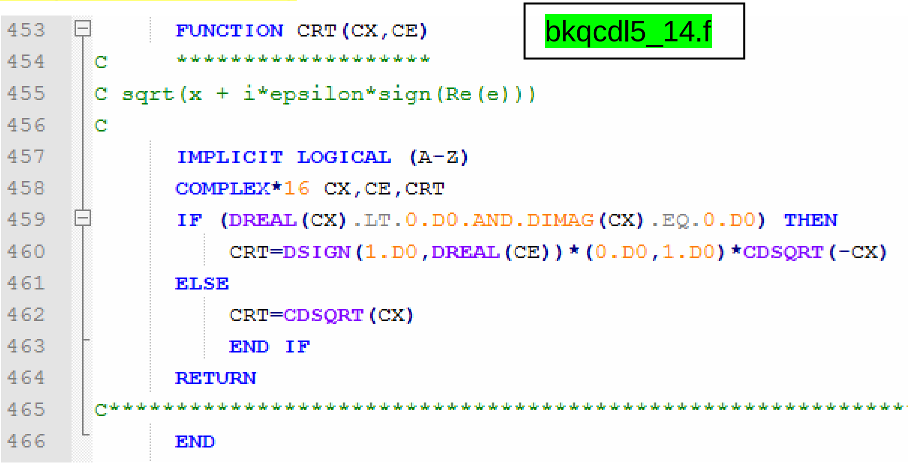


```
      F(0)=1.D0
      DO 2 I=1,N
2        F(I)=F(I-1)*DFLOAT(I)
      CZ=-CDLOG(CONE-CX)
      CP=CONE
      CLI3H=CZERO
      DO 3 I=1,N
         COLD=CLI3H
         CP=CP*CZ
         CC=CZERO
         DO 4 J=1,I
4           CC=CC+DCMPLX(B(J-1)*B(I-J)/(F(J)*F(I-J)))
         CLI3H=CLI3H+CP*CC/DCMPLX(DFLOAT(I))
3     IF (CDABS(COLD/CLI3H-CONE).LT.EPS) GOTO 5
5     RETURN
C*********************************************************
      END
```

Match 20

```
// sqrt(x + i*epsilon*sign(Re(e)))
std::complex<Double_t> GSM::ZMath::CSqrt( const std::complex<Double_t>& x, const std::complex<Double_t>& e )
{
  std::complex<Double_t> crt(0.0,0.0);
  std::complex<Double_t> I(0.0,1.0);

  Double_t sign;
  if (real(e) < 0) sign = -1.0;
  else             sign = 1.0;

  if (real(x) < 0 && imag(x) == 0) crt = sign*I*sqrt(-x);
  else                             crt = sqrt(x);

  return crt;
}
```

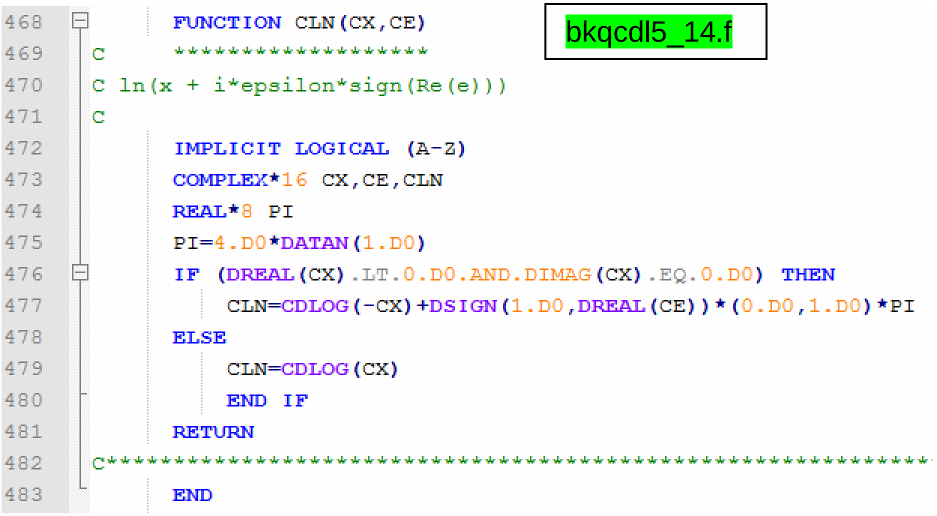
bkqcdl5_14.f



```
      FUNCTION CRT(CX,CE)
C     *******************
C sqrt(x + i*epsilon*sign(Re(e)))
C
      IMPLICIT LOGICAL (A-Z)
      COMPLEX*16 CX,CE,CRT
      IF (DREAL(CX).LT.0.D0.AND.DIMAG(CX).EQ.0.D0) THEN
         CRT=DSIGN(1.D0,DREAL(CE))*(0.D0,1.D0)*CDSQRT(-CX)
      ELSE
         CRT=CDSQRT(CX)
      END IF
      RETURN
C***************************************************************
      END
```

```
// ln(x + i*epsilon*sign(Re(e)))
std::complex<Double_t> GSM::ZMath::CLog( const std::complex<Double_t>& x, const std::complex<Double_t>& e )
{
  std::complex<Double_t> cln(0.0,0.0);
  std::complex<Double_t> I(0.0,1.0);

  Double_t sign;
  if (real(e) < 0) sign = -1.0;
  else             sign = 1.0;

  if (real(x) < 0 && imag(x) == 0) cln = ZMath::Log(-x) + sign*I*TMath::Pi();
  else                             cln = ZMath::Log(x);

  return cln;
}
```

