Scilab Textbook Companion for Nuclear Physics by D. C. Tayal[1]

Created by
Arjun Singh
Nuclear and Particle Physics
Physics
Shri Mata Vaishno Devi University
College Teacher
Mr. Pankaj Biswas
Cross-Checked by
Chaya Ravindra

July 31, 2019

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Nuclear Physics

Author: D. C. Tayal

Publisher: Himalaya Publishing House, Mumbai

Edition: 5

Year: 2011

ISBN: 978-93-5024-743-3

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

General Properties of Atomic Nucleus

Scilab code Exa 1.1 Distance of closest approach

```
// Scilab code Exa1.1 :  : Page 51 (2011)
clc; clear;
Z = 79;         // Atomic number of Gold
z = 1;          // Atomic number of Hydrogen
e = 1.60218e-019;          // Charge of an electron ,
    coulomb
K = 9e+09;               // Coulomb constant , newton
    metre square per coulomb square
E = 2*1.60218e-013;         // Energy of the proton ,
    joule
b = Z*z*e^2*K/E;             // Distance of closest
    approach , metre
printf("\nDistance of closest approach : %7.5e metre
    ", b);

// Result
// Distance of closest approach : 5.69575e-014 meter
```

Scilab code Exa 1.2 Nuclear Spin

```
// Scilab code Exa1.2 :  : Page 51 (2011)
clc; clear;
A = 14;          // Number of protons
Z = 7;          // Number of neutrons
N = A-Z;          // Number of electrons
i = modulo((N+A),2);    // Remainder
// Check for even and odd number of particles !!!!!
if  i == 0 then          // For even number of
   particles
    printf("\nParticles have integral spin");
    s = 1;    // Nuclear spin
end
 if i == 1 then          // For odd number of
    particles
    printf(" \nParticles have half integral spin ");
    s = 1/2;
end
if s == 1 then
    printf( "\nMeasured value agree with the
       assumption");
end
if s == 1/2 then
    printf( "\nMeasured value disagree with the
       assumption" );
end

// Result
// Particles have half integral spin
// Measured value disagree with the assumption
```

Scilab code Exa 1.3 Kinetic energy and Coulomb energy for an electron confined wit

```
// Scilab code Exa1.3 : : Page 52 (2011)
clc; clear;
p = 62;                     // Momentum of the electron , MeV/c
K = 9e+09;                  // Coulomb constant
E = 0.511;                  // Energy of the electron , MeV
e = 1.60218e-019;           // Charge of an electron , C
Z = 23;                     // Atomic number
R = 0.5*10^-14;             // Diameter of the nucleus , meter
T = sqrt(p^2+E^2)-E;        // Kinetic energy of the electron ,MeV
E_c = -Z*K*e^2/(R*1.60218e-013);                  // Coulomb energy , MeV
printf("\nKinetic energy of the electron : %5.2f MeV \nCoulomb energy per electron : %5.3f MeV",T,E_c);

// Result
// Kinetic energy of the electron : 61.49 MeV
// Coulomb energy per electron : -6.633 MeV
```

Scilab code Exa 1.4 Scattering of electron from target nucleus

```
// Scilab code Exa1.4 : : Page 52 (2011)
clc; clear;
K = 500*1.60218e-013;          // Kinetic energy of the electron ,joule
h = 6.6262e-034;               // Planck's constant , joule sec
C = 3e+08;                     // Velocity of light , metre per sec
```

```
p = K/C;                              // Momentum of the electron, joule sec per meter
lambda = h/p;                         // de Broglie wavelength, metre
A = 30*%pi/180;                       // Angle (in radian)
r = lambda/(A*10^-15);                // Radius of the target nucleus, femtometre
printf("\nRadius of the target nucleus : %4.2f fm", r);

// Result
// Radius of the target nucleus : 4.74 fm
```

Scilab code Exa 1.5 Positron emission from Cl33 decays

```
// Scilab code Exa1.5 : : Page 52 (2011)
clc; clear;
e = 1.60218e-019;             // Charge of an electron, C
A = 33;                    // Atomic mass of Chlorine, amu
K = 9e+09;                 // Coulomb constant, newton metre sqaure per coulomb square
E = 6.1*1.60218e-013;                 // Coulomb energy, joule
R_0 = 3/5*K/E*e^2*(A)^(2/3); // Distance of closest approach, metre
R = R_0*A^(1/3);                      // Radius of the nucleus, metre
printf("\nRadius of the nucleus : %4.2e metre", R);

// Result
// Radius of the nucleus : 4.6805e-015 metre
```

Scilab code Exa 1.6 Charge accelerated in mass spectrometer

```
// Scilab code Exa1.6:  : Page 53 (2011)
clc; clear;
V = 1000;              // Potential difference , volts
R = 18.2e-02;          // Radius of the orbit , metre
B = 1000e-04;          // Magnetic field , tesla
e = 1.60218e-019;           // Charge of an electron , C
n = 1;                       // Number of the ion
v = 2*V/(R*B);               // Speed of the ion , metre
    per sec
M = 2*n*e*V/v^2;               // Mass of the ion , Kg
printf("\nSpeed of the ion: %6.4e m/s \nMass of the
   ion : %4.2f u", v, M/1.67e-027);

// Result
// Speed of the ion: 1.0989e+05 m/s
// Mass of the ion : 15.89 u
```

Scilab code Exa 1.7 Ionized atoms in Bainbridge mass spectrograph

```
// Scilab code Exa1.7 : : Page 53 (2011)
clc; clear;
M = 20*1.66054e-027;          //
v = 10^5;              // Speed of the ion , metre per
   sec
B = 0.08;          // Magnetic field , tesla
e = 1.60218e-019;   // Charge of an electron , C
n = 1;              // Number of the ion
R_20 = M*v/(B*n*e)  // Radius of the neon-20, metre
R_22 = 22/20*R_20;  // Radius of the neon-22, metre
printf("\nRadius of the neon-20 : %5.3f metre \
   nRadius of the neon-22 : %5.3f metre", R_20, R_22
   );

```

```
// Result
// Radius of the neon-20 : 0.259 metre
// Radius of the neon-22 : 0.285 metre
```

Scilab code Exa 1.8 Calculating the mass of hydrogen

```
// Scilab code Exa1.8 : : Page 53 (2011)
clc; clear;
a = 17.78e-03;                 // First doublet mass
    difference, u
b = 72.97e-03;                 // Second doublet mass
    difference, u
c = 87.33e-03;                  // Third doublet mass
    difference, u
M_H = 1+1/32*(4*a+5*b-2*c);         // Mass of the
    hydrogen,amu
printf("\nMass of the hydrogen: %8.6f amu",M_H);

// Result
// Mass of the hydrogen: 1.008166 amu
```

Scilab code Exa 1.9 Silver ions in Smith mass spectrometer

```
// Scilab code Exa1.9 : : Page 54 (2011)
clc; clear;
e = 1.60218e-019;              // Charge of an
    electron,C
B = 0.65;                      // Magnetic field,
    tesla
d_S1_S2 = 27.94e-02;           // Distance between
    slit S1 and S2, metre
R_1 = d_S1_S2/2;               // Radius of orbit of
    ions entering slit S2,metre
```

```
d_S4_S5 = 26.248e-02;         // Distance between slit S4 and S5, metre
R_2 =  d_S4_S5/2;             //Radius of orbit of ions leaving slit S4,metre
M = 106.9*1.66054e-027;              // Mass of an ion (Ag+)Kg,
T_1 = B^2*e^2*R_1^2/(2*M*1.60218e-019);   // Kinetic energy of the ion entering slit S2,eV
T_2 = B^2*e^2*R_2^2/(2*M*1.60218e-019);   // Kinetic energy of the ion leaving slit S4,eV
printf("\nKinetic energy of the ion entering slit S2 : %d eV \nKinetic energy of the ion leaving slit S4 : %d eV ",T_1,T_2)

// Result
// Kinetic energy of the ion entering slit S2 : 3721 eV
// Kinetic energy of the ion leaving slit S4 : 3284 eV
```

Scilab code Exa 1.10 Calculation of energy released during nuclear fusion reaction

```
// Scilab code Ex1.10 :  : Page 55 (2011)
clc; clear;
M_Li = 7.0116004;    // Mass of lithium nucleus, u
M_Be = 7.016929;     // Mass of beryllium nucleus, u
m_e = 0.511;         // Mass of an electron, MeV
if (M_Li-M_Be)*931.48 < 2*m_e then
    printf("\nThe Li-7 is not a beta emitter");
else
    printf("\nThe Li-7 is a beta emitter");
end
if (M_Be-M_Li)*931.48 > 2*m_e then
    printf("\nThe Be-7 is a beta emitter");
else
```

```
    printf("\nThe Be−7 is not a beta emitter");
end

// Result
// The Li−7 is not a beta emitter
// The Be−7 is a beta emitter
```

Scilab code Exa 1.11 Binding energies calculation

```
// Scilab code Ex1.11 : : Page 55 (2011)
clc; clear;
M_n =  1.008665;                  // Mass of neutron , amu
M_p = 1.007825;                  // Mass of proton , amu
N_Ni = 36;                        // Number of neutron in Ni−64
Z_Ni = 28;                        // Atomic number of Ni−64
N_Cu = 35;                        //  Number of neutron in Cu−64
Z_Cu = 29;                        //  Atomic number of Cu−64
A = 64;                            // Mass number , amu
M_Ni = 63.927958;                  // Mass of Ni−64
M_Cu = 63.929759;                  // Mass of Cu−64
m_e = 0.511;                       // Mass of an electron , MeV
d_M_Ni = N_Ni*M_n+Z_Ni*M_p-M_Ni;         // Mass defect , amu
d_M_Cu = N_Cu*M_n+Z_Cu*M_p-M_Cu;         // Mass defect , amu
B_E_Ni = d_M_Ni*931.49;                 // Binding energy of Ni−64, MeV
B_E_Cu = d_M_Cu*931.49;                 // Binding energy of Cu−64, MeV
Av_B_E_Ni = B_E_Ni/A;                  //  Average
```

```
binding energy of Ni−64, MeV
Av_B_E_Cu = B_E_Cu/A;                          // Average
    binding energy of Cu−64, MeV
printf("\nBinding energy of Ni−64 : %7.3f MeV \
    nBinding energy of CU−64 : %7.3f MeV \nAverage
    binding energy of Ni−64 : %5.3f MeV \nAverage
    binding energy of Cu−64 : %5.3f MeV ", B_E_Ni,
    B_E_Cu, Av_B_E_Ni, Av_B_E_Cu);
if (M_Cu - M_Ni)*931.48 > 2*m_e then
    printf("\nNi−64 is not a beta emitter but Cu−64
        is a beta emitter");
end

// Result
// Binding energy of Ni−64 : 561.765 MeV
// Binding energy of CU−64 : 559.305 MeV
// Average binding energy of Ni−64 : 8.778 MeV
// Average binding energy of Cu−64 : 8.739 MeV
// Ni−64 is not a beta emitter but Cu−64 is a beta
    emitter
```

Scilab code Exa 1.12 Calculation of energy released during nuclear fusion reaction

```
// Scilab code Exa1.12 : : Page 55 (2011)
clc; clear;
M_n =  1.008665*931.49;                 // Mass of
    neutron, MeV
M_p = 1.007825*931.49;                 // Mass of proton
    , MeV
M_He = 2*M_p+2*M_n-28;          // Mass of He−4
    nucleus, MeV
M_H = M_p+M_n-2.2;                // Mass of H−2 nucleus
    , MeV
d_E = 2*M_H-M_He;                    // Energy released
    during fusion reaction, MeV
```

```
printf("\nEnergy released during fusion reaction : %4.1f MeV ",d_E);

// Result
// Energy released during fusion reaction : 23.6 MeV
```

Scilab code Exa 1.13 `To find the stable Isobar`

```
// Scilab code Ex1.13 : : P.No.55 (2011)
// We have to determine for mass numbers 80 and 97.
clc; clear;
A = [80, 97];    // Matrix of Mass numbers
Element = ["Br","Mo"];   // Matrix of elements
M_n = 939.6;     // Mass of neutron, MeV
M_H = 938.8;     // Mass of proton, MeV
a_v = 14.0;      // Volume energy, MeV
a_s = 13.0;      // Surface energy, MeV
a_c = 0.583;     // Coulomb energy, MeV
a_a = 19.3;      // Asymmetry energy, MeV
a_p = 33.5;      // Pairing energy, MeV
for i = 1:1:2
Z = poly(0,'Z');         // Declare the polynomial variable
M_AZ = M_n*(A(i)-Z)+M_H*Z-a_v*A(i)+a_s*A(i)^(2/3)+a_c*Z*(Z-1)*A(i)^(-1/3)+a_a*(A(i)-2*Z)^2/A(i)+a_p*A(i)^(-3/4); // Mass of the nuclide, MeV/c^2
Z = roots(derivat(M_AZ));
printf("\nFor A = %d, the most stable isobar is %s(%d,%d)", A(i), Element(i), Z, A(i));
end

// Result
// For A = 80, the most stable isobar is Br(35,80)
// For A = 97, the most stable isobar is Mo(42,97)
```

Scilab code Exa 1.14 To calculate the pairing energy term

```
// Scilab code Exa1.14 : : P.no. 56(2011)
clc; clear;
A = 50;                    // Mass number
M_Sc = 49.951730;          // Mass of scandium, atomic mass unit
M_Ti = 49.944786;           // Mass of titanium, atomic mass unit
M_V = 49.947167;           // Mass of vanadium, atomic mass unit
M_Cr = 49.946055;           // Mass of chromium, atomic mass unit
M_Mn = 49.954215;           // Mass of manganese, atomic mass unit
a_p = (M_Mn-M_Cr+M_V-M_Ti)/(8*A^(-3/4))*931.5;    // Pairing energy temr, mega electron volts
printf("\nPairing energy term : %5.2f MeV", a_p);

// Result
// Pairing energy term : 23.08 MeV
```

Scilab code Exa 1.17 Relative error in the electric potential at the first Bohr ra

```
// Scilab code Ex1.17 : : Page 57 (2011)
clc; clear;
b = 1;  // For simplicity assume minor axis length to be unity, unit
a = 10/100+b;   // Major axis length, unit
A = 125;    // Mass number of medium nucleus
r = 0.53e-010;  // Bohr's radius, m
eps = (a-b)/(0.5*a+b);  // Deformation parameter
```

```
R = 1.2e-015*A^(1/3);     // Radius of the nucleus, m
Q = 1.22/15*R^2    // Electric Quadrupole moment,
    metre square
V_rel_err = Q/r^2;        // Relative error in the
    potential
printf("\nThe relative error in the electric
    potential at the first Bohr radius : %e",
    V_rel_err);

// Result
// The relative error in the electric potential at
    the first Bohr radius : 1.042364e-09
```

Scilab code Exa 1.21 Spherical symmetry of Gadolinium nucleus

```
// Scilab code Exa1.21 :  : Page-58(2011)
clc; clear;
Q = 130;     // Quadrupole moment, square femto metre
A = 155;     // Mass number of gadolinium
R_0 = 1.4*A^(1/3)    // Distance of closest approach,
     fm
Z = 64;      // Atomic number
delR0 = 5*Q/(6*Z*R_0^2)*100;     // Change in the
    value of R_0, percent
printf("\nChange in the value of fractional change
    in R_0 is only %4.2f percent \nThus, we can
    assumed that Gadolinium nucleus is spherical.",
    delR0);

// Result
// Change in the value of fractional change in R_0
    is only 2.99 percent
// Thus, we can assumed that Gadolinium nucleus is
    spherical.
```

Chapter 2

Radioactivity and Isotopes

Scilab code Exa 2.1 `Weight of one Curie and one Rutherford of RaB`

```
// Scilab code Exa2.1: : Page-88 (2011)
clc; clear;
T = 26.8*60;     // Half life of the substance, s
C = 3.7e+010;    // One curie, disintegration per sec
N = 6.022137e+026; // Avogadro number, per kmol
m = 214; // Molecular weight of RaB, kg/kmol
R = 1e+006; // One Rutherford, disintegration per sec.
W_C = C*T*m/(N*0.693); // Weight of one Curie of RaB, Kg
W_R = R*T*m/(N*0.693); // Weight of one Rutherford of RaB, Kg
printf("\nWeight of one Curie of RaB : %5.3e Kg \nWeight of one Rutherford of RaB : %5.3e Kg", W_C, W_R);

// Result
// Weight of one Curie of RaB : 3.051e-011 Kg
// Weight of one Rutherford of RaB : 8.245e-016 Kg
```

Scilab code Exa 2.2 Induced radioactivity of sodium by neutron bombardment

```
// Scilab code Exa2.2 : : Page 88 (2011)
clc; clear;
T_h = 14.8; // Half life of Na-24, hours
Q = 1e+008; // Production rate of Na-24, per sec
L = 0.693/T_h; // Decay constant , per sec
t = 2; // Time after the bombardment , hours
A = Q/3.7e+010*1000; // The maximum activity of Na
    -24, mCi
T = -1*log(0.1)/L; // The time needed to produced 90
    % of the maximum activity , hour
N = 0.9*Q*3600/L*%e^(-L*t); // Number of atoms of Na
    -24 left two hours after bombardment was stopped
printf("\nThe maximum activity of Na-24 = %3.1f mCi\
    nThe time needed to produced 90 percent of the
    maximum activity = %4.1f hrs \nNumber of atoms of
     Na-24 left two hours after bombardment was
    stopped = %4.2e ", A, T, N);

// Result
// The maximum activity of Na-24 = 2.7 mCi
// The time needed to produced 90 percent of the
    maximum activity = 49.2 hrs
// Number of atoms of Na-24 left two hours after
    bombardment was stopped = 6.30e+012
```

Scilab code Exa 2.3 Activity of K40 in man of weight 100 Kg

```
// Scilab code Exa2.3: : Page 89 (2011)
clc; clear;
```

```
T = 1.31e+09*365*24*60*60;      // Half life of the
    substance,sec
N = 6.022137e+026; // Avogadro number.
m = 0.35*0.012*10^-2; // Mass of K-40, Kg.
A = m*N*0.693/(T*40); // Activity of K-40,
    disintegrations/sec.
printf("\nThe activity of K-40 = %5.3e
    disintegrations/sec = %5.3f micro-curie", A, A
    /3.7e+004);

// Result
// The activity of K-40 = 1.061e+004 disintegrations
    /sec = 0.287 micro-curie
```

Scilab code Exa 2.4 `Age of an ancient wooden boat`

```
// Scilab code Exa2.4 : : Page 89 (2011)
clc; clear;
T = 5568;      // Half life of the C-14,years
lambda = 0.693/T; // Disintegration constant, years
    ^-1.
N_0 = 15.6/lambda; // Activity of fresh carbon, dpm
    .gm
N = 3.9/lambda; // Activity of an ancient wooden
    boat,dpm.gm.
t = 1/(lambda)*log(N_0/N); // Age of the boat, years
printf("\nThe age of the boat : %5.3e years", t);

// Result
// The age of the boat : 1.114e+004 years
```

Scilab code Exa 2.5 `Activity of the U234`

```
// Scilab code Exa2.5 : : Page 90 (2011)
clc; clear;
m_0 = 3e-06;// Initial mass of the U-234, Kg
A = 6.022137e+026; //Avagadro's number, atoms
N_0 = m_0*A/234; // Initial number of atoms
T = 2.50e+05; // Half life, years
lambda = 0.693/T; // Disintegration constant
t = 150000; // Disintegration time, years
m = m_0*%e^(-lambda*t); // Mass after time t,Kg
activity = m*lambda/(365*24*60*60)*A/234; // Activity of U-234 after time t,dps
printf("\nThe activity of U-234 after %6d yrs = %5.3e disintegrations/sec", t, activity);

// Result
// The activity of U-234 after 150000 yrs = 4.478e+005 disintegrations/sec
```

Scilab code Exa 2.6 Number of alpha decays in Th232

```
// Scilab code Exa2.6 : : Page 90 (2011)
clc; clear;
A = 6.022137e+023; //Avagadro's number, atoms
N_0 = A/232; // Initial number of atoms
t = 3.150e+07; // Decay time, sec
lambda = 1.58e-018; // Disintegration constant,sec^-1
N = lambda*t*N_0; // Number of alpha decays in Th-232
printf("\nThe number of alpha decays in Th-232 = %5.2e ", N);

// Result
// The number of alpha decays in Th-232 = 1.29e+011
```

Scilab code Exa 2.7 Maximum possible age of the earth crust

```
// Scilab code Exa2.7 : : Page 90 (2011)
clc; clear;
T_238 = 4.5e+09;// Half life of U-238, years
T_235 = 7.13e+08; // Half life of U-238, years
lambda_238 = 0.693/T_238; // Disintegration constant
    of U-238, years^-1
lambda_235 = 0.693/T_235; // Disintegration constant
    of U-235, years^-1
N = 137.8; // Abundances of U-238/U-235
t = log(N)/(lambda_235 - lambda_238);// Age of the
   earth's crust, years
printf("\nThe maximum possible age of the earth
   crust = %5.3e years", t);

// Result
// The maximum possible age of the earth crust =
   6.022e+009 years
```

Scilab code Exa 2.8 Number of radon half lives

```
// Scilab code Exa2.8 : : Page 91 (2011)
clc; clear;
N = 10; // Number of atoms left undecayed in Rn-222
n = log(10)/log(2); // Number of half lives in Ra
   -222
printf("\nThe number of half lives in radon-222 = %5
   .3f ", n);

// Result
// The number of half lives in radon-222 = 3.322
```

Scilab code Exa 2.9 Weight and initial acivity of Po210

```
// Scilab code Exa2.9 : : Page 91 (2011)
clc; clear;
M_Po = 209.9829; // Mass of Polonium, g
M_Pb = 205.9745; // Mass of lead, g
A = 6.22137e+023; // Avogadro's number
M_He = 4.0026; // Mass of alpha particle, g
C = 3e+08; // Velocity of light, m/s
T = 138*24*3600; // Half life, sec
P = 250; // Power produced, joule/sec
Q = [M_Po-M_Pb-M_He]*931.25; // disintegration energy, MeV
lambda = 0.693/T; // Disintegration constant, per year
N = P/(lambda*Q*1.60218e-013); // Number of atoms, atom
N_0 = N*%e^(1.833); // Number of atoms present initially, atom
W = N_0/A*210; // Weight of Po-210 after one year, g
A_0 = N_0*lambda/(3.7e+010); // Initial activity, curie
printf("\nThe weight of Po-210 after one year = %5.2f g \nThe initial activity of the material = %4.2e curies", W, A_0);

// Result
// The weight of Po-210 after one year = 10.49 g
// The initial activity of the material = 4.88e+004 curies
```

Scilab code Exa 2.10 Radioactive disintegration of Bi

```
// Scilab code Exa2.10 : : Page 91 (2011)
clc; clear;
lambda_t = 0.693/(60.5*60);// Total decay constant, per sec
lambda_a = 0.34*lambda_t;// Decay constant for alpha_decay, per sec
lambda_b = 0.66*lambda_t;// Decay constant for beta_decay, per sec
printf("\nThe decay constant for total emission = %4.2e /sec", lambda_t);
printf("\nThe decay constant for beta_decay lambda_b = %4.2e /sec", lambda_b);
printf("\nThe decay constant for alpha_decay lambda_a = %4.2e /sec", lambda_a);

// Result
// The decay constant for total emission = 1.91e-004 /sec
// The decay constant for beta_decay lambda_b = 1.26e-004 /sec
// The decay constant for alpha_decay lambda_a = 6.49e-005 /sec
```

Scilab code Exa 2.13 Half life of Pu239

```
// Scilab code Exa2.13 :  : Page 93 (2011)
clc; clear;
M_A = 4;    // Mass of alpha particle, amu
M_U = 235;    //Mass of U-235, amu
M_P = 239;    // Mass of P-239, amu
Amount = 120.1;    // quantity of P-239, g
E_A = 5.144;    // Energy of emitting alpha particles, Mev
E_R = (2*M_A)/(2*M_U)*E_A;    // The recoil energy of U-235, Mev
```

```
E = E_R + E_A;    // The energy released per disintegration , Mev
P = 0.231;    // Evaporation rate , watt
D = P/(E*1.60218e-013);    // Disintegration rate , per sec
A = 6.022137e+023;    // Avagadro 's number , atoms
N = Amount/M_P*A;    // Number of nuclei in 120.1g of P−239
T = 0.693/(D*3.15e+07)*N;    // Half life of Pu_239 , years
printf("\nThe half life of Pu−239 = %3.2e years", T);

// Result
// The half life of Pu−239 = 2.42e+004 years
```

Scilab code Exa 2.14 Disintegration rate of Au199

```
// Scilab code Exa2.14 : : Page 93 (2011)
clc; clear;
T_h_1 = 2.7*24*3600; // Half life of Au−198, sec
T_h_2 = 3.15*24*3600; // Half life of Au−199, sec
S_1 = 99e-028; // Crossection for first reaction , Sq.m
S_2 = 2.6e-024; // Crossection for second reaction , Sq.m
I = 1e+018; // Intensity of radiation , per Sq.m per sec
L_1 = I*S_1;    // Decay constant of Au−197, per sec
L_2 = 0.693/T_h_1+I*S_2; // Decay constant of Au−198, per sec
L_3 = 0.693/T_h_2; // Decay constant of Au−199, per sec
N_0 = 6.022137e+023; // Avogadro number
N_1 = N_0/197; // Initial number of atoms of Au−197
```

```
t = 30*3600; // Given time, sec
p = [exp(-L_1*t)]/[(L_2-L_1)*(L_3-L_1)];
q = [exp(-L_2*t)]/[(L_1-L_2)*(L_3-L_2)];
r = [exp(-L_3*t)]/[(L_1-L_3)*(L_2-L_3)];
N3 = N_1*L_1*L_2*[p+q+r];
N_199 = N3;
L = L_3*N_199; // Disintegration rate of Au-199, per sec
printf("\nThe disintegration rate of Au-199 = %3.1e", L);

// Result
// The disintegration rate of Au-199 = 1.9e+012 (Wrong answer in the textbook)
```

Scilab code Exa 2.15 Activity of Na24

```
// Scilab code Exa2.15 : : Page 94 (2011)
clc; clear;
Y = 110e-03;// Yield of Na-24, mCi/hr
T = 14.8;// Half life of Na-24, hours
t = 8;// Time after which activity to be compute, hours
lambda = 0.693/T;// Disintegration constant, hours^-1
A = 1.44*Y*T;// Maximum activity of Na-24, Ci
A_C = A*[1-%e^(-lambda*t)];// Activity after a continuous bombardment, Ci
Activity = A_C*(%e^(-lambda*t));// Activity after 8 hours, Ci
printf("\nThe maximum activity of Na-24 = %5.3f Ci\nThe activity after a continuous bombardment = %6.4f Ci\nThe activity after 8hours = %7.5f Ci",A, A_C, Activity);

```

```
// Result
// The maximum activity of Na−24 = 2.344 Ci
// The activity after a continuous bombardment =
   0.7324 Ci
// The activity after 8hours = 0.50360 Ci
```

Scilab code Exa 2.16 Radiation dose absorbed in 24 hr by the tissue in REP

```
// Scilab code Exa2.16 : : Page 94 (2011)
clc; clear;
A_0 = 3.7e+07;    // Initial activity,
   disintegrations per sec
T = 12.6;    // Half life of I−130, hours
t = 24*3600;    // time for dose absorbed
   calculation, sec
E = 0.29*1.6e-06;    // Average energy of beta rays,
    ergs
m = 2;    // Mass of iodine thyroid tissue, gm
lambda = 0.693/(T*3600);    // Disintegration
   constant, sec^−1
N_0 = A_0/lambda;    // Initial number of atoms
N = N_0*[1-%e^(-lambda*t)];    // Number of average
   atoms disintegrated
E_A = N*E;    // Energy of beta rays emitted, ergs
E_G = E_A/(2*97.00035);    // Energy of beta rays
   emitted per gram of tissue, REP
printf("\nThe energy of beta rays emitted per gram
   of tissue = %6.1f REP", E_G);

// Result
// The energy of beta rays emitted per gram of
   tissue = 4245.0 REP
```

Scilab code Exa 2.18 Activity and the maximum amount of Au198 produced in the foil

```
// Scilab code Exa2.18 : : Page 95 (2011)
clc; clear;
N_0 = 6.022137e+023; //  Avagadro number
d = 0.02; // Thickness of the foil , cm
R = 19.3; // Density of Au,g/cc
N_1 = d*R/197*N_0; // Initial number of Au-197
    nuclei per unit area of foil ,cm^-2
T_H = 2.7*24*3600; // Half life of Au-198,sec
L = log(2)/T_H; // Decay constant for Au-198,sec^-1
I = 10^12; // Intensity of neutron beam,neutrons/cm
    ^2/sec
S = 97.8e-024; // Cross section for reaction ,cm^-2
t = 5*60; // Reaction time,s
A = S*I*N_1*(1-%e^(-L*t)); // Activity of Au-198,cm
    ^-2sec^-1
N_2 = S*I*N_1/L; // The maximum amount of Au-198
    produced ,cm^-2
printf("\nThe activity of Au-198 = %5.3e per Sq.cm
    per sec\nThe maximum amount of Au-198 produced =
    %4.2e per Sq.cm", A, N_2);

// Result
// The activity of Au-198 = 1.028e+008 per Sq.cm per
    sec
// The maximum amount of Au-198 produced = 3.88e+016
    per Sq.cm
```

Scilab code Exa 2.19 Pu238 as power source in space flights

```
// Scilab code Exa2.19 : : Page 95 (2011)
clc; clear;
N_0 = 6.022137e+023; //  Avagadro number
T_P = 90*365*24*3600; // Half life of Pu-238,s
```

```
L_P = 0.693/T_P ; // Decay constant of Pu−238,s^−1
E = 5.5; // Energy of alpha particle , MeV
P =E*L_P*N_0; // Power released by the gm molecule
    of Pu−238,MeV/s
t = log(8)/(L_P*365*24*3600); // Time in which power
     reduces to 1/8 time of its initial value
printf("\nThe power released by the gm molecule of
    Pu−238 = %4.2e MeV/s \nThe time in which power
    reduces to 1/8 time of its initial value = %d yrs
    ",P,t)

// Result
// The power released by the gm molecule of Pu−238 =
     8.09e+014 MeV/s
// The time in which power reduces to 1/8 time of
    its initial value = 270 yrs
```

Scilab code Exa 2.20 Series radioactive decay of parent isotope

```
// Scilab code Exa2.20 : : Page 96 (2011)
clc; clear;
N_1 = 10^20; // Number of nuclei of parent isotopes
T_P = 10^4; // Half life of parent nucleus ,years
T_D = 20; // Half life of daughter nucleus ,years
T = 10^4; // Given time ,years
L_P = 0.693/T_P ; // Decay constant of parent
    nucleus ,years^−1
L_D = 0.693/T_D ; // Decay constant of daughter
    nucleus ,years^−1
t_0 = log(0.03)/(L_P-L_D); // Required time for
    decay of daughter nucleus ,years
N = L_P/L_D*(%e^(-L_P*T)-%e^(-L_D*T))*N_1; // Number
     of nuclei of daughter isotope
printf("\nThe required time for decay of daughter
    nucleus = %d yr \nThe number of nuclei of
```

```
    daughter isotope = %1.0e ", t_0, N);

// Result
// The required time for decay of daughter nucleus =
    101 yr
// The number of nuclei of daughter isotope = 1e+017
```

Chapter 3

Interactions of Nuclear Radiations with Matter

Scilab code Exa 3.1 `Alpha particle impinging on an aluminium foil`

```
// Scilab code Exa3.1 : : Page-123 (2011)
clc; clear;
E = 9;     // Energy of the alpha particle , MeV
S = 1700; //  Stopping power of Al
D = 2700; // Density of Al, Kg per cubic metre
R_air = 0.00318*E^(3/2); // Range of an alpha
    particle in air ,metre
R_Al =  R_air/S; // Range of an alpha particle in Al
    , metre
T = D*1/S; // Thickness in Al of 1m air , Kg per
    square metre
printf("\nThe range of an alpha particle = %4.2e
    metre \nThe thickness in Al of 1 m air = %4.2f Kg
     per square metre", R_Al, T);

// Result
// The range of an alpha particle = 5.05e-05 metre
// The thickness in Al of 1 m air = 1.59 Kg per
    square metre
```

Scilab code Exa 3.4 Thickness of beta absorption

```
// Scilab code Exa3.4: : Page-124 (2011)
clc; clear;
E_max = 1.17; // Maximum energy of the beta particle
    , mega electron volts
D = 2.7; // Density of Al,gram per cubic metre
u_m = 22/E_max; // Mass absorption coefficient ,
    centimetre square per gram
x_h = log(2)/(u_m*D); // Half value thickness for
    beta absorption , cm
printf("\nThe Half value thickness for beta
    absorption = %5.3f cm", x_h);

// Result
// The Half value thickness for beta absorption =
    0.014 cm
```

Scilab code Exa 3.7 Beta particles passing through lead

```
// Scilab code Exa3.7: : Page 125(2011)
clc; clear;
Z = 82;     // Atomic number
E = 1; // Energy of the beta paricle , MeV
I_l = 800;      // Ionisation loss , MeV
R = Z*E/I_l;      // Ratio of radiation loss to
    ionisation loss
E_1 = I_l/Z;      // Energy of the beta particle when
     radiation radiation loss is equal to ionisation
    loss , MeV

```

```
printf("\nThe ratio of radiation loss to ionisation
    loss = %5.3e  \nThe energy of the beta particle =
    %4.2f MeV ", R, E_1);

// Result
// The ratio of radiation loss to ionisation loss =
    1.025e-01
// The energy of the beta particle = 9.76 MeV
```

Scilab code Exa 3.8 `Thickness of gamma absorption`

```
// Scilab code Exa3.8 : : Page 125(2011)
clc; clear;
x = 0.25;      // Thickness of Al, metre
U_l = 1/x*log(50);      // Linear  absorption
    coefficient
d = 2700;                // density of the Al, Kg per
    cubic centimetre
x_h = log(2)/U_l;        // Half value thickness of Al,
     metre
U_m =  U_l/d;            // Mass absorption coefficient
    , square metre per Kg
printf("\nThe half value thickness of Al = %6.4f Kg
    per cubic metre \nThe mass absorption coefficient
     = %7.5f square metre per Kg ",x_h, U_m);

// Result
// The half value thickness of Al = 0.0443 Kg per
    cubic metre
// The mass absorption coefficient = 0.00580 square
    metre per Kg
```

Scilab code Exa 3.9 `The energy of recoil electrons`

```
// Scilab code Exa3.9 : : Page-125(2011)
clc; clear;
E_g = 2.19*1.6e-013; // Energy of the gamma rays,
    joule
m_e = 9.10939e-031; // Mass of the electron, Kg
C = 3e+08; // Velocity of light, m/s
E_max = [E_g/(1+(m_e*C^2)/(2*E_g))]/(1.6e-013); //
    Energy of the compton recoil electron, MeV
printf("\nThe energy of the compton recoil electrons
    = %5.3f MeV", E_max);

// Result
// The energy of the compton recoil electrons =
    1.961 MeV
```

Scilab code Exa 3.10 `Average energy of the positron`

```
// Scilab code Exa3.10 : : Page-125(2011)
clc; clear;
m_e = 9.1e-31;     // Mass of the positron, Kg
e = 1.6e-19;          // Charge of the positron,
    coulomb
c = 3e+08;                // Velocity of the light,
    metre per sec
eps = 8.85e-12;          // Absolute permittivity of
    free space, per N per metre-square per coulomb
    square
h = 6.6e-34;          // Planck's constant, joule sec
E = e^2*m_e*c/(eps*h*1.6e-13);          // Average
    energy of the positron, mega electron volts
printf("\nThe average energy of the positron = %6.4
    fZ MeV", E);

// Result
// The average energy of the positron = 0.0075Z MeV
```

Scilab code Exa 3.11 To calculate the refractive index of the material

```
// Scilab code Exa3.11 : : Page-125(2011)
clc; clear;
P = 1;      // Momentum of the proton, GeV/c
M_0 = 0.94;     // Rest mass of the proton, GeV/c-square
G = sqrt((P/M_0)^2+1) // Lorentz factor
V = sqrt(1-1/G^2);      // Minimum velocity of the electron, m/s
u = 1/V;      // Refractive index of the gas
printf("\nThe refractive index of the gas = %4.2f", u);
u = 1.6;      // Refractive index
theta = round (acos(1/(u*V))*180/3.14);      // Angle at which cerenkov radiatin is emitted, degree
printf("\nThe angle at which Cerenkov radiation is emitted = %d degree",theta)

// Result
//  The refractive index of the gas = 1.37
// The angle at which Cerenkov radiation is emitted = 31 degree
```

Scilab code Exa 3.12 Minimum kinetic energy of the electron to emit Cerenkov radia

```
// Scilab code Exa3.12 : : Page-126(2011)
clc; clear;
n = 1+1.35e-04; // Refractive index of the medium
V_min = 1/n; // Minimum velocity of the electron, m/s
```

```
p = (1+V_min)*(1-V_min); // It is nothing but just to take the product
G_min = 1/sqrt(p); // Lorentz  factor
m_e = 9.10939e-031; // Mass of the electron , Kg
C = 3e+08; // Velocity of light , metre per  sec
T_min = [(G_min-1)*m_e*C^2]/(1.602e-013); // Minimum  kinetic energy required by an  electro to emit cerenkov radiation , mega electron volts
printf("\nThe minimum kinetic energy required to electron to emit cerenkov radiation = %5.2f MeV", T_min);

// Result
// The minimum kinetic energy required to  electron to emit cerenkov radiation = 30.64 MeV
```