bkqcdl5_14.f



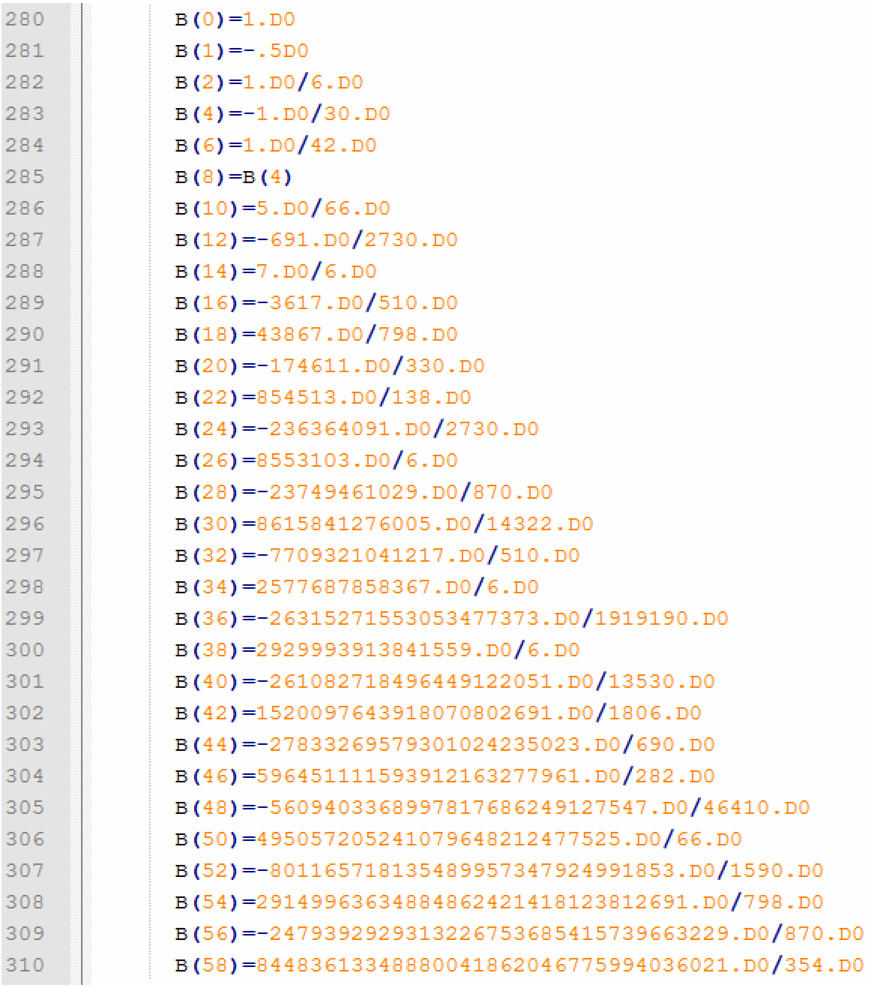
```
      FUNCTION CLN(CX,CE)
C     *******************
C ln(x + i*epsilon*sign(Re(e)))
C
      IMPLICIT LOGICAL (A-Z)
      COMPLEX*16 CX,CE,CLN
      REAL*8 PI
      PI=4.D0*DATAN(1.D0)
      IF (DREAL(CX).LT.0.D0.AND.DIMAG(CX).EQ.0.D0) THEN
         CLN=CDLOG(-CX)+DSIGN(1.D0,DREAL(CE))*(0.D0,1.D0)*PI
      ELSE
         CLN=CDLOG(CX)
      END IF
      RETURN
C***************************************************************
      END
```