Chapter 4

Detection and Measurement of Nuclear Radiations

Scilab code Exa 4.1 Resultant pulse height recorded in the fission chamber

```
// Scilab code Exa4.1 : : Page 178 (2011)
clc; clear;
N = 200e+006/35;    // Total number of ion−pairs
e = 1.60218e-019;     // Charge of an ion, coulomb
Q = N*e;    // Total charge produced in the chamber,
    coulomb
C = 25e-012;      // Capacity of the collector, farad
V = Q/C;     // Resultant pulse height, volt
printf("\nThe resultant pulse height recorded in the
    fission chamber = %4.2e volt", V);

// Result
// The resultant pulse height recorded in the
   fission chamber = 3.66e−002 volt
```

Scilab code Exa 4.2 Energy of the alpha particles

```
// Scilab code Exa4.2 : : Page 178 (2011)
clc; clear;
V = 0.8/4;     // Pulse height , volt
e = 1.60218e-019;      // Charge of an ion , coulomb
C = 0.5e-012;      // Capacity of the collector , farad
Q = V*C;     // Total charge produced , coulomb
N = Q/e;      // Number of ion pairs
E_1 = 35;     // Energy of one ion pair , electron volt
E = N*E_1/10^6;     // Energy of the alpha particles , mega electron volt
printf("\nThe energy of the alpha particles = %4.3f MeV", E);

// Result
// The energy of the alpha particles = 21.845 MeV ( The answer is wrong in the textbook)
```

Scilab code Exa 4.3 `Height of the voltage pulse`

```
// Scilab code Exa4.3 : : Page 178 (2011)
clc; clear;
E = 10e+06;     // Energy produced by the ion pairs , electron volts
N = E/35;         // Number of ion pair produced
m = 10^3;         // Multiplication factor
N_t = N*m;         // Total number of ion pairs produced
e = 1.60218e-019;   // Charge of an ion , coulomb
Q = N_t*e;          // Total charge flow in the counter , coulomb
t = 10^-3;          // Pulse time , sec
R = 10^4;          // Resistance , ohm
I = Q/t;          // Current passes through the
```

```
resistor, ampere
V = I*R;          // Height of the voltage pulse, volt
 printf("\nTotal number of ion pairs produced: %5.3e \nTotal charge flow in the counter : %5.3e coulomb \nHeight of the voltage pulse : %5.3e volt", N_t, Q, V);

// Result
// Total number of ion pairs produced: 2.857e+008
// Total charge flow in the counter : 4.578e-011 coulomb
// Height of the voltage pulse : 4.578e-004 volt
```

Scilab code Exa 4.4 Radial field and life time of Geiger Muller Counter

```
// Scilab code Exa4.4 : : Page 178 (2011)
clc; clear;
V = 1000;     // Operating voltage of Counter, volt
x = 1e-004;           // Time taken, sec
b = 2;               // Radius of the cathode, cm
a = 0.01;          // Diameter of the wire, cm
E_r = V/(x*log(b/a)); // Radial electric field, V/m
C = 1e+009;                   // Total counts in the GM counter
T = C/(50*60*60*2000); // Life of the G.M. Counter, year
printf("\nThe radial electric field: %4.2eV/m\nThe life of the G.M. Counter : %5.3f years", E_r, T);

// Result
// The radial electric field: 1.89e+006V/m
// The life of the G.M. Counter : 2.778 years
```

Scilab code Exa 4.5 `Avalanche voltage in Geiger Muller tube`

```
// Scilab code Exa4.5 : : Page 178 (2011)
clc; clear;
I = 15.7;     // Ionisation potential of argon, eV
b = 0.025;              // Radius of the cathode,
    metre
a = 0.006e-02;          // Radius of the wire, metre
L = 7.8e-06;      // Mean free path, metre
V = round(I*a*log(b/a)/L);     // Avalanche voltage
    in G.M. tube, volt
printf("\nThe avalanche voltage in G.M. tube = %d
    volt", V);

// Result
// The avalanche voltage in G.M. tube = 729 volt
```

Scilab code Exa 4.6 `Voltage fluctuation in GM tube`

```
// Scilab code Exa4.6 :  : Page 179 (2011)
clc; clear;
C_r = 0.1e-02;      // Counting rate of GM tube
S = 3;              // Slope of the curve
V = C_r*100*100/S;        // Voltage fluctuation,
    volt
printf("\nThe voltage fluctuation GM tube = %4.2f
    volt", V);

// Result
// The voltage fluctuation GM tube = 3.33 volt
```

Scilab code Exa 4.7 `Time measurement of counts in GM counter`

```
// Scilab code Exa4.7 : : Page-179 (2011)
clc; clear;
R_t = 100;          // Actual count rate, per sec
R_B = 25;           // Backward count rate, per sec
V_S = 0.03;          // Coefficient of variation
R_S = R_t-R_B;          // Source counting rate,per sec
T_t = (R_t+sqrt(R_t*R_B))/(V_S^2*R_S^2);    // Time measurement for actual count, sec
T_B = T_t*sqrt(R_B/R_t);          // Time measurement for backward count, sec
printf("\nTime measurement for actual count : %5.3f sec \nTime measurement for backward count : %4.1f sec", T_t, T_B);

// Result
//  Time measurement for actual count : 29.630 sec
//  Time measurement for backward count : 14.8 sec
```

Scilab code Exa 4.8 Capacitance of the silicon detector

```
// Scilab code Exa4.8 : : Page-179 (2011)
clc; clear;
A = 1.5e-4;            // Area of capacitor plates, square metre
K = 12;                // Dielectric constant
D = K*8.8542e-012;    // Electrical permittivity of the medium, per newton-metre-square coulomb square
x = 50e-06;            // Width of depletion layer, metre
C = A*D/x*10^12;       // Capacitance of the silicon detector, pF
E = 4.5e+06;           // Energy produced by the ion pairs, eV
```

```
N = E/3.5;              // Number of ion pairs
e = 1.60218e-019;       // Charge of each ion, coulomb
Q = N*e;                // Total charge, coulomb
V = Q/C*10^12;          //  Potential applied across the capacitor, volt
printf("\nThe capacitance of the detector : %6.2f pF\nThe potential applied across the capacitor : %4.2e volt", C, V);

// Result
// The capacitance of the detector : 318.75 pF
// The potential applied across the capacitor : 6.46e-004 volt
```

Scilab code Exa 4.9 `Statistical error on the measured ratio`

```
// Scilab code Exa4.9 : : Page-180 (2011)
clc; clear;
N_A = 1000;         // Number of count observed for radiation A
N_B = 2000;         // Number of count observed for radiation B
r = N_A/N_B;         // Ratio of count A to the count B
E_r = sqrt(1/N_A+1/N_B);         // Statistical error
printf("\nThe statistical error of the measured ratio = %4.2f", E_r*r);

// Result
// The statistical error of the measured ratio = 0.02 (Wrong answer in the textbook)
```

Scilab code Exa 4.10 `Charge collected at the anode of photo multiplier tube`

```
// Scilab code Exa4.10 : : Page 180 (2011)
clc; clear;
E = 4e+006;          // Energy lost in the
    scintillator , eV
N_pe = E/10^2*0.5*0.1;          // Number of
    photoelectrons emitted
G = 10^6;                       // Gain of
    photomultiplier tube
e = 1.6e-019;                   // Charge of the
    electron , C
Q = N_pe*G*e;                   // Charge collected at
     the anode of photo multiplier tube , C
printf("\nThe charge collected at the anode of photo
     multiplier tube : %6.4e C", Q);

// Result
// The charge collected at the anode of photo
    multiplier tube : 3.2000e-010 C
```

Scilab code Exa 4.11 Charge collected at the anode of photo multiplier tube

```
// Scilab code Exa11 : : Page 180 (2011)
E = 4e+06;          // Energy lost in the scintillator
    , eV
N_pe = E/10^2*0.5*0.1;          // Number of
    photoelectrons emitted
G = 10^6;                       // Gain
e = 1.6e-019;                   // Charge of the
    electron , C
Q = N_pe*G*e;                       // Charge collected
     at the anode of photo multiplier tube , C
printf("\nCharge collected at the anode of photo
    multiplier tube : %6.4e C",Q);
// Result
//  Charge collected at the anode of photo
```

```
multiplier tube : 3.2000e-010 C
```

Scilab code Exa 4.12 Measurement of the number of counts and determining standard

```
// Scilab code Exa4.12 : : Page 181 (2011)
// Defining an array
clc; clear;
n = cell (1,6); // Declare the cell matrix of 1X6
n(1,1).entries = 10000;
n(2,1).entries = 10200;
n(3,1).entries = 10400;
n(4,1).entries = 10600;
n(5,1).entries = 10800;
n(6,1).entries = 11000;
g = 0; //
k = 6;
H = 0;
for i = 1:k;
    g = g + n(i,1).entries
end;
N = g/k; // Mean of the count
D = sqrt(N);
for i = 1:k;
 H = H+((n(i,1).entries-N)*(n(i,1).entries-N))
end;
S_D = round(sqrt(H/(k-1)));
printf("\nStandard deviation of the reading  : %d",
   S_D);
delta_N = sqrt(N);
if (S_D > delta_N) then
    printf("\nThe foil cannot be considered uniform
       ..!");
else
    printf("\nThe foil can be considered uniform.");
end
```

```

// Result
// Standard deviation of the reading : 374
// The foil cannot be considered uniform..!
```

Scilab code Exa 4.13 Beta particle incident on the scintillator

```
// Scilab code Exa4.13 : : Page 181 (2011)
clc; clear;
V = 2e-03;          // Voltage impulse, volt
C = 120e-012;     // Capacitance of the capacitor, F
e = 1.6e-019;     // Charge of the electron, C
n = C*V/(15*e);      // No. of electons
N = n^(1/10);      // No. of electrons in the output
printf("\nNo. of electrons in the output : %4.2f (
   approx)", N);

// Result
// No. of electrons in the output : 3.16 (approx)
```

Scilab code Exa 4.14 Time of flight of proton in scintillation counter

```
// Scilab code Exa4.14 : : Page 181 (2011)
clc; clear;
m_p = 0.938;     // Mass of the proton, GeV
E = 1.4;         // Total energy of proton, GeV
gama = E/m_p;      // Boost parameter
bta = sqrt(1-1/gama^2);     // Relativistic factor
d = 10;          // Distance between two counters,m
C = 3e+08;       // Velocity of light ,m/s
t_p = d/(bta*C);     // Time of flight of proton ,sec
T_e = d/C;          // Time of flight of electron, sec
```

```
printf("\nTime of flight of proton: %4.2f ns \nTime of flight of electron : %4.2f ns ", t_p/1e-009, T_e/1e-009);

// Result
// Time of flight of proton: 44.90 ns
// Time of flight of electron : 33.33 ns
```

Scilab code Exa 4.15 Fractional error in rest mass of the particle with a Cerenkov

```
// Scilab code Exa4.15 : : Page 182 (2011)
clc; clear;
p = 100;          // Momentum of the particle , GeV
n = 1+1.35e-04;     // Refractive index of the gas
m_0 = 1;           // Mass , GeV per square coulomb
gama = sqrt((p^2+m_0^2)/m_0);      // Boost parameter
bta = sqrt (1-1/gama^2);         // Relativistic parameter
d_theta = 1e-003;     // Error in the emission angle , radian
theta = acos(1/(n*bta));          // Emision angle of photon , radian
F_err = (p^2*n^2*2*theta*10^-3)/(2*m_0^2);         // Fractional error
printf("\nThe fractional error in rest mass of the particle = %4.2f", F_err);

// Result
// The fractional error in rest mass of the particle = 0.13
```

Scilab code Exa 4.16 Charged particles passing through the Cerenkov detector

```
// Scilab code Exa4.16 : : Page 182 (2011)
clc; clear;
u = 1.49;          // Refractive index
E = 20*1.60218e-019;          // Energy of the electron, joule
m_e = 9.1e-031;               // Mass of the electron, Kg
C = 3e-08;                    // Velocity of the light, m/s
bta = (1 + {1/(E/(m_e*C^2)+1)}^2 );    // Boost parameter
z = 1;          //
L_1 = 4000e-010;              // Initial wavelength, metre
L_2 = 7000e-010;              // Final wavelength, metre
N = 2*%pi*z^2/137*(1/L_1-1/L_2)*(1-1/(bta^2*u^2));          // Number of quanta of visible light, quanta per centimetre
printf("\nThe total number of quantas during emission of visible light = %d quanta/cm", round(N/100));

// Result
// The total number of quantas during emission of visible light = 270 quanta/cm
```

Chapter 5

Alpha Particles

Scilab code Exa 5.1 `Disintegration energy of alpha particle`

```
// Scilab code Exa5.1 : : Page 203 (2011)
clc; clear;
E_a = 8.766;     // Energy of the alpha particle , MeV
A = 212;            // Atomic mass of Po-212, amu
M_a = 4;           // Atomic mass of alpha particle , amu
e = 1.6e-019;          // Charge of an electron , coulomb
Z = 82;                    // Atomic number of Po-212
R_0 = 1.4e-015;          // Distance of closest approach , metre
K = 8.99e+09;               // Coulomb constant
E = E_a*A/(A-M_a);     // Disintegration energy , mega electron volts
B_H = 2*Z*e^2*K/(R_0*A^(1/3)*1.6*10^-13);     // Barrier height for an alpha particle within the nucleus , MeV
printf("\nDisintegration energy : %5.3f MeV \nBarrier height for alpha-particle: %5.2f MeV", E,B_H);

```

```
// Result
// Disintegration energy : 8.935 MeV
// Barrier height for alpha-particle: 28.26 MeV
```

Scilab code Exa 5.2 Calculation of the barrier height

```
// Scilab code Exa5.2 : : Page 203 (2011)
// We have to make calculation for alpha particle and for  proton
clc; clear;
E_a = 8.766;      // Energy of the alpha particle , mega electron volts
A_Bi = 209;       // Atomic mass of Bi-209, atomic mass unit
A_a = 4;          // Atomic mass of alpha particle , atomic mass unit
A_p = 1;          // Atomic mass of proton , atomic mass unit
e = 1.6e-019;     // Charge of an electron , coulomb
Z = 83;           // Atomic number of bismuth
R_0 = 1.4e-015;   // Distance of closest approach , metre
K = 8.99e+09;     // Coulomb constant
B_H_a = 2*Z*e^2*K/(R_0*1.6e-013*(A_Bi^(1/3)+A_a^(1/3)));    // Barrier height for an alpha particle , mega electron volts
B_H_p = 1*Z*e^2*K/(R_0*1.6e-013*(A_Bi^(1/3)+A_p^(1/3)));    // Barrier height for proton , mega electron volts
printf("\nBarrier height for the alpha particle = %5.2f MeV \nBarrier height for the proton = %5.2f MeV", B_H_a,B_H_p);

// Result
// Barrier height for the alpha particle = 22.67 MeV
```

```
// Barrier height for the proton = 12.30 MeV
```

Scilab code Exa 5.3 Speed and BR value of alpha particles

```
// Scilab code Exa5.3 : : Page 203 (2011)
// We have also calculate the value of magnetic
   field in a particular orbit.
clc; clear;
C = 3e+08;                    // Velocity of light, m/S
M_0 = 6.644e-027*(C)^2/(1.60218e-013);         //
   Rest mass of alpha particle, MeV
T = 5.998;                    // Kinetic energy of alpha
    particle emitted by Po-218
q = 2*1.60218e-019;          // Charge of alpha
   particle, C
V = sqrt(C^2*T*(T+2*M_0)/(T+M_0)^2);            //
   Velocity of alpha particle,metre per sec
B_r = V*M_0*(1.60218e-013)/(C^2*q*sqrt(1-V^2/C^2));
                  // magnetic field in a particular
   orbit, Web per mtere
printf("\nThe velocity of alpha particle : %5.3e m/s
   \nThe magnetic field in a particular orbit : %6.4
   f Wb/m", V , B_r);

// Result
// The velocity of alpha particle : 1.699e+007 m/s
// The magnetic field in a particular orbit : 0.3528
    Wb/m
```

Scilab code Exa 5.4 Transmission probability for an alpha particle through a poten

```
// Scilab code Exa5.4: : Page 204 (2011)
clc; clear;
```

```
a = 10^-14;          // Width of the  potential barrier, m
E = 5*1.60218e-013;          // Energy of the alpha particle, joule
V = 10*1.60218e-013;          // Potential height, joule
M_0 = 6.644e-027;          // Rest mass of the alpha particle, joule
h_red = 1.05457e-034;          // Reduced value of Planck's constant,joule sec
T = 4*exp(-2*a*sqrt(2*M_0*(V-E)/h_red^2));    // Probability of leakage through through potential barrier
printf("\nThe probability of leakage of alpha-particle through potential barrier = %5.3e ",T);

// Result
// The probability of leakage of alpha-particle through potential barrier = 1.271e-008
```

Scilab code Exa 5.6 `Difference in life times of Polonium isotopes`

```
// Scilab code Exa5.6: : Page 204 (2011)
clc; clear;
Z_D = 82;     // Atomic number of Po
E_Po210 = 5.3;     // Alpha-source for Po210, MeV
E_Po214 = 7.7;     // Alpha-source for Po214, MeV
log_lambda_Po210 = -1*1.72*Z_D*E_Po210^(-1/2);
log_lambda_Po214 = -1*1.72*Z_D*E_Po214^(-1/2);
delta_OM_t = log_lambda_Po214 - log_lambda_Po210;     // Difference in order of magnitude of life times of Po214 and Po210
printf("\nThe disintegration constant increases by a factor of some 10^%2d", delta_OM_t);

```

```
// Result
// The disintegration constant increases by a factor of some 10^10
```

Scilab code Exa 5.8 Half life of plutonium

```
// Scilab code Exa5.8: : Page 205 (2011)
clc; clear;
N = 120.1*6.023e+023/239;     // Number of Pu nuclei
P_rel = 0.231;        // Power released, watt
E_rel = 5.323*1.6026e-13;        // Energy released, joule
decay_rate = P_rel/E_rel;        // Decay rate of Pu239, per hour
t_half = N*log(2)/(decay_rate*365*86400);    // Half life of Po239, sec
printf("\nThe half life of Pu = %4.2e yr", t_half);

// Result
// The half life of Pu = 2.46e+004 yr
```

Scilab code Exa 5.9 Slope of alpha decay energy versus atomic number

```
// Scilab code Exa5.9 : : Page 205(2011)
clc; clear;
a_v = 14;        // Volume energy constant, MeV
a_s = 13;        // Surface energy constant, MeV
a_c = 0.60;      // Coulomb energy constant, MeV
a_a = 19;        // Asymmetric energy constant, MeV
A = 202;        // Mass number
Z = 82;         //  Atomic number
```

```
dE_by_dN = -8/9*a_s/A^(4/3)-4/3*a_c*Z/A^(4/3)*(1-4*Z/(3*A))-16*a_a*Z/A^2*(1-2*Z/A);          // Slope, mega electron volts per nucleon
printf("\nThe slope of alpha decay energy versus atomic number = %7.5f MeV/nucleon", dE_by_dN);

// Result
// The slope of alpha decay energy versus atomic number = -0.15007 MeV/nucleon
```

Scilab code Exa 5.10 `Degree of hindrance for alpha particle from U238`

```
// Scilab code Exa5.10 : : Page 206 (2011)
clc; clear;
h_kt = 1.05457e-34;          // Reduced Planck's constant, joule sec
e = 1.60218e-19;          // Charge of an electron, coulomb
l = 2;                    // Orbital angular momentum
eps_0 = 8.5542e-12;          // Absolute permittivity of free space, coulomb square per newton per metre square
Z_D = 90;                // Atomic number of daughter nucleus
m = 6.644e-27;            // Mass of alpha particle, Kg
R = 8.627e-15;            // Radius of daughter nucleus, metre
T1_by_T0 = exp(2*l*(l+1)*h_kt/e*sqrt(%pi*eps_0/(Z_D*m*R)));     // Hindrance factor
printf("\nThe hindrance factor for alpha particle = %5.3f" ,T1_by_T0);

// Result
// The hindrance factor for alpha particle = 1.768
```

Chapter 6

Beta Decay

Scilab code Exa 6.1 Disintegration of the beta particles by Bi210

```
// Scilab code Exa6.1: : Page-240 (2011)
clc; clear;
T = 5*24*60*60;     // Half life of the substance, sec
N = 6.023e+026*4e-06/210;          // Number of atoms
lambda = 0.693/T;                 // Disintegration constant, per sec
K = lambda*N;                    // Rate of disintegration,
E = 0.34*1.60218e-013;            // Energy of the beta particle, joule
P = E*K;                         //    Rate at which energy is emitted, watt
printf("\nThe rate at which energy is emitted = %d watt", P);

// Result
// The rate at which energy is emitted = 1 watt
```

Scilab code Exa 6.2 Beta particle placed in the magnetic field

```
// Scilab code Exa6.2 : : Page-241 (2011)
clc; clear;
M_0 = 9.10939e-031;          // Rest mass of the
    electron, Kg
C = 2.92e+08;             // Velocity of the light,
    metre per sec
E = 1.71*1.60218e-013;          // Energy of the beta
    particle, joule
e = 1.60218e-019;                // Charge of the
    electron, C
R = 0.1;                       // Radius of the orbit,
     metre
B = M_0*C*(E/(M_0*C^2)+1)*1/(R*e); // Magnetic field
     perpendicular to the beam of the particle,
    weber per square metre

printf("\nThe magnetic field perpendicular to the
    beam of the particle = %5.3f Wb/square-metre", B)
    ;

// Result
// The magnetic field perpendicular to the beam of
    the particle = 0.075 Wb/square-metre
```

Scilab code Exa 6.3 K conversion

```
// Scilab code Exa6.3 : : Page-241 (2011)
clc; clear;
m_0 = 9.10963e-031;          // Rest mass of the
    electron, Kg
e = 1.60218e-019;            // Charge of the
    electron, C
c = 2.9979e+08;            // Velocity of the light,
```

```
    metre per sec
BR = 3381e-006;      // Field-radius product, tesla-m
E_k = 37.44;     // Binding energy of k-electron
v = 1/sqrt((m_0/(BR*e))^2+1/c^2); // Velocity of the
    converson electron, m/s
E = m_0*c^2*(1/sqrt(1-v^2/c^2)-1)/(e*1e+003);    //
    Energy of the electron, keV
E_C = E+E_k;              // Energy of the converted
    gamma ray photon, KeV
printf("\nThe energy of the electron = %6.2f keV \
    nThe energy of the converted gamma ray photon =
    %6.2f keV", E, E_C);

// Result
// The energy of the electron = 624.11 keV
// The energy of the converted gamma ray photon =
    661.55 keV
```

Scilab code Exa 6.4 Average energy carried away by neutrino during beta decay proc

```
// Scilab code Exa6.4 : : Page-241 (2011)
clc; clear;
E = 18.1;                  // Energy carried by beta
    particle, keV
E_av = E/3;                 // Average energy carried
    away by beta particle, keV
E_r = E-E_av;              // The rest energy carried
    out by the neutrino, keV

printf("\nThe rest energy carried out by the
    neutrino : %5.3f KeV", E_r);

// Result
// The rest energy carried out by the neutrino :
    12.067 KeV
```

Scilab code Exa 6.5 Maximum energy available to the electrons in the beta decay of

```
// Scilab code Exa6.5: : Page-242(2011)
clc; clear;
M_Na = -8420.40;          // Mass of sodium 24, keV
M_Mg = -13933.567;        // Mass of magnesium 24, keV
E = (M_Na-M_Mg)/1000;     // Energy of the electron,
   MeV
printf("\nThe maximum energy available to the
    electrons in the beta decay = %5.3f MeV", E);

// Result
// The maximum energy available to the electrons in
   the beta decay = 5.513 MeV
```

Scilab code Exa 6.6 Linear momenta of particles during beta decay process

```
// Scilab code Exa6.6: : Page-242 (2011)
clc; clear;
c = 1;   // For simplicity assume speed of light to
   be unity, m/s
E_0 = 0.155;          // End point energy, mega
    electron volts
E_beta = 0.025;          // Energy of beta particle,
   mega electron volts
E_v = E_0-E_beta;          // Energy of the neutrino,
   mega electron volts
p_v = E_v/c;              // Linear momentum of
    neutrino, mega electron volts per c
m = 0.511;              // Mass of an electron, Kg
M = 14*1.66e-27;          // Mass of carbon 14,Kg
```

```
c = 3e+8;                    // Velocity of light, metre per sec
e = 1.60218e-19;                // Charge of an electron, coulomb
p_beta = sqrt(2*m*E_beta);     // Linear momentum of beta particle, MeV/c
sin_theta = p_beta/p_v*sind(45);     // Sine of angle theta
p_R = p_beta*cosd(45)+p_v*sqrt(1-sin_theta^2);   // Linear momemtum of recoil nucleus, MeV/c
E_R = (p_R*1.6e-13/2.9979e+08)^2/(2*M*e);   // Recoil energy of product nucleus, MeV
printf("\nThe linear momentum of neutrino = %4.2f MeV/c \nThe linear momentum of beta particle = %6.4f MeV/c \nThe energy of the recoil nucleus = %4.2f eV", p_v, p_beta, E_R);

// Result
// The linear momentum of neutrino = 0.13 MeV/c
// The linear momentum of beta particle = 0.1598 MeV/c
// The energy of the recoil nucleus = 1.20 eV
```

Scilab code Exa 6.7 Energies during disintergation of Bi210

```
// Scilab code Exa6.7: : Page-242 (2011)
clc; clear;
N = 3.7e+10*60;          // Number of disintegration, per sec
H = 0.0268*4.182;          // Heat produced at the output, joule
E = H/(N*1.6e-013);          // Energy of the beta particle, joule
M_Bi = -14.815;              // Mass of Bismuth, MeV
M_Po = -15.977;              // Mass of polonium, MeV
```

```
E_0 = M_Bi-M_Po;                // End point energy, MeV
E_ratio = E/E_0;                // Ratio of beta particle energy with end point energy
printf("\nThe energy of the beta particle = %5.3f MeV \nThe ratio of beta particle energy with end point energy = %5.3f ", E, E_ratio);

// Result
// The energy of the beta particle = 0.316 MeV
// The ratio of beta particle energy with end point energy = 0.272
```

Scilab code Exa 6.9 The unstable nucleus in the nuclide pair

```
// Scilab code Exa6.9: : Page-243(2011)
clc; clear;
M = rand(4,2);
M(1,1) = 7.0182*931.5;     // Mass of lithium, MeV
M(1,2) = 7.0192*931.5;     // Mass of beryllium, MeV
M(2,1) = 13.0076*931.5;     // Mass of carbon, MeV
M(2,2) = 13.0100*931.5;     // Mass of nitrogen, MeV
M(3,1) = 19.0045*931.5;     // Mass of fluorine, MeV
M(3,2) = 19.0080*931.5;     // Mass of neon, MeV
M(4,1) = 33.9983*931.5;     // Mass of phosphorous, MeV
M(4,2) = 33.9987*931.5;     // Mass of sulphur, MeV
j = 1;
// Check the stability !!!!
for i = 1:4
    if round (M(i,j+1)-M(i,j)) == 1 then
        printf("\n From pair a :")
        printf("\n         Be(4,7) is unstable");
      elseif round (M(i,j+1)-M(i,j)) ==  2 then
           printf("\n From pair b :")
            printf("\n         N(7,13) is unstable")
```

```
;
        elseif round (M(i,j+1)-M(i,j)) == 3 then
              printf("\n From pair c :")
              printf("\n          Ne(10,19) is unstable");
        elseif round (M(i,j+1)-M(i,j)) == 0 then
               printf("\n From pair d :")
              printf("\n          P(15,34) is unstable");
      end
end

// Result
//
// From pair a :
//          Be(4,7) is unstable
// From pair b :
//          N(7,13) is unstable
// From pair c :
//          Ne(10,19) is unstable
// From pair d :
//          P(15,34) is unstable
```

Scilab code Exa 6.10 `Half life of tritium`

```
// Scilab code Exa6.10: : Page-244 (2011)
clc; clear;
tau_0 = 7000;         // Time constant, sec
M_mod_sqr = 3;          // Nuclear matrix
E_0 = 0.018;          // Energy of beta spectrum, MeV
ft = 0.693*tau_0/M_mod_sqr;    // Comparative half life
fb = 10^(4.0*log10(E_0)+0.78+0.02);     //
t = 10^(log10(ft)-log10(fb));    // Half life of H3, sec
```

```
printf("\nThe half life of H3 = %4.2e sec", t);

// Result
// The half life of H3 = 2.44e+009 sec
```

Scilab code Exa 6.11 Degree of forbiddenness of transition

```
// Scilab code Exa6.11: : Page-244 (2011)
clc; clear;
t_p = 33/0.92*365*84800;    // Partial half life for beta emission, sec
E_0 = 0.51;         // Kinetic energy
Z = 55;             // Atomic number of cesium
log_fb = 4.0*log10(E_0)+0.78+0.02*Z-0.005*(Z-1)*log10(E_0);    // Comparitive half life
log_ft1 = log_fb+log10(t_p);     // Forbidden tansition
// For 8 percent beta minus emission
t_p = 33/0.08*365*84800;    // Partial half life, sec
E_0 = 1.17;         // Kinetic energy
Z = 55;             // Atomic energy
log_fb = 4.0*log10(E_0)+0.78+0.02*Z-0.005*(Z-1)*log10(E_0);    // Comparitive half life
log_ft2 = log_fb+log10(t_p);    // Forbidden transition
// Check the degree of forbiddenness !!!!!
if log_ft1 <= 10 then
    printf("\nFor 92 percent beta emission :")
    printf("\n\tTransition is once forbidden and parity change");
end
if log_ft2 >= 10 then
    printf("\nFor 8 percent beta emission :")
    printf("\n\t ransition is twice forbidden and no
```

```
        parity change");
end

// Result
// For 92 percent beta emission :
//  Transition is once forbidden and parity change
// For 8 percent beta emission :
//  Transition is twice forbidden and no parity
   change
```

Scilab code Exa 6.12 Coupling constant and ratio of coupling strengths for beta tr

```
// Scilab code Exa6.12: : Page-244(2011)
clc; clear;
h_kt = 1.05457e-34;     // Reduced planck's constant,
    joule sec
c = 3e+08;              // velocity of light, metre
   per sec
m_e = 9.1e-31;          // Mass of the electron, Kg
ft_O = 3162.28;         // Comparative half life for
   oxygen
ft_n = 1174.90;         // Comparative half life for
   neutron
M_f_sqr = 2             // Matrix element
g_f = sqrt(2*%pi^3*h_kt^7*log(2)/(m_e^5*c^4*ft_O*
   M_f_sqr));    // Coupling constant, joule cubic
   metre
C_ratio = (2*ft_O/(ft_n)-1)/3;     // Ratio of
   coupling strength
printf("\nThe value of coupling constant = %6.4e
   joule cubic metre\nThe ratio of coupling constant
    = %5.3f", g_f, C_ratio);

// Result
// The value of coupling constant = 1.3965e-062
```

```
    joule cubic metre
// The ratio of coupling constant = 1.461
```

Scilab code Exa 6.13 Relative capture rate in holmium for 3p to 3s sublevels

```
// Scilab code Exa6.13: : Page-245 (2011)
clc; clear;
Q_EC = 850;          // Q value for holmium 161, keV
B_p = 2.0;      // Binding energy for p-orbital electron, keV
B_s = 1.8;          // Binding energy for s-orbital electron, keV
M_ratio = 0.05*(Q_EC-B_p)^2/(Q_EC-B_s)^2;     // Matrix ratio
Q_EC = 2.5;          // Q value for holmium 163, keV
C_rate = M_ratio*(Q_EC-B_s)^2/(Q_EC-B_p)^2*100;    // The relative capture rate in holmium, percent
printf("\nThe relative capture rate in holmium 161 = %3.1f percent", C_rate);

// Result
// The relative capture rate in holmium 161 = 9.8 percent
```

Scilab code Exa 6.14 Tritium isotope undergoing beta decay

```
// Scilab code Exa6.14: : Page-246 (2011)
clc; clear;
t_half = 12.5*365*24;          // Half life of hydrogen 3, hour
lambda = log(2)/t_half;        // Decay constant, per hour
```

```
N_0 = 6.023e+26;               // Avogadro's number,
    per mole
m = 0.1e-03;                   // Mass of tritium, Kg
dN_by_dt = lambda*m*N_0/3;     // Decay rate, per hour
H = 21*4.18;                   // Heat produed, joule
E = H/dN_by_dt;                // The average energy of
     the beta particle, joule
printf("\nThe average energy of beta particles = %4
    .2e joule = %3.1f keV", E, E/1.6e-016);

// Result
// The average energy of beta particles = 6.91e-016
    joule = 4.3 keV
```

Scilab code Exa 6.15 Fermi and Gamow Teller selection rule for allowed beta transi

```
// Scilab code Exa6.15: : Page-246 (2011)
clc; clear;
S = string(rand(2,1))
S(1,1) = 'antiparallel spin'
S(2,1) = 'parallel spin'

for i = 1:2
    if S(i,1) == 'antiparallel spin' then
       printf("\nFor Fermi types :")
        printf("\n\n The selection rules for allowed
            transitions are :  \n\tdelta I is zero \
           n\tdelta pi is plus \nThe emited neutrino
            and electron have %s",S(i,1))
      elseif S(i,1) == 'parallel spin' then
        printf("\nFor Gamow-Teller types :")
        printf("\nThe selection rules for allowed
           transitions are : \n\tdelta I is zero,
           plus one and minus one\n\tdelta pi is
           plus\nThe emited neutrino and electron
```

```
have %s",S(i,1))
    end
 end
// Calculation of ratio of transition probability
M_F = 1;    // Matrix for Fermi particles
g_F = 1;        // Coupling constant of fermi particles
M_GT = 5/3;        // Matrix for Gamow Teller
g_GT = 1.24;        // Coupling constant of Gamow Teller
T_prob = g_F^2*M_F/(g_GT^2*M_GT);    // Ratio of transition probability
// Calculation of Space phase factor
e = 1.6e-19;        // Charge of an electron, coulomb
c = 3e+08;            // Velocity of light, metre per sec
K = 8.99e+9;        // Coulomb constant
R_0 = 1.2e-15;        // Distance of closest approach, metre
A = 57;            // Mass number
Z = 28;            // Atomic number
m_n = 1.6749e-27;    // Mass of neutron, Kg
m_p = 1.6726e-27;    // Mass of proton, Kg
m_e = 9.1e-31;        // Mass of electron. Kg
E_1 = 0.76;        // First excited state of nickel
delta_E = ((3*e^2*K/(5*R_0*A^(1/3))*((Z+1)^2-Z^2))-(m_n-m_p)*c^2)/1.6e-13;        // Mass difference, mega electron volts
E_0 = delta_E-(2*m_e*c^2)/1.6e-13;    // End point energy, mega electron volts
P_factor = (E_0-E_1)^5/E_0^5;    // Space phase factor
 printf("\nThe ratio of transition probability = %4.2f\nThe space phase factor = %4.2f", T_prob, P_factor);

// Result
```

```
// The emited neutrino and electron have
   antiparallel spin
// For Gamow-Teller types :
// The selection rules for allowed transitions are :
//  delta I is zero,plus one and minus one
//  delta pi is plus
// The emited neutrino and electron have parallel
   spin
// The ratio of transition probability = 0.39
// The space phase factor = 0.62
```

Chapter 7

Gamma Radiation

Scilab code Exa 7.1 Bragg reflection for first order in a bent crystal spectromete

```
// Scilab code Exa7.1: : Page-292 (2011)
clc; clear;
h = 6.6261e-034;                // Planck's constant, joule sec
C = 2.998e+08;                  // Velocity of light, metre per sec
f = 2;                          // Radius of focal circle, metre
d = 1.18e-010;                  // Interplaner spacing for quartz crystal, metre
E_1 = 1.17*1.6022e-013;         // Energy of the gamma rays, joule
E_2 = 1.33*1.6022e-013;         // Energy of the gamma rays, joule
D = h*C*f*(1/E_1-1/E_2)*1/(2*d);        //Distance to be moved for obtaining first order reflection for two different energies, metre
printf("\nThe distance to be moved for obtaining first order Bragg reflection = %4.2e metre", D);

// Result
```

```
// The distance to be moved for obtaining first
   order Bragg reflection = 1.08e-003 metre
```

Scilab code Exa 7.2 Energy of the gamma rays from magnetic spectrograph data

```
// Scilab code Exa7.2: : Page-293 (2011)
clc; clear;
m_0 = 9.1094e-031;     // Rest mass of the electron,
   Kg
B_R = 1250e-06;        // Magnetic field,tesla metre
e =  1.6022e-019;      // Charge of the electron,
   coulomb
C = 3e+08;             // Velocity of the light,
   metre per sec
E_k = 0.089;           // Binding energy of the K-
   shell electron,MeV
v = B_R*e/(m_0*sqrt(1+B_R^2*e^2/(m_0^2*C^2)));  //
   Velocity of the photoelectron, metre per sec
E_pe = m_0/(1.6022e-013)*C^2*(1/sqrt(1-v^2/C^2)-1);
    // Energy of the photoelectron,MeV
E_g = E_pe+E_k;    // Energy of the gamma rays, MeV
printf("\nThe energy of the gamma rays = %5.3f MeV",
    E_g);

// Result
// The energy of the gamma rays = 0.212 MeV
```