Ende Match 20

```
// Coefficients for Euler polylogarithm Li_n , see for example
// "The Standard Model in the Making" by Bardin and Passarino
void GSM::ZMath::GetCoefficients( Double_t Coef[59] )
{
  for (Int_t i=0; i<59; i++) Coef[i] = 0;

  Coef[0]  = 1.0;
  Coef[1]  = -.50;
  Coef[2]  = 1.0/6.0;
  Coef[4]  = -1.0/30.0;
  Coef[6]  = 1.0/42.0;
  Coef[8]  = Coef[4];
  Coef[10] = 5.0/66.0;
  Coef[12] = -691.0/2730.0;
  Coef[14] = 7.0/6.0;
  Coef[16] = -3617.0/510.0;
  Coef[18] = 43867.0/798.0;
  Coef[20] = -174611.0/330.0;
  Coef[22] = 854513.0/138.0;
  Coef[24] = -236364091.0/2730.0;
  Coef[26] = 8553103.0/6.0;
  Coef[28] = -23749461029.0/870.0;
  Coef[30] = 8615841276005.0/14322.0;
  Coef[32] = -7709321041217.0/510.0;
  Coef[34] = 2577687858367.0/6.0;
  Coef[36] = -26315271553053477373.0/1919190.0;
  Coef[38] = 2929993913841559.0/6.0;
  Coef[40] = -261082718496449122051.0/13530.0;
  Coef[42] = 1520097643918070802691.0/1806.0;
  Coef[44] = -27833269579301024235023.0/690.0;
  Coef[46] = 596451111593912163277961.0/282.0;
  Coef[48] = -5609403368997817686249127547.0/46410.0;
  Coef[50] = 495057205241079648212477525.0/66.0;
  Coef[52] = -801165718135489957347924991853.0/1590.0;
  Coef[54] = 29149963634884862421418123812691.0/798.0;
  Coef[56] = -2479392929313226753685415739663229.0/870.0;
  Coef[58] = 84483613348880041862046775994036021.0/354.0;
}
```

Match 21

bkqcdl5_14.f, line 280-310 bzw. line 404-434



```
280        B(0)=1.D0
281        B(1)=-.5D0
282        B(2)=1.D0/6.D0
283        B(4)=-1.D0/30.D0
284        B(6)=1.D0/42.D0
285        B(8)=B(4)
286        B(10)=5.D0/66.D0
287        B(12)=-691.D0/2730.D0
288        B(14)=7.D0/6.D0
289        B(16)=-3617.D0/510.D0
290        B(18)=43867.D0/798.D0
291        B(20)=-174611.D0/330.D0
292        B(22)=854513.D0/138.D0
293        B(24)=-236364091.D0/2730.D0
294        B(26)=8553103.D0/6.D0
295        B(28)=-23749461029.D0/870.D0
296        B(30)=8615841276005.D0/14322.D0
297        B(32)=-7709321041217.D0/510.D0
298        B(34)=2577687858367.D0/6.D0
299        B(36)=-26315271553053477373.D0/1919190.D0
300        B(38)=2929993913841559.D0/6.D0
301        B(40)=-261082718496449122051.D0/13530.D0
302        B(42)=1520097643918070802691.D0/1806.D0
303        B(44)=-27833269579301024235023.D0/690.D0
304        B(46)=596451111593912163277961.D0/282.D0
305        B(48)=-5609403368997817686249127547.D0/46410.D0
306        B(50)=495057205241079648212477525.D0/66.D0
307        B(52)=-801165718135489957347924991853.D0/1590.D0
308        B(54)=29149963634884862421418123812691.D0/798.D0
309        B(56)=-2479392929313226753685415739663229.D0/870.D0
310        B(58)=84483613348880041862046775994036021.D0/354.D0
```