Scilab code Exa 7.3 Attenuation of beam of X rays in passing through human tissue

```
// Scilab code Exa7.3: : Page-292 (2011)
clc; clear;
a_c = 0.221;       // Attenuation coefficient, cm^2/g
```

```
A = (1-exp(-0.22))*100;     // Attenuation of beam of X-rays in passing through human tissue
printf("\nThe attenuation of beam of X-rays in passing through human tissue = %d percent", ceil(A));

// Result
// The attenuation of beam of X-rays in passing through human tissue = 20 percent
```

Scilab code Exa 7.4 `Partial half life for gamma emission of Hg195 isomer`

```
// Scilab code Exa7.4: : Page-293 (2011)
clc; clear;
alpha_k = 45;              // Ratio between decay constants
sum_alpha = 0.08;          // Sum of alphas
P = 0.35*1/60;             // Probability of the isomeric transition,per hour
lambda_g = P*sum_alpha/alpha_k;    // Decay constant of the gamma radiations, per hour
T_g = 1/(lambda_g*365*24);         // Partial life time for gamma emission,years
printf("\nThe partial life time for gamma emission = %5.3f years", T_g);

// Result
// The partial life time for gamma emission = 11.008 years
```

Scilab code Exa 7.5 `Estimating the gamma width from Weisskopf model`

```
// Scilab code Exa7.5: : Page-294 (2011)
```

```
clc; clear;
A = 11;                 // Mass number of boron
E_g = 4.82;             // Energy of the gamma radiation, mega electron volts
W_g = 0.0675*A^(2/3)*E_g^3;         // Gamma width, mega electron volts
printf("\nThe required gamma width = %5.2f MeV", W_g);

// Result
// The required gamma width = 37.39 MeV
```

Scilab code Exa 7.8 `K electronic states in indium`

```
// Scilab code Exa7.8:  : Page-295 (2011)
clc; clear;
e = 1.6022e-19;         // Charge of an electron, coulomb
BR = 2370e-06;          // Magnetic field in an orbit, tesla metre
m_0 = 9.1094e-31;       // Mass of an electron, Kg
c = 3e+08;              // Velocity of light, metre per sec
v = 1/sqrt((m_0/(BR*e))^2+1/c^2);       // velocity of the particle, metre per sec
E_e = m_0*c^2*((1-(v/c)^2)^(-1/2)-1)/1.6e-13;     // Energy of an electron, MeV
E_b = 0.028;         // Binding energy, MeV
E_g = E_e+E_b;         // Excitation energy, MeV
alpha_k = 0.5;         // K conversion coefficient
Z = 49;                 // Number of protons
alpha = 1/137;         // Fine structure constant
L = (1/(1-(Z^3/alpha_k*alpha^4*(2*0.511/0.392)^(15/2))))/2;     // Angular momentum
l = 1;          // Orbital angular momentum
```

```
I = 1-1/2;      // Parity
printf("\nFor K-electron state:\nThe excitation
    energy = %5.3f MeV\nThe angular momentum = %d\
    nThe parity : %3.1f", E_g, ceil(L), I);
// Result
// For K-electron state:
// The excitation energy = 0.393 MeV
// The angular momentum = 5
// The parity : 0.5
```

Scilab code Exa 7.9 Radioactive lifetime of the lowest energy electric dipole tran

```
// Scilab code Exa7.9: : Page-295 (2011)
clc; clear;
c = 3e+10;              // Velocity of light,
    centimetre per sec
R_0 = 1.4e-13;          // Distance of closest
    approach, centimetre
alpha = 1/137;          // Fine scattering constant
A = 17;                  // Mass number
E_g = 5*1.6e-06;          // Energy of gamma
    transition, ergs
h_cut = 1.054571628e-27;          // Reduced planck
    constant, ergs per sec
lambda = c/4*R_0^2*alpha*(E_g/(h_cut*c))^3*A^(2/3);
            // Disintegration constant, per sec
tau = 1/lambda;          // Radioactive lifr\e time,
    sec
printf("\nThe radioactive life time = %1.0e sec",
    tau);

// Result
// The radioactive life time = 9e-018 sec
```

Scilab code Exa 7.10 Electric and magnetic multipolarities of gamma rays from tran

```
// Scilab code Exa7.10: : Page-296 (2011)
clc; clear;
l = 2,3,4
printf("\nThe possible multipolarities are ")
for l = 2:4
    if l == 2 then
        printf("E%d,", l);
        elseif l == 3 then
        printf(" M%d", l);
        elseif l == 4 then
        printf(" and E%d", l);
    end
end
for l = 2:4
    if l == 2 then
     printf("\nThe transition E%d dominates",l);
    end
end

// Result
// The possible multipolarities are E2, M3 and E4
// The transition E2 dominates
```

Scilab code Exa 7.13 Relative source absorber velocity required to obtain resonanc

```
// Scilab code Exa7.13: : Page-297 (2011)
clc; clear;
E_0 = 0.014*1.6022e-13;    // Energy of the gamma
   rays, joule
A = 57;                    // Mass number
```

```
m = 1.67e-27;                    // Mass of each nucleon,
    Kg
c = 3e+08;                       // Velocity of light,
    metre per sec
N = 1000;                        // Number of atoms in the
    lattice
v = E_0/(A*N*m*c);               // Ralative velocity,
    metre per sec
printf("\nThe relative source absorber velocity = %5
    .3f m/s", v);

// Result
// The relative source absorber velocity = 0.079 m/s
```

Scilab code Exa 7.14 Estimating the frequency shift of a photon

```
// Scilab code Exa7.14: : Page-297 (2011)
clc; clear;
g = 9.8;                    // Acceleration due to gravity,
    metre per square sec
c = 3e+08;                       // Velocity of light,
    metre per sec
y = 20;                     // Vertical distance between
    source and absorber, metre
delta_v = g*y/c^2;               // Frequency shift
printf("\nThe required frequency shift of the photon
     = %4.2e ", delta_v);

// Result
// The required frequency shift of the photon = 2.18
    e-015
```

Chapter 8

Beta Decay

Scilab code Exa 8.3 Neutron and proton interacting within the deuteron

```
// Scilab code Exa8.3 : : Page-349 (2011)
clc; clear;
b = 1.9e-15;         // Width of square well potential , metre
h_kt = 1.054571e-034;         // Reduced planck 's constant , joule sec
c = 3e+08;                    // Velocity of light , metre per sec
m_n = 1.67e-27;               // Mass of a nucleon , Kg
V_0 = 40*1.6e-13;              // Depth , metre
E_B = (V_0-(1/(m_n*c^2)*(%pi*h_kt*c/(2*b))^2))/1.6e-13;          // Binding energy , mega electron volts
alpha = sqrt(m_n*c^2*E_B*1.6e-13)/(h_kt*c);     // scattering co efficient , per metre
P = (1+1/(alpha*b))^-1;          // Probability
R_mean = sqrt (b^2/2*(1/3+4/%pi^2+2.5));     // Mean square radius , metre
printf("\nThe probability that the proton moves within the range of neutron = %4.2f \nThe mean square radius of the deuteron = %4.2e metre", P,
```

```
    R_mean);

// Result
// The probability that the proton moves within the
    range of neutron = 0.50
// The mean square radius of the deuteron = 2.42e
    -015 metre
```

Scilab code Exa 8.5 Total cross section for np scattering at neutron energy

```
// Scilab code Exa8.5 : : Page-349 (2011)
clc; clear;
a_t = 5.38e-15;
a_s = -23.7e-15;
r_ot = 1.70e-15;
r_os = 2.40e-15;
m = 1.6748e-27;
E = 1.6e-13;
h_cut = 1.0549e-34;
K_sqr = m*E/h_cut^2;
sigma = 1/4*(3*4*%pi*a_t^2/(a_t^2*K_sqr+(1-1/2*K_sqr
    *a_t*r_ot)^2)+4*%pi*a_s^2/(a_s^2*K_sqr+(1-1/2*
    K_sqr*a_s*r_os)^2))*1e+028; // Total cross-
    section for n-p scattering , barn
printf("\nThe total cross section for n-p scattering
     = %5.3f barn", sigma);

// Result
// The total cross section for n-p scattering =
    2.911 barn
```

Scilab code Exa 6.8 Beta decayed particle emission of Li8

```
// Scilab code Exa6.8: : Page-243 (2011)
clc; clear;
l = 2;     // Orbital angular momentum quantum number
P = (+1)^2*(-1)^l;     // Parity of the 2.9 MeV level
    in Be-8
M_Li = 7.0182;          // Mass of lithium, MeV
M_Be = 7.998876;          // Mass of beryllium, MeV
m_n = 1;                  // Mass of neutron, MeV
E_th = (M_Li+m_n-M_Be)*931.5;     // Threshold energy
    , MeV
printf("\nThe parity of the 2.9 MeV level in be-8 =
    +%d \nThe threshold energy for lithium 7 neutron
    capture = %d MeV",P, E_th);

// Result
// The parity of the 2.9 MeV level in be-8 = +1
// The threshold energy for lithium 7 neutron
    capture = 18 MeV
```

Scilab code Exa 8.8 Possible angular momentum states for the deuterons in an LS co

```
// Scilab code Exa8.8 : : Page-351 (2011)
clc; clear;
S = 1;          // Spin angular momentum(s1+-s2),
    whereas s1 is the spin of proton and s2 is the
    spin of neutron.
m = 2*S+1;      // Spin multiplicity
j = 1;          // Total angular momentum
printf("\nThe possible angular momentum states with
    their parities are as follows : ");
        printf("\n          %dS%d has even parity ",
            m, j);
        printf("\n          %dP%d has odd parity ", m
            , j);
        printf("\n          %dD%d has even parity", m
```

```
               , j);
S = 0;
m = 2*S+1
    printf("\n          %dP%d has odd parity ", m, j)
        ;

// Result
// The possible angular momentum states with their
   parities are as follows :
//          3S1 has even parity
//          3P1 has odd parity
//          3D1 has even parity
//          1P1 has odd parity
```

Scilab code Exa 8.9 States of a two neutron system with given total angular moment

```
// Scilab code Exa8.9 : : Page-351 (2011)
clc; clear;
printf("\nThe possible states are : ");
// For s = 0
s = 0;                // Spin angular momentum
m = 2*s+1;            // Spin multiplicity
for j = 0:2           // Total angular momentum
    l = j
    if l == 0 then
       printf("\n      %dS%d, ", j,m);
        elseif l == 2 then
       printf(" %dD%d, ", j,m);
    end
end
// For s = 1
s = 1;
m = 2*s+1;
 l = 2
for j = 0:2
```

```
    if j == 0 then
       printf(" %dP%d, ", j,m);
       elseif j ==1 then
       printf(" %dP%d, ", j,m);
        elseif j ==2 then
       printf("%dP%d and ", j,m);
    end
end
for j = 2
    printf(" %dF%d", j,m)
end

// Result
// Possible states are :
// The possible states are :
//     0S1, 2D1, 0P3, 1P3, 2P3 and 2F3
```

Scilab code Exa 8.10 Kinetic energy of the two interacting nucleons in different f

```
// Scilab code Exa8.10 : : Page-352 (2011)
clc; clear;
r = 2e-015;           // Range of nuclear force, metre
h_kt = 1.0546e-34;    // Reduced value of Planck's constant, joule sec
m = 1.674e-27;        // Mass of each nucleon, Kg
K = round (2*h_kt^2/(2*m*r^2*1.6023e-13));        // Kinetic energy of each nucleon in centre of mass frame, mega electron volts
K_t = 2*K;        // Total kinetic energy, mega electron volts
K_inc = 2*K_t;    // Kinetic energy of the incident nucleon, mega electron volts
printf("\nThe kinetic energy of each nucleon = %d MeV\nThe total kinetic energy = %d MeV\nThe
```

```
    kinetic energy of the incident nucleon = %d MeV",
     K, K_t, K_inc);

// Result
//
```

Chapter 9

Nuclear Models

Scilab code Exa 9.1 Estimating the Fermi energies for neutrons and protons

```
// Scilab code Exa9.1 : : Page-389 (2011)
clc; clear;
h_cut = 1.054e-034;    // Reduced Planck's constant, joule sec
rho = 2e+044;     // Density of the nuclear matter, kg per metre cube
V = 238/rho;     // Volume of the nuclear matter, metre cube
// For neutron
N = 238-92;     // Number of neutrons
M = 1.67482e-027;     // Mass of a neutron, kg
e = 1.602e-019;     // Energy equivalent of 1 eV, J/eV
E_f = (3*%pi^2)^(2/3)*h_cut^2/(2*M)*(N/V)^(2/3)/e;     // Fermi energy of neutron, eV
printf("\nThe Fermi energy of neutron = %5.2f MeV", E_f/1e+006);
// For proton
N = 92;     // Number of protons
M = 1.67482e-027;     // Mass of a proton, kg
e = 1.602e-019;     // Energy equivalent of 1 eV, J/
```

```
eV
E_f = (3*%pi^2)^(2/3)*h_cut^2/(2*M)*(N/V)^(2/3)/e;    // Fermi energy of neutron , eV
printf("\nThe Fermi energy of proton = %5.2f MeV", E_f/1e+006);

// Result
// The Fermi energy of neutron = 48.92 MeV
// The Fermi energy of proton = 35.96 MeV
```

Scilab code Exa 9.3 General propeties of a neutron star

```
// Scilab code Exa9.3 : : Page-390 (2011)
clc; clear;
h_cut = 1.0545e-34; // Reduced Planck's constant , joule sec
G = 6.6e-11;          // Gravitational constant , newton square metre per square Kg
m = 10^30;          // Mass of the star , Kg
m_n = 1.67e-27;          // Mass of the neutron , Kg
R = (9*%pi/4)^(2/3)*h_cut^2/(G*(m_n)^3)*(m_n/m)^(1/3);          // Radius of the neutron star , metre
printf("\nThe radius of the neutron star = %3.1e metre", R);

// Result
// The radius of the neutron star = 1.6e+004 metre
```

Scilab code Exa 9.4 Stability of the isobar using the liquid drop model

```
// Scilab code Exa9.4 : : Page-391 (2011)
clc; clear;
```

```
A = 77;          // Mass number of the isotopes
Z = round (A/((0.015*A^(2/3))+2));     // Atomic
    number of stable isotope
// Check the stability !!!!!
  if Z == 34 then
    printf("\nSe( %d,%d) is stable \nAs (%d,%d) and
        Br(%d,%d) are unstable", Z, A, Z-1, A, Z+1, A
        );
  elseif Z == 33 then
    printf("\nAs( %d,%d) is stable \nSe (%d,%d) and
        Br(%d,%d) are unstable", Z, A, Z+1, A, Z+2, A
        );
   elseif Z == 35 then
    printf("\nBr( %d,%d) is stable \nSe (%d,%d) and
        As(%d,%d) are unstable",Z,A,Z-2,A,Z-1,A);
end

// Result
// Se( 34,77) is stable
// As (33,77) and Br(35,77) are unstable
```

Scilab code Exa 9.5 Energy difference between neutron shells

```
// Scilab code Exa9.5 : : Page-391 (2011)
clc; clear;
m_40 = 39.962589;          // Mass of calcium 40,
    atomic mass unit
m_41 = 40.962275;          // Mass of calcium 41,
    atomic mass unit
m_39 = 38.970691;           // Mass of calcium 39,
    atomic mass unit
m_n = 1.008665;             // Mass of the neutron,
    atomic mass unit
BE_1d = (m_39+m_n-m_40)*931.5;          // Binding
    energy of 1d 3/2 neutron, mega electron volts
```

```
BE_1f = (m_40+m_n-m_41)*931.5;          // Binding energy of 1f 7/2 neutron, mega electron volts
delta = BE_1d-BE_1f;          // Energy difference between neutron shells, mega electron volts
printf("\nThe energy difference between neutron shells = %4.2f MeV", delta);

// Result
// The energy difference between neutron shells = 7.25 MeV
```

Scilab code Exa 9.7 Angular frequency of the nuclei

```
// Scilab code Exa9.7 : : Page-392 (2011)
clc; clear;
h_cut = 1.0545e-34;         // Reduced Planck's constant, joule sec
R = 1.2e-15;                // Distance of closest approach, metre
m = 1.67482e-27;            // Mass of the nucleon, Kg
// For O-17
for A = 17:60               // Mass numbers
if A == 17 then
omega_O = 5*3^(1/3)*h_cut*17^(-1/3)/(2^(7/3)*m*R^2);       // Angular frequency of oxygen
// For Ni-60
elseif A == 60 then
omega_Ni =  5*3^(1/3)*h_cut*60^(-1/3)/(2^(7/3)*m*R^2);  // Angular frequency of nickel
end
end
printf("\nThe angular frequency for oxygen 17 = %4.2e \nThe angular frequency for nickel 60 = %4.2e", omega_O, omega_Ni);

```

```
// Result
// The angular frequency for oxygen 17 = 2.43e+022
// The angular frequency for nickel 60 = 1.60e+022
```

Scilab code Exa 9.9 Angular momenta and parities

```
// Scilab code Exa9.9 : : Page-393 (2011)
clc; clear;
Z = rand(5,1);
N = rand(5,1);
E = string (rand(5,1));
// Elements allocated
E(1,1) = 'Carbon'
E(2,1) = 'Boron'
E(3,1) = 'Oxygen'
E(4,1) = 'Zinc'
E(5,1) = 'Nitrogen'
Z(1,1) = 6;          // Number of proton in carbon
    nuclei
Z(2,1) = 5;           // Number of proton in boron
    nuclei
Z(3,1) = 8;           // Number of proton in oxygen
    nuclei
Z(4,1) = 30;           // Number of proton in zinc
    nuclei
Z(5,1) = 7;           // Number of proton in nitrogen
    nuclei
N(1,1) = 6;          // Mass number of carbon
N(2,1) = 6;           // Mass number of boron
N(3,1) = 9;           // Mass number of oxygen
N(4,1) = 37;           // Mass number of zinc
N(5,1) = 9;           // Mass number of nitrogem
for i = 1:5
    if Z(i,1) == 8  then
            printf("\nThe angular momentum is 5/2
```

```
and the parity is +1 for %s ", E(i,1));
    elseif Z(i,1) == 5 then
            printf("\nThe angular momentum is 3/2 and the parity is −1 for %s", E(i,1));
        end
    if Z(i,1) == N(i,1) then
        printf("\nThe angular mometum is 0 and the parity is +1 for %s", E(i,1));
    end
    if N(i,1)-Z(i,1) == 2  then
        printf("\nThe angular momentum is 2 and the parity is −1 for %s", E(i,1));
    end
    if N(i,1)-Z(i,1) == 7 then
        printf("\nThe angular momentum is 5/2 and the parity is −1 for %s", E(i,1));
    end
end

// Result
// The angular mometum is 0 and the parity is +1 for Carbon
// The angular momentum is 3/2 and the parity is −1 for Boron
// The angular momentum is 5/2 and the parity is +1 for Oxygen
// The angular momentum is 5/2 and the parity is −1 for Zinc
// The angular momentum is 2 and the parity is −1 for Nitrogen
```

Scilab code Exa 9.11 `Quadrupole and magnetic moment of ground state of nuclides`

```
// Scilab code Exa9.11 : : Page-394 (2011)
clc; clear;
R_0 = 1.2e-015;         // Distance of closest approach, metre
// Mass number of the nuclei are allocated below :
N = rand(4,1)
N(1,1) = 17;         // for oxygen
N(2,1) = 33;         // for sulphur
N(3,1) = 63;         // for copper
N(4,1) = 209;         // for bismuth
for i = 1:4

   if N(i,1) == 17 then
       printf("\n For Oxygen : ")
        I = 5/2;         // Total angular momentum
        l = 2;         // Orbital angular momentum
        mu = -1.91;         // for odd neutron and I = l+1/2
        Q = -3/5*(2*I-1)/(2*I+2)*(R_0*N(i,1)^(1/3))^2*10^28;     // Quadrupole moment of oxygen, barn
        printf("\n         The value of magnetic moment is : %4.2f \n         The value of quadrupole moment is : %6.4f barn", mu, Q);
    elseif N(i,1) == 33 then
        printf("\n\n For Sulphur : ")
         I = 3/2;         // Total angular momentum
         l = 2;                // Orbital angular momentum
         mu = 1.91*I/(I+1);         // for odd neutron and I = l-1/2
         Q = -3/5*(2*I-1)/(2*I+2)*(R_0*N(i,1)^(1/3))^2*10^28;     // Quadrupole moment of sulphur, barn
        printf("\n         The value of magnetic moment is : %5.3f \n         The value of quadrupole moment is : %6.4f barn", mu,
```

```
Q);
        elseif N(i,1) == 63 then
            printf("\n\n For Copper : ")
         I = 3/2;          // Total angular momentum
         l = 1;              // Orbital angular momentum
         mu = I+2.29;              // for odd protons and I = l+1/2
         Q = -3/5*(2*I-1)/(2*I+2)*(R_0*N(i,1)^(1/3))^2*10^28;     // Quadrupole momentum of copper, barn
        printf("\n          The value of magnetic moment is : %4.2f \n          The value of quadrupole moment is : %6.4f barn", mu, Q);
        elseif N(i,1) == 209 then
            printf("\n\n For Bismuth : ")
         I = 9/2;     // Total angular momentum
         l = 5;          // Orbital angular momentum
         mu = I-2.29*I/(I+1);      // for odd protons and I = l-1/2
         Q = -3/5*(2*I-1)/(2*I+2)*(R_0*N(i,1)^(1/3))^2*10^28;     // Quadrupole momentum of bismuth, barn
        printf("\n          The value of magnetic moment is : %4.2f \n          The value of quadrupole moment is : %5.3f barn", mu, Q);
    end
end

// Result
// For Oxygen :
//          The value of magnetic moment is : -1.91
//          The value of quadrupole moment is : -0.0326 barn

// For Sulphur :
```

```
//          The value of magnetic moment is : 1.146
//          The value of quadrupole moment is :
    -0.0356 barn

// For Copper :
//          The value of magnetic moment is : 3.79
//          The value of quadrupole moment is :
    -0.0547 barn

// For Bismuth :
//          The value of magnetic moment is : 2.63
//          The value of quadrupole moment is :
    -0.221 barn
```

Scilab code Exa 9.12 Kinetic energy of iron nucleus

```
// Scilab code Exa9.12 : : Page-395 (2011)
clc; clear;
h_cut = 1.054571628e-34;     // Redued planck's
    constant, joule sec
a = 1e-014;              // Distance of closest
    approach, metre
m = 1.67e-27;          // Mass of each nucleon, Kg
KE = 14*%pi^2*h_cut^2/(2*m*a^2*1.6e-13);        //
    Kinetic energy of iron nucleus, MeV
printf("\nThe kinetic energy of iron nuclei = %5.2f
    MeV", KE);

// Result
// The kinetic energy of iron nuclei = 28.76 MeV
```

Scilab code Exa 9.14 Electric quadrupole moment of scandium

```
// Scilab code Exa9.14 : : Page-396 (2011)
clc; clear;
R_0 = 1.2e-15;  // Distance of closest approach, metre
j = 7/2;        // Total angular momentum
A = 41;         // Mass number of Scandium
Z = 20;         // Atomic number of Calcium
Q_Sc = -(2*j-1)/(2*j+2)*(R_0*A^(1/3))^2;      // Electric quadrupole of Scandium nucleus, Sq. m
Q_Ca = Z/(A-1)^2*abs(Q_Sc);        // Electric quadrupole of calcium nucleus, Sq. m
printf("\nThe electric quadrupole of scandium nucleus = %4.2e square metre \nThe electric quadrupole of calcium nucleus = %4.2e square metre", Q_Sc, Q_Ca);

// Result
// The electric quadrupole of scandium nucleus = -1.14e-029 square metre
// The electric quadrupole of calcium nucleus = 1.43e-031 square metre
```

Scilab code Exa 9.16 Energy of lowest lying tungsten states

```
// Scilab code Exa9.16 : : Page-398 (2011)
clc; clear;
h_cut_sqr_upon_2f = 0.01667;        // A constant value, joule square per sec cube
for I = 4:6
    if I == 4 then
        E = I*(I+1)*h_cut_sqr_upon_2f;
       printf("\nThe energy for 4+ tungsten state = %5.3f MeV", E);
    elseif I == 6 then
        E = I*(I+1)*h_cut_sqr_upon_2f;
```

```
        printf("\nThe energy for 6+ tungsten state =
           %5.3f MeV", E);
    end
end

// Result
// The energy for 4+ tungsten state = 0.333 MeV
// The energy for 6+ tungsten state = 0.700 MeV
```

Chapter 10

Nuclear Reactions

Scilab code Exa 10.1 `Q value for the formation of P30 in the ground state`

```
// Scilab code Exa10.1 : : Page-455 (2011)
clc; clear;
M = 47.668;              // Total mass of reaction , MeV
E = 44.359;              // Total energy , MeV
Q = M-E;                  // Q-value , MeV
printf("\nThe Q-value for the formation of P30 = %5.3f MeV", Q);

// Result
// The Q-value for the formation of P30 = 3.309 MeV
```

Scilab code Exa 10.2 `Q value of the reaction and atomic mass of the residual nucle`

```
// Scilab code Exa10.2 : : Page-455 (2011)
clc; clear;
E_x = 7.70; // Energy of the alpha particle , MeV
E_y = 4.44; // Energy of the proton , MeV
```

```
m_x = 4.0;   // Mass number of alpha particle
m_y = 1.0;    // Mass number of protium ion
M_X = 14;    // Mass number of nitrogen nucleus
M_Y = 17;  // Mass number of oxygen nucleus
theta = 90*3.14/180; //  Angle between incident beam direction and emitted proton, degree
A_x = 4.0026033; // Atomic mass of alpha particle, u
A_X = 14.0030742; // Atomic mass of nitrogen nucleus, u
A_y = 1.0078252;  // Atomic mass of proton, u
Q = ((E_y*(1+m_y/M_Y))-(E_x*(1-m_x/M_Y))-2/M_Y*sqrt((m_x*m_y*E_x*E_y))*cos(theta))/931.5;     // Q-value, u
A_Y = A_x+A_X-A_y-Q;     // Atomic mass of O-17, u
printf("\nThe Q-value of the reaction = %9.7f u \nThe atomic mass of the O-17 = %10.7f u", Q, A_Y);

// Result
// The Q-value of the reaction = -0.0012755 u
// The atomic mass of the O-17 = 16.9991278 u
//  Atomic mass of the O-17 : 16.9991278 u
```

Scilab code Exa 10.3 Kinetic energy of the neutrons emitted at given angle to the

```
// Scilab code Exa10.3 : : Page-455 (2011)
clc; clear;
m_p = 1.007276;       // Atomic mass of the proton, u
m_H = 3.016049;         // Atomic mass of the tritium, u
m_He = 3.016029;          // Atomic mass of the He ion, u
m_n = 1.008665;           // Atomic mass of the emitted neutron, u
```

```
Q = (m_p+m_H-m_He-m_n)*931.5;          // Q-value in MeV
E_p = 3;                              // Kinetic energy of the proton , MeV
theta = 30*3.14/180;                   // angle , radian
u = sqrt(m_p*m_n*E_p)/(m_He+m_n)*cos(theta);            //
v = ((m_He*Q)+E_p*(m_He-m_p))/(m_He+m_n);                //
E_n = (u+sqrt(u^2+v))^2;                        // Kinetic energy of the emitted neutron ,MeV
printf("\nThe kinetic energy of the emitted neutron = %5.3f MeV", E_n);

// Result
// The kinetic energy of the emitted neutron = 1.445 MeV
```

Scilab code Exa 10.4 Estimating the temperature of nuclear fusion reaction

```
// Scilab code Exa10.4 : : Page-456 (2011)
clc; clear;
r_min = 4e-015;          // Distance between two deutrons , metre
k = 1.3806504e-023;        // Boltzmann 's constant , Joule per kelvin
alpha = 1/137;             // Fine structure constant
h_red = 1.05457168e-034;          // Reduced planck 's constant , Joule sec
C = 3e+08;                        // Velocity of light , meter per second
T = alpha*h_red*C/(r_min*k);
printf("\nThe temperature in the fusion reaction is = %3.1e K", T);

```

```
// Result
// The temperature in the fusion reaction is = 4.2e
   +009 K
```

Scilab code Exa 10.5 Excitation energy of the compound nucleus

```
// Scilab code Exa11.5 : : Page-456 (2011)
clc; clear;
E_0 = 4.99;          // Energy of the proton, MeV
m_p = 1;              // Mass number of the proton
m_F = 19;              // Mass number of the flourine
E = E_0/(1+m_p/m_F);         // Energy of the
   relative motion, MeV
A_F = 18.998405;         // Atomic mass of the
   fluorine, amu
A_H = 1.007276;           //  Atomic mass of the
   proton, amu
A_Ne = 19.992440;          // Atomic mass of the neon
   , amu
del_E = (A_F+A_H-A_Ne)*931.5;         // Binding
   energy of the absorbed proton, MeV
E_exc = E+del_E;              // Excitation energy of
   the compound nucleus, MeV
printf("\nThe excitation energy of the compound
   nucleus = %6.3f MeV", E_exc);

// Result
// The excitation energy of the compound nucleus =
   17.074 MeV
```

Scilab code Exa 10.6 Excitation energy and parity for compound nucleus

```
// Scilab code Exa10.6 : : Page-457 (2011)
```

```
clc; clear;
E_d = 0.6;          // Energy of the deutron, MeV
m_d = 2;            // Mass number of the deutron
m_Li = 19;          // Mass number of the Lithium
E = E_d/(1+m_d/m_Li);        // Energy of the relative motion, MeV
A_Li = 6.017;        // Atomic mass of the Lithium, amu
A_d = 2.015;         //  Atomic mass of the deutron, amu
A_Be = 8.008;        // Atomic mass of the Beryllium, amu
del_E = (A_Li+A_d-A_Be)*931.5;       // Binding energy of the absorbed proton, MeV
E_exc = E+del_E;              // Excitation energy of the compound nucleus, MeV
l_f = 2;           // orbital angular momentum of two alpha particle
P = (-1)^l_f*(+1)^2;        // Parity of the compound nucleus
printf("\nThe excitation energy of the compound nucleus = %6.3f MeV\nThe parity of the compound nucleus = %d", E_exc, P);

// Result
// The excitation energy of the compound nucleus = 22.899 MeV
// The parity of the compound nucleus = 1
```

Scilab code Exa 10.7 Cross section for neutron induced fission

```
// Scilab code Exa10.7 : : Page-457 (2011)
clc; clear;
lambda = 1e-016;        // Disintegration constant, per sec
```

```
phi = 10^11;                  // Neutron flux, neutrons per square cm per sec
sigma = 5*lambda/(phi*10^-27);                 // Cross section, milli barns
printf("\nThe cross section for neutron induced fission = %d milli barns", sigma);

// Result
// The cross section for neutron induced fission = 5 milli barns
```

Scilab code Exa 10.8 Irradiance of neutron beam with the thin sheet of Co59

```
// Scilab code Exa10.8 : : Page-457 (2011)
clc; clear;
N_0 = 6.02252e+026;          // Avogadro's constant
rho = 8.9*10^3;              // Nuclear density of Co-59, Kg per cubic metre
M = 59;                      // Mass number
sigma = 30e-028;               // Cross section, per square metre
phi = 10^16;                 // Neutron flux, neutrons per square metre per sec
d = 0.04e-02;                  // Thickness of Co-59 sheet, metre
t = 3*60*60;                  // Total reaction time, sec
t_half = 5.2*365*86400;         // Half life of Co-60, sec
lambda = 0.693/t_half;         // Disintegration constant, per sec
N_nuclei = round(N_0*rho/M*sigma*phi*d*t);         // Number of nuclei of Co-60 produced
Init_activity = lambda*N_nuclei;         // Initial activity, decays per sec
```

```
printf("\nThe number of nuclei of Co60 produced = %5
    .2e \nThe initial activity per Sq. metre = %1.0g
    decays per sec", N_nuclei, Init_activity);

// Result
// The number of nuclei of Co60 produced = 1.18e+019
// The initial activity per Sq. metre = 5e+010
    decays per sec
```

Scilab code Exa 10.9 Bombardment of protons on Fe54 target

```
// Scilab code Exa10.9 : : Page-458 (2011)
clc; clear;
d = 0.1;                    // Thickness of Fe-54 sheet,
     Kg per squre metre
M = 54;                      // Mass number of Fe
m = 1.66e-027;               // Mass of the proton, Kg
n = d/(M*m);                // Number of nuclei in unit
    area of the target, nuclei per square metre
ds = 10^-5;                 // Area, metre square
r = 0.1;                    // Distance between detector
     and target foil, metre
d_omega =ds/r^2;            // Solid angle, steradian
d_sigma = 1.3e-03*10^-3*10^-28;          //
    Differential cross section, square metre per
    nuclei
P = d_sigma*n;                    // Probablity, event
    per proton
I = 10^-7;                        // Current, ampere
e = 1.6e-19;                   // Charge of the proton,
     C
N = I/e;               // Number of protons per second
    in the incident beam, proton per sec
dN = P*N;               // Number of events detected
    per second, events per sec
```

```
printf("\nThe number of events detected = %d events per sec", dN);

// Result
// The number of events detected = 90 events per sec
```

Scilab code Exa 10.10 `Fractional attenuation of neutron beam on passing through ni`

```
// Scilab code Exa10.10 : : Page-458 (2011)
clc; clear;
N_0 = 6.02252e+26;          // Avogadro's constant
sigma = 3.5e-28;              // Cross section, square metre
rho = 8.9e+03;                  // Nuclear density, Kg per cubic metre
M = 58;                           // Mass number
summation = rho/M*N_0*sigma;     // Macroscopic cross section, per metre
x = 0.01e-02;                    // Thickness of nickel sheet, metre
I0_ratio_I = exp(summation*x/2.3026);    // Fractional attenuation of neutron beam on passing through nickel sheet
printf("\nThe fractional attenuation of neutron beam on passing through nickel sheet = %6.4f", I0_ratio_I);

// Result
// The fractional attenuation of neutron beam on passing through nickel sheet = 1.0014
// Wrong answer given in the textbook
```

Scilab code Exa 10.11 `Scattering contribution to the resonance`

```
// Scilab code Exa10.11 : : Page-458 (2011)
clc; clear;
lambda = sqrt(1.45e-021/(4*%pi));        // Wavelength, metre
W_ratio = 2.3e-07;                          // Width ratio
sigma = W_ratio*(4*%pi)*lambda^2*10^28;    // Scattering contribution, barn
printf("\nThe scattering contribution to the resonance = %4.2f barns", sigma);

// Result
// The scattering contribution to the resonance = 3.33 barns
```

Scilab code Exa 10.12 Estimating the relative probabilities interactions in the in

```
// Scilab code Exa10.12 : : Page-458 (2011)
clc; clear;
sigma = 2.8e-024;          // Cross section, metre square
lambda = 2.4e-11;          // de Broglie wavelength, metre
R_prob = %pi*sigma/lambda^2;          // Relative probabilities of (n,n) and (n,y) in indium
printf("\nThe relative probabilities of (n,n) and (n,y) in indium = %5.3f", R_prob);

// Result
// The relative probabilities of (n,n) and (n,y) in indium = 0.015
```

Scilab code Exa 10.13 Peak cross section during neutron capture

```
// Scilab code Exa10.13 : : Page-459 (2011)
clc; clear;
h = 6.625e-34;              // Planck's constant, joule sec
m_n = 1.67e-27;            // Mass of neutron, Kg
E = 4.906;            // Energy, joule
w_y = 0.124;                // radiation width, eV
w_n = 0.007*E^(1/2);                // Probability of elastic emission of neutron, eV
I = 3;                    // Total angular momentum
I_c = 2;                    // Total angular momentum in the compound state
sigma = ((h^2)*(2*I_c+1)*w_y*w_n)*10^28/(2*%pi*m_n*E*1.602e-019*(2*I+1)*(w_y+w_n)^2);    // Cross section, barns
printf("\nThe cross section of neutron capture = %5.3e barns", sigma);

// Result
// The cross section of neutron capture = 3.755e+004 barns
```

Scilab code Exa 10.14 Angle at which differential cross section is maximumat a giv

```
// Scilab code Exa10.14 : : Page-459 (2011)
clc; clear;
R = 5;                    // Radius, femto metre
k_d = 0.98;                // The value of k for deutron
k_p = 0.82;             // The value of k for triton
theta = rand(1,5);          // Angles at which differetial cross section is maximum, degree
// Use of for loop for angles calculation(in degree)
for l = 0:4
    theta = round((acos((k_d^2+k_p^2)/(2*k_d*k_p)-l
```

```
       ^2/(2*k_d*k_p*R^2)))*180/3.14);
    printf("\nFor l = %d", l);
    printf(",the value of theta_max = %d degree",
       ceil(theta));
    end

// Result
// For l = 0,the value of theta_max = 0 degree
// For l = 1,the value of theta_max = 8 degree
// For l = 2,the value of theta_max = 24 degree
// For l = 3,the value of theta_max = 38 degree
// For l = 4,the value of theta_max = 52 degree
```

Scilab code Exa 10.15 Estimating the angular momentum transfer

```
// Scilab code Exa10.15 : : Page-459 (2011)
clc; clear;
k_d = 2.02e+30;          // The value of k for deutron
k_t = 2.02e+30;          // The value of k for triton
theta = 23*3.14/180;          // Angle, radiams
q = sqrt (k_d+k_t-2*k_t*cos(theta))*10^-15;
    // the value of q in femto metre
R_0 = 1.2;          // Distance of closest approach,
    femto metre
A = 90;               // Mass number of Zr-90
z = 4.30;               // Deutron size, femto metre
R = R_0*A^(1/3)+1/2*z;          // Radius of the
    nucleus, femto metre
l = round(q*R);               // Orbital angular
    momentum
I = l+1/2                    // Total angular
    momentum
printf("\nThe total angular momentum transfer = %3.1
    f ", I);

```

```
// Result
// The total angular momentum transfer = 4.5
```

Chapter 11

Particle Accelerators

Scilab code Exa 11.1 `Optimum number of stages and ripple voltage in Cockcroft Walt`

```
// Scilab code Exa11.1 : : Page-535(2011)
clc; clear;
V_0 = 10^5;          // Accelerating voltage, volts
C = 0.02e-006;          // Capacitance, farad
I = 4*1e-003;              // Current, ampere
f = 200;               // Frequency, cycles per sec
n = sqrt (V_0*f*C/I);     // Number of particles
delta_V = I*n*(n+1)/(4*f*C);
printf("\nThe optimum number of stages in the
    accelerator = %d", n);
printf("\nThe ripple voltage = %4.1f kV", delta_V/1e
    +003);

// Result
// The optimum number of stages in the accelerator =
    10
// The ripple voltage = 27.5 kV
```

Scilab code Exa 11.2 `Charging current and potential of an electrostatic generator`

```
// Scilab code Exa11.2 : : Page-536 (2011)
clc; clear;
s = 15;          // Speed, metre per sec
w = 0.3;          // Width of the electrode, metre
E = 3e+06;          // Breakdown strength, volts per metre
eps = 8.85e-12;     // Absolute permitivity of free space, farad per metre
C = 111e-12;          // Capacitance, farad
i = round (2*eps*E*s*w*10^6);     // Current, micro ampere
V = i/C*10^-12;               // Rate of rise of electrode potential, mega volts per sec
printf("\nThe charging current = %d micro-ampere \nThe rate of rise of electrode potential = %4.2f MV/sec", i, V);

// Result
// The charging current = 239 micro-ampere
// The rate of rise of electrode potential = 2.15 MV/sec
```

Scilab code Exa 11.3 Linear proton accelerator

```
// Scilab code Exa11.3 : : Page-536 (2011)
clc; clear;
f = 200*10^6;          // Frequency of the accelerator, cycle per sec
M = 1.6724e-27;          // Mass of the proton, Kg
E = 45.3*1.6e-13;          // Accelerating energy, joule
L_f = round (1/f*sqrt(2*E/M)*100);     // Length of the final drift tube, centi metre
L_1 = 5.35*10^-2;                    // Length of the first drift tube, metre
```

```
K_E = (1/2*M*L_1^2*f^2)/1.6e-13;     // Kinetic energy of the injected proton , MeV
E_inc = E/1.6e-13-K_E;          // Increase in energy , MeV
q = 1.6e-19;                    // Charge of the proton , C
V = 1.49e+06;                // Accelerating voltage , volts
N = E_inc*1.6e-13/(q*V);     // Number of drift protons
L = 1/f*sqrt(2*q*V/M)*integrate('n^(1/2)', 'n', 0, N);     // Total length of the accelerator , metre
printf("\nThe length of the final drift tube = %d cm\nThe kinetic energy of the injected protons = %4.2f MeV\nThe total length of the accelerator = %3.1f metre", L_f, K_E, L);

// Result
// The length of the final drift tube = 47 cm
// The kinetic energy of the injected protons = 0.60 MeV
// The total length of the accelerator = 9.2 metre
```

Scilab code Exa 11.5 Energy and the frequency of deuterons accelerated in cyclotro

```
// Scilab code Exa11.5 : : Page-536 (2011)
clc; clear;
B = 1.4;          // Magnetic field , tesla
R = 88e-002;          // Radius of the orbit , metre
q = 1.6023e-019;                    // Charge of the deutron , C
M_d = 2.014102*1.66e-27;          // Mass of the deutron , Kg
M_He = 4.002603*1.66e-27;          // Mass of the He ion , Kg
```

```
E = B^2*R^2*q^2/(2*M_d*1.6e-13);        // Energy og
    the emerging deutron , mega electron volts
f = B*q/(2*%pi*M_d)*10^-6;          // Frequency of
    the deutron voltage , mega cycles per sec
B_He = 2*%pi*M_He*f*10^6/(2*q);     // Magnetic field
    required for He(++) ions , weber per square metre
B_change = B-B_He;          // Change in magnetic
    field , tesla
printf("\nThe energy of the emerging deutron = %4.1f
    MeV\nThe frequency of the dee voltage = %5.2f
    MHz\nThe change in magnetic field = %4.2f tesla",
    E, f, B_change);

// Result
// The energy of the emerging deutron = 36.4 MeV
// The frequency of the dee voltage = 10.68 MHz
// The change in magnetic field = 0.01 tesla
```

Scilab code Exa 11.6 Protons extracted from a cyclotron

```
// Scilab code Exa11.6: : Page-537 (2011)
clc; clear;
K_E = 7.5*1.6023e-13;           // Kinetic energy ,
    joule
r = 0.51;                       // Radius of the proton
    's orbit , metre
E = 5*10^6;                     // Electric field , volts
    per metre
m = 1.67e-27;               // Mass of the proton , Kg
q = 1.6023e-19;                     // Charge of the
    proton , C
v = sqrt(2*K_E/m);          // Velocity of the proton ,
    metre per sec
B_red = E/v;                    // The effective
    reduction in magnetic field , tesla
```

```
B = m*v/(q*r);              // Total magnetic field
   produced , tesla
r_change = r*B_red/B;          // The change in orbit
   radius , metre
 printf("\nThe effective reduction in magnetic field
    = %5.3f tesla  \nThe change in orbit radius =
    %5.3f metre ", B_red, r_change);

// Result
// The effective reduction in magnetic field = 0.132
    tesla
// The change in orbit radius = 0.087 metre
```

Scilab code Exa 11.7 Energy of the electrons in a betatron

```
// Scilab code Exa11.7 : : Page-537 (2011)
clc; clear;
B = 0.4;          // Magnetic field , tesla
e = 1.6203e-19;          // Charge of an electron , C
R = 30*2.54e-02;          // Radius , metre
c = 3e+08;               // Capacitance , farad
E = B*e*R*c/1.6e-13;          // The energy of the
   electron , mega electron volts
f = 50;                  // Frequency , cycles per sec
N = c/(4*2*%pi*f*R);          // Total number of
   revolutions
Avg_E_per_rev = E*1e+006/N;          // Average energy
    gained per revolution , electron volt
printf("\nThe energy of the electron = %4.1f MeV\
   nThe average energy gained per revolution = %6.2f
    eV", E, Avg_E_per_rev);

// Result
// The energy of the electron = 92.6 MeV
// The average energy gained per revolution = 295.57
```

```
eV
// Note: Wrong answer is given in the textbook
//   Average energy gained per revolution : 295.57 electron volts
```

Scilab code Exa 11.8 Electrons accelerated into betatron

```
// Scilab code Exa11.8 : : Page-537 (2011)
clc; clear;
R = 0.35;              // Orbit radius, metre
N = 100e+06/480;          // Total number of revolutions
L = 2*%pi*R*N;              // Distance traversed by the electron, metre
t = 2e-06;                  // Pulse duration, sec
e = 1.6203e-19;              // Charge of an electron, C
n = 3e+09;                  // Number of electrons
f = 180;                  // frequency, hertz
I_p = n*e/t;              // Peak current, ampere
I_avg = n*e*f;              // Average current, ampere
tau = t*f;                  // Duty cycle
 printf("\nThe peak current = %3.1e ampere  \nThe average current = %4.2e ampere \nThe duty cycle = %3.1e", I_p, I_avg, tau);

// Result
// The peak current = 2.4e-004 ampere
// The average current = 8.75e-008 ampere
// The duty cycle = 3.6e-004
```

Scilab code Exa 11.9 Deuterons accelerated in synchrocyclotron

```
// Scilab code Exa11.9 : : Page-538 (2011)
clc; clear;
q = 1.6023e-19;         // Charge of an electron, C
B_0 = 1.5;              // Magnetic field at the centre, tesla
m_d = 2.014102*1.66e-27;        // Mass of the deutron, Kg
f_max = B_0*q/(2*%pi*m_d*10^6);         // Maximum frequency of the dee voltage, mega cycles per sec
B_prime = 1.4310;           // Magnetic field at the periphery of the dee, tesla
f_prime = 10^7;             // Frequency, cycles per sec
c = 3e+08;              // Velocity of the light, metre per sec
M = B_prime*q/(2*%pi*f_prime*1.66e-27);         // Relativistic mass, u
K_E = (M-m_d/1.66e-27)*931.5;           // Kinetic energy of the particle, mega electron volts
 printf("\nThe maximum frequency of the dee voltage = %5.2f MHz\nThe kinetic energy of the deuteron = %5.1f MeV", f_max, K_E);

// Result
// The maximum frequency of the dee voltage = 11.44 MHz
// The kinetic energy of the deuteron = 171.6 MeV
```

Scilab code Exa 11.10 Electrons accelerated in electron synchrotron

```
// Scilab code Exa11.10 : : Page-538 (2011)
clc; clear;
e = 1.6023e-19;         // Charge of an electron, C
E = 70*1.6e-13;         // Energy, electron volts
R = 0.28;             // Radius of the orbit, metre
```

```
c = 3e+08;              // Velocity of light, metre per sec
B = E/(e*R*c);          // Magnetic field intensity, tesla
f = e*B*c^2/(2*%pi*E);          // Frequency, cycle per sec
del_E = 88.5*(0.07)^4*10^3/(R);      // Energy radiated by an electron, electron volts
printf("\nThe frequency of the applied electric field = %5.3e cycles per sec \nThe magnetic field intensity = %4.3f tesla\nThe energy radiated by the electron = %3.1f eV", f, B, del_E);

// Result
// The frequency of the applied electric field = 1.705e+008 cycles per sec
// The magnetic field intensity = 0.832 tesla
// The energy radiated by the electron = 7.6 eV
```

Scilab code Exa 11.11 Kinetic energy of the accelerated nitrogen ion

```
// Scilab code Exa11.11 : : Page-538 (2011)
clc; clear;
E = 3;          // Energy of proton synchrotron, giga electron volts
m_0_c_sq = 0.938;          // Relativistic energy, mega electron volts
P_p = sqrt(E^2-m_0_c_sq^2);          // Momentum of the proton, giga electron volts per c
P_n = 6*P_p;          // Momentum of the N(14) ions, giga electron volts
T_n = sqrt(P_n^2+(0.938*14)^2)-0.938*14;          // Kinetic energy of the accelerated nitrogen ion
printf("\nThe kinetic energy of the accelerated
```

```
    nitrogen ion = %4.2f MeV", T_n);

// Result
// The kinetic energy of the accelerated nitrogen
    ion = 8.43 MeV
```

Scilab code Exa 11.12 Maximum magnetic flux density and frequency of proton in cos

```
// Scilab code Exa11.12 : : Page-539 (2011)
clc; clear;
e = 1.6e-19;          // Charge of an electron, C
R = 9.144;               // Radius, metre
m_p = 1.67e-027;          // Mass of the proton, Kg
E = 3.6*1.6e-13;          // Energy, joule
L = 3.048;           // Length of the one synchrotron
    section, metre
T = 3;               // Kinetic energy, giga electron
    volts
c = 3e+08;           // Velocity of the light, metre
    per sec
m_0_c_sq = 0.938;     // Relativistic energy, mega
    electron volts
B = round (sqrt(2*m_p*E)/(R*e)*10^4);          //
    Maximum magnetic field density, web per square
    metre
v = B*10^-4*e*R/m_p;          // Velocity of the
    proton, metre per sec
f_c = v/(2*%pi*R*10^6);          // Frequency of the
    circular orbit, mega cycles per sec
f_0 = 2*%pi*R*f_c*10^3/(2*%pi*R+4*L);     // Reduced
    frequency, kilo cycles per sec
B_m = 3.33*sqrt(T*(T+2*m_0_c_sq))/R;     //
    Relativistic field, web per square metre
f_0 = c^2*e*R*B*1e-004/((2*%pi*R+4*L)*(T+m_0_c_sq)*e
    *1e+015);     // Maximum frequency of the
```

```
   accelerating voltage, mega cycles per sec
printf("\nThe maximum magnetic flux density = %5.3f
    weber/Sq.m\nThe maximum frequency of the
    accelerating voltage = %4.2f MHz", B_m, f_0);

// Result
// The maximum magnetic flux density = 1.393 weber/
   Sq.m
// The maximum frequency of the accelerating voltage
   = 0.09 MHz
// Answer is given wrongly in the textbook
```

Scilab code Exa 11.13 Energy of the single proton in the colliding beam

```
// Scilab code Exa11.13 : : Page-539 (2011)
clc; clear;
E_c = 30e+009;           // Energy of the proton
   accelerator, GeV
m_0_c_sq = 0.938*10^6;          // Relativistic energy
   , GeV
E_p = (4*E_c^2-2*m_0_c_sq^2)/(2*m_0_c_sq) ;     //
   Energy of the proton, GeV
printf("\nThe energy of the proton = %5.2e GeV", E_p
   /1e+009);

// Result
// The energy of the proton = 1.92e+006 GeV
// Wrong answer given in the textbook
```

Scilab code Exa 11.14 Energy of the electron during boson production

```
// Scilab code Exa11.14 : : Page-539 (2011)
clc; clear;
```

```
M_z = 92;          // Mass of the boson, giga electron volts
E_e = M_z/2;          // Energy of the electron, giga electron volts
c = 3e+08;          // Velocity of the light, metre per second
m_e = 9.1e-31*c^2/(1.6e-019*1e+009);          // Mass of electron, giga electron volts
E_e_plus = M_z^2/(2*m_e);          // Threshold energy for the positron, giga electron volts
 printf("\nThe energy of the electron = %d GeV\nThe threshold energy of the positron = %4.2e GeV", E_e, E_e_plus);

// Result
// The energy of the electron = 46 GeV
// The threshold energy of the positron = 8.27e+006 GeV
```

Chapter 12

Neutrons

Scilab code Exa 12.1 Maximum activity induced in 100 mg of Cu foil

```
// Scilab code Exa12.1 : : Page-573 (2011)
clc; clear;
N_0 = 6.23e+23;    // Avogadro's number, per mole
m = 0.1;          // Mass of copper foil, Kg
phi = 10^12;          // Neutron flux density, per
    square centimetre sec
a_63 = 0.691;          // Abundance of Cu-63
a_65 = 0.309;          // Abundance of Cu-65
W_m = 63.57;          // Molecular weight, gram
sigma_63 =  4.5e-24;     // Activation cross section
    for Cu-63, square centi metre
sigma_65 = 2.3e-24;          // Activation cross
    section for Cu-65, square centi metre
A_63 = phi*sigma_63*m*a_63/W_m*N_0;          //
    Activity for Cu-63, disintegrations per sec
A_65 = phi*sigma_65*m*a_65/W_m*N_0;          //
    Activity for Cu-65, disintegrations per sec
printf("\nThe activity for Cu-63 is = %4.3e
    disintegrations per sec \nThe activity for Cu-65
    is = %4.2e disintegrations per sec", A_63, A_65);

```

```
// Result
// The activity for Cu-63 is = 3.047e+009
   disintegrations per sec
// The activity for Cu-65 is = 6.97e+008
   disintegrations per sec
```

Scilab code Exa 12.2 `Energy loss during neutron scattering`

```
// Scilab code Exa12.2 : : Page-573 (2011)
clc; clear;
A_Be = 9;          // Mass number of beryllium
A_U = 238;         // Mass number of uranium
E_los_Be = (1-((A_Be-1)^2/(A_Be+1)^2))*100;     //
   Energy loss for beryllium
E_los_U = round((1-((A_U-1)^2/(A_U+1)^2))*100);
   // Energy loss for uranium
printf("\nThe energy loss for beryllium is = %d
   percent \nThe energy loss for uranium is = %d
   percent", E_los_Be, E_los_U);

// Check for greater energy loss !!!!
if E_los_Be >= E_los_U then
    printf("\nThe energy loss is greater for
       beryllium");
else
    printf("\nThe energy loss is greater for uranium
       ");
end

// Result
// The energy loss for beryllium is = 36 percent
// The energy loss for uranium is = 2 percent
// The energy loss is greater for beryllium
```

Scilab code Exa 12.3 Energy loss of neutron during collision with carbon

```
// Scilab code Exa12.3 : : Page-574 (2011)
clc; clear;
A = 12;          // Mass number of Carbon
alpha = (A-1)^2/(A+1)^2;          // Scattering
   coefficient
E_loss = 1/2*(1-alpha)*100;          // Energy loss of
    neutron
printf("\nThe energy loss of neutron = %5.3f percent
   ",E_loss)

// Result
// The energy loss of neutron = 14.201 percent
```

Scilab code Exa 12.4 Number of collisions for neutron loss

```
// Scilab code Exa12.4 : : Page-574 (2011)
clc; clear;
zeta = 0.209;          // Moderated assembly
E_change = 100/1;          // Change in energy of the
   neutron
E_thermal = 0.025;          // Thermal energy of the
   neutron, electron volts
E_n = 2*10^6;          // Energy of the neutron,
   electron volts
n = 1/zeta*log(E_change);          // Number of
   collisions of neutrons to loss 99 percent of
   their energies
n_thermal = 1/zeta*log(E_n/E_thermal);          //
   Number of collisions of neutrons to reach thermal
    energies
```

```
printf("\nThe number of collisions  of neutrons to loss 99 percent of their energies = %d \nThe number of collisions of neutrons to reach thermal energies = %d",n,n_thermal)

// Result
// The number of collisions  of neutrons to loss 99 percent of their energies = 22
// The number of collisions of neutrons to reach thermal energies = 87
```

Scilab code Exa 12.5 Average distance travelled by a neutron

```
// Scilab code Exa12.5 : : Page-574 (2011)
clc; clear;
L = 1;    // For simplicity assume thermal diffusion length to be unity, unit
x_bar = integrate('x*exp(-x/L)', 'x', 0, 100);    // Average distance travelled by the neutron, unit
x_rms = sqrt(integrate('x^2*exp(-x/L)', 'x', 0, 100));    // Root mean square of the distance trvelled by the neutron, unit
printf("\nThe average distance travelled by the neutron = %d*L", x_bar);
printf("\nThe root mean square distance travelled by the neutron = %5.3fL = %5.3fx_bar", x_rms, x_rms);

// Result
// The average distance travelled by the neutron = 1*L
// The root mean square distance travelled by the neutron = 1.414L = 1.414x_bar
```

Scilab code Exa 12.6 Neutron flux through water tank

```
// Scilab code Exa12.6 : : Page-574 (2011)
clc; clear;
Q = 5e+08;          // Rate at which neutrons produce,
     neutrons per sec
r = 20;              // Distance from the source,
    centi metre
// For water
lambda_wtr = 0.45;     //  Transport mean free path,
    centi metre
L_wtr = 2.73;          // Thermal diffusion length,
    centi metre
phi_wtr = 3*Q/(4*%pi*lambda_wtr*r)*exp(-r/L_wtr);
       // Neutron flux for water, neutrons per square
     centimetre per sec
// For heavy water
lambda_h_wtr = 2.40;         //  Transport mean free
    path, centi metre
L_h_wtr = 171;              // Thermal diffusion
    length, centi metre
phi_h_wtr = 3*Q/(4*%pi*lambda_h_wtr*r)*exp(-r/
    L_h_wtr);     // Neutron flux for heavy water,
    neutrons per square centimetre per sec
printf("\nThe neutron flux through water = %5.3e
    neutrons per square cm per sec \nThe neutron flux
     through heavy water = %5.3e neutrons per square
    cm per sec", phi_wtr, phi_h_wtr);

// Result
// The neutron flux through water = 8.730e+003
    neutrons per square cm per sec
// The neutron flux through heavy water = 2.212e+006
     neutrons per square cm per sec
```

Scilab code Exa 12.7 Diffusion length and neutron flux for thermal neutrons

```
// Scilab code Exa12.7 : : Page-575 (2011)
clc; clear;
k = 1.38e-23;          // Boltzmann constant , joules
    per kelvin
T = 323;              // Temperature , kelvin
E = (k*T)/1.6e-19;     // Thermal energy , joules
sigma_0 = 13.2e-28;    // Cross section , square metre
E_0 = 0.025;             // Energy of the neutron ,
    electron volts
sigma_a = sigma_0*sqrt(E_0/E);         // Absorption
    cross section , square metre
t_half = 2.25;              // Half life , hours
lambda = 0.69/t_half;          // Decay constant , per
    hour
N_0 = 6.023e+026;              // Avogadro's number ,
    per
m_Mn = 55;                  // Mass number of mangnese
w = 0.1e-03;              // Weight of mangnese foil ,
    Kg
A = 200;                  // Activity , disintegrations
     per sec
N = N_0*w/m_Mn;         // Number of mangnese nuclei
    in the foil
x1 = 1.5;                  // Base , metre
x2 = 2.0;                  // Height , metre
phi = A/(N*sigma_a*0.416);         // Neutron flux ,
    neutrons per square metre per sec
phi1 = 1;     // For simplicity assume initial
    neutron flux to be unity , neutrons/Sq.m-sec
phi2 = 1/2*phi1;     // Given neutron flux , neutrons/
    Sq.m-sec
L1 = 1/log(phi1/phi2)/(x2-x1);     // Thermal
```

```
    diffusion length for given neutron flux, m
L = sqrt(1/((1/L1)^2+(%pi/x1)^2+(%pi/x2)^2));
          // Diffusion length, metre
printf("\nThe neutron flux = %3.2e neutrons per
    square metre per sec \nThe diffusion length = %4
    .2f metre", phi, L);

// Result
// The neutron flux = 3.51e+008 neutrons per square
    metre per sec
// The diffusion length = 0.38 metre
// Note: the difussion length is solved wrongly in
    the testbook
```

Scilab code Exa 12.8 Diffusion length for thermal neutrons in graphite

```
// Scilab code Exa12.8 : : Page-575(2011)
clc; clear;
N_0 = 6.023e+026;          // Avogadro's number, per
    mole
rho = 1.62e+03;          // Density, kg per cubic
    metre
sigma_a = 3.2e-31;          // Absorption cross
    section, square metre
sigma_s = 4.8e-28;          // Scattered cross section
    , square metre
A = 12;                   // Mass number
lambda_a = A/(N_0*rho*sigma_a);          // Absorption
     mean free path, metre
lambda_tr = A/(N_0*rho*sigma_s*(1-2/(3*A)));
    // Transport mean free path, metre
L = sqrt(lambda_a*lambda_tr/3);          // Diffusion
    length for thermal neutron
printf("\nThe diffusion length for thermal neutron =
     %5.3f metre ",L)
```

```

// Result
// The diffusion length for thermal neutron = 0.590 metre
```

Scilab code Exa 12.9 Neutron age and slowing down length of neutrons in graphite a

```
// Scilab code Exa12.9 : : Page-575 (2011)
clc; clear;
E_0 = 2e+06;         // Average energy of the neutron, electron volts
E = 0.025;             // Thermal energy of the neutron, electron volts
// For graphite
A = 12               // Mass number
sigma_g = 33.5;     // The value of sigma for graphite
tau_0 = 1/(6*sigma_g^2)*(A+2/3)/(1-2/(3*A))*log(E_0/E);     // Age of neutron for graphite, Sq.m
L_f = sqrt(tau_0);     // Slowing down length of neutron through graphite, m
printf("\nFor Graphite, A = %d", A);
printf("\nNeutron age = %d Sq.cm", tau_0*1e+004);
printf("\nSlowing down length = %5.3f m", L_f);
// For beryllium
A = 9               // Mass number
sigma_b = 57;     // The value of sigma for beryllium
tau_0 = 1/(6*sigma_b^2)*(A+2/3)/(1-2/(3*A))*log(E_0/E);     // Age of neutron for beryllium, Sq.m
L_f = sqrt(tau_0);     // Slowing down length of neutron through graphite, m
printf("\n\nFor Beryllium, A = %d", A);
printf("\nNeutron age = %d Sq.cm", tau_0*1e+004);
printf("\nSlowing down length = %3.1e m", L_f);

```

```
// Result
// For Graphite, A = 12
// Neutron age = 362 Sq.cm
// Slowing down length = 0.190 m

// For Beryllium, A = 9
// Neutron age = 97 Sq.cm
// Slowing down length = 9.9e-002 m
```

Scilab code Exa 12.10 Energy of the neutrons reflected from the crystal

```
// Scilab code Exa12.10 : : Page-576 (2011)
clc; clear;
theta = 3.5*%pi/180;    // Reflection angle, radian
d = 2.3e-10;            // Lattice spacing, metre
n = 1;                  // For first order
h = 6.6256e-34;         // Planck's constant, joule sec
m = 1.6748e-27;         // Mass of the neutron, Kg
E = n^2*h^2/(8*m*d^2*sin(theta)^2*1.6023e-19);   // Energy of the neutrons, electron volts
printf("\nThe energy of the neutrons = %4.2f eV", E);

// Result
// The energy of the neutrons = 1.04 eV
```

Chapter 13

Nuclear Fission and Fusion

Scilab code Exa 13.1 Fission rate and energy released during fission of U235

```
// Scilab code Exa13.1 : : Page-600 (2011)
clc; clear;
E = 200*1.6023e-13;      // Energy released per fission , joule
E_t = 2;                 // Total power produced , watt
R_fiss = E_t/E;         // Fission rate , fissions per sec
m = 0.5;                 // Mass of uranium , Kg
M = 235;                 // Mass number of uranium
N_0 = 6.023e+26;         // Avogadro's number , per mole
N = m/M*N_0              // Number of uranium nuclei
E_rel = N*E/4.08*10^-3;         // Energy released , kilocalories
printf("\nThe rate of fission of U-235 = %4.2e fissions per sec \nEnergy released = %e kcal", R_fiss, E_rel);

// Result
// The rate of fission of U-235 = 6.24e+010 fissions
```

```
    per sec
// Energy released = 1.006535e+010 kcal
```

Scilab code Exa 13.2 Number of free neutrons in the reactor

```
// Scilab code Exa13.2 : : Page-600 (2011)
clc; clear;
E = 200*1.6e-13;          // Energy released per fission, joules per neutron
t = 10^-3;                // Time, sec
P = E/t;                  // Power produced by one free neutron, watt per neutron
P_l = 10^9;               // Power level, watt
N = P_l/P;                // Number of free neutrons in the reactor, neutrons
printf("\nThe number of free neutrons in the reactor = %5.3e neutrons", N);

// Result
// The number of free neutrons in the reactor = 3.125e+016 neutrons
```

Scilab code Exa 13.3 Number of neutrons released per absorption

```
// Scilab code Exa13.3 : : Page-600 (2011)
clc; clear;
N_0_235 = 1;          // Number of uranium 235 per 238
N_0_238 = 20;         // Number of uranium 238 for one uranium 235
sigma_a_235 = 683;    // Absorption cross section for uranium 235, barn
sigma_a_238 = 2.73;   // Absorption cross section for uranium 238, barn
```

```
sigma_f_235 = 583;    // Fission cross section , barn
sigma_a = (N_0_235*sigma_a_235+N_0_238*sigma_a_238)
   /(N_0_235+N_0_238); //Asorption cross sec , barn
sigma_f = N_0_235*sigma_f_235/(N_0_235+N_0_238);
           // Fisssion cross section
v = 2.43;
eta = v*sigma_f/sigma_a;    //    Average number of
   neutron released per absorption
printf("\nThe average number of neutrons released
   per absorption = %5.3f", eta);

// Result
// The average number of neutrons released per
   absorption = 1.921
```

Scilab code Exa 13.4 Excitation energy for uranium isotopes

```
// Scilab code Exa13.4 : : Page-600(2011)
clc; clear;
a_v = 14.0;         // Volume binding energy constant
   , mega electron volts
a_s = 13.0;         // Surface binding energy
   constant , mega electron volts
a_c = 0.583;         // Coulomb constant , mega
   electron volts
a_a = 19.3;         // Asymmetric constant , mega
   electron volts
a_p = 33.5;         // Pairing energy constant , mega
   electron volts
Z = 92;             // Atomic number
// For U-236
A = 235;            // Mass number
E_exc_236 = a_v*(A+1-A)-a_s*((A+1)^(2/3)-A^(2/3))-
   a_c*(Z^2/(A+1)^(1/3)-Z^2/A^(1/3))-a_a*((A+1-2*Z)
   ^2/(A+1)-(A-2*Z)^2/A)+a_p*(A+1)^(-3/4);        //
```

```
Excitation energy for uranium 236, mega electron volts
// For U-239
A = 238;              // Mass number
E_exc_239 = a_v*(A+1-A)-a_s*((A+1)^(2/3)-A^(2/3))-a_c*(Z^2/(A+1)^(1/3)-Z^2/A^(1/3))-a_a*((A+1-2*Z)^2/(A+1)-(A-2*Z)^2/A)+a_p*((A+1)^(-3/4)-A^(-3/4));     // Excitation energy for uranium 239
// Now calculate the rate of spontaneous fissioning for U-235
N_0 = 6.02214e+23;         // Avogadro's constant, per mole
M = 235;                   // Mass number
t_half = 3e+17*3.15e+7;         // Half life, years
lambda = 0.693/t_half;        // Decay constant, per year
N = N_0/M;                    // Mass of uranium 235, Kg
dN_dt = N*lambda*3600;              // Rate of spontaneous fissioning of uranium 235, per hour
printf("\nThe excitation energy for uranium 236 = %3.1f MeV\nThe excitation energy for uranium 239 = %3.1f MeV\nThe rate of spontaneous fissioning of uranium 235 = %4.2f per hour", E_exc_236, E_exc_239, dN_dt);

// Result
// The excitation energy for uranium 236 = 6.8 MeV
// The excitation energy for uranium 239 = 5.9 MeV
// The rate of spontaneous fissioning of uranium 235 = 0.68 per hour
```

Scilab code Exa 13.5 Total energy released in fusion reaction

```
// Scilab code Exa13.5 : : Page-601 (2011)
```

```
clc; clear;
a = 10^5;          // Area of the lake, square mile
d = 1/20;          // Depth of the lake, mile
V = a*d*(1.6e+03)^3;    // Volume of the lake, cubic metre
rho = 10^3;              // Density of water, kg per cubic metre
M_water = V*rho;          // Total mass of water in the lake, Kg
N_0 = 6.02214e+26;        // Avogadro's constant, per mole
A = 18;                   // Milecular mass of water
N = M_water*N_0/A;        // Number of molecules of water, molecules
abund_det = 0.0156e-02; // Abundance of deterium
N_d = N*2*abund_det;      // Number of deterium atoms
E_per_det = 43/6;         // Energy released per deterium atom, mega electron volts
E_t = N_d*E_per_det;      // Total energy released during fusion, mega electron volt
printf("\nThe total energy released during fusion = %4.2e MeV", E_t);

// Result
// Total energy released during fusion = 1.53e+039 MeV
```

Scilab code Exa 13.6 Maximum temperature attained by thermonuclear device

```
// Scilab code Exa13.6 : : Page-601 (2011)
clc; clear;
r = 1/2;              // Radius of the tube, metre
a = %pi*r^2;          // Area of the torus, square metre
V = 3*%pi*a;          // Volume of the torus, cubic
```

```
    metre
P = 10^-5*13.6e+3*9.81;    // Pressure of the gas,
    newton per square metre
C = 1200e-6;         // Capacitance, farad
v = 4e+4;             //  potential, volts
T_room = 293;         // Room temperature, kelvin
N_k = P*V/T_room;     // From gas equation
E = 1/2*C*v^2;        // Energy stored, joules
T_k = 1/6*E/(N_k*10);     // Temperature attained by
    thermonuclear device, kelvin
printf("\nThe temperature attained by thermonuclear
    device = %4.2e K", T_k);

// Result
// The temperature attained by thermonuclear device
   = 4.75e+005 K
```

Scilab code Exa 13.7 Energy radiated and the temperature of the sun

```
// Scilab code Exa13.7 : : Page-601 (2011)
clc; clear;
G = 6.67e-11;         // Gravitational constant,
    newton square m per square kg
r = 7e+08;         // Radius of the sun, metre
M_0 = 2e+30;         // Mass of the sun, kg
E_rel = 3/5*G*M_0^2/r;       // Energy released by
    the sun, joule
E_dia_shrink_10 = E_rel/9;   // Energy released when
    sun diameter shrink by 10 percent, joule
R = 8.314;        // Universal gas constant, joule
    per kelvin per kelvin per mole
T = E_rel/(M_0*R);  // Temperature of the sun,
    kelvin
printf("\nThe energy released by the sun = %4.2e
    joule \nThe energy released when sun diameter is
```

```
    shrinked by 10 percent = %4.2e joule \nThe
    temperature of the sun = %4.2e kelvin ",E_rel,
    E_dia_shrink_10, T);

// Result
// The energy released by the sun = 2.29e+041 joule
// The energy released when sun diameter is shrinked
     by 10 percent = 2.54e+040 joule
// The temperature of the sun = 1.38e+010 kelvin
```

Scilab code Exa 13.8 Estimating the Q value for symmetric fission of a nucleus

```
// Scilab code Exa13.8 : : Page-602 (2011)
clc; clear;
A_0 = 240;          // Mass number of parent nucleus
A_1 = 120;          // Mass number of daughter nucleus
B_120 = 8.5;           // Binding energy of daughter
    nucleus
B_240 = 7.6;           //  Binding energy of parent
    nucleus
Q = 2*A_1*B_120-A_0*B_240;     // Estimated Q-value,
    mega electron volts
printf("\nThe estimated Q-value is = %d MeV", Q);

// Result
// The estimated Q-value is = 216 MeV
```

Scilab code Exa 13.9 Estimating the asymmetric binding energy term

```
// Scilab code Exa13.9 : : Page-602 (2011)
clc; clear;
E = 31.7;          // Energy, MeV
```

```
a_a = 5/9*2^(-2/3)*E;          // Asymmetric binding
    energy term, mega electron volts
printf("\nThe asymmetric binding energy term = %4.1f
    MeV", a_a);

// Result
// The asymmetric binding energy term = 11.1 MeV
```

Chapter 15

Nuclear Fission Reactors

Scilab code Exa 15.1 Estimation of the leakage factor for thermal reactor

```
// Scilab code Exa15.1 : : Page-652 (2011)
clc; clear;
N_0_235 = 1;          // Number of uranium atom
N_0_c = 10^5;          // Number of graphite atoms per
    uranium atom
sigma_a_235 = 698;     // Absorption cross section
   for uranium, barns
sigma_a_c = 0.003;          // Absorption cross
   section for graphite, barns
f = N_0_235*sigma_a_235/(N_0_235*sigma_a_235+N_0_c*
   sigma_a_c );    // Thermal utilization factor
eta = 2.08;          // Number of fast fission neutron
    produced
k_inf = eta*f;          // Multiplication factor
L_m = 0.54;              // Material length, metre
L_sqr = ((L_m)^2*(1-f));    // diffusion length,
   metre
tau = 0.0364;                   // Age of the neutron
B_sqr = 3.27;             // Geometrical buckling
k_eff = round (k_inf*exp(-tau*B_sqr)/(1+L_sqr*B_sqr)
   );    // Effective multiplication factor
```

```
N_lf = k_eff/k_inf;     // Non leakage factor
lf = (1-N_lf)*100;         // Leakage factor , percent
printf("\n Total leakage factor = %4.1f percent",lf)

// Result
//  Total leakage factor = 31.3 percent
```

Scilab code Exa 15.2 Neutron multiplication factor of uranium reactor

```
// Scilab code Exa15.2 : : Page-652 (2011)
clc; clear;
N_m = 50;          // Number of molecules of heavy
   water per uranium molecule
N_u = 1;          // Number of uranium molecules
sigma_a_u = 7.68;          // Absorption cross section
    for uranium , barns
sigma_s_u = 8.3;          // Scattered cross section
   for uranium , barns
sigma_a_D = 0.00092;     // Absorption cross section
   for heavy water , barns
sigma_s_D = 10.6;          // Scattered cross section
   for uranium , barns
f = N_u*sigma_a_u/(N_u*sigma_a_u+N_m*sigma_a_D );
          // Thermal utilization factor
zeta = 0.570;          // Average number of collisions
N_0 = N_u*139/140;          // Number of U-238 atoms
   per unit volume
sigma_s = N_m/N_0*sigma_s_D;     // Scattered cross
   section , barns
sigma_a_eff = 3.85*(sigma_s/N_0)^0.415;     //
   Effective absorption cross section , barns
p = exp(-sigma_a_eff/sigma_s);          // Resonance
   escape probablity
eps = 1;                    // Fast fission factor
eta = 1.34;               // Number of fast fission
```

```
    neutron produced
k_inf = eps*eta*p*f;          // Effective
    multiplication factor
printf("\nNeutron multiplication factor = %4.1f ",
    k_inf);

// Result
// Neutron multiplication factor =  1.2
```

Scilab code Exa 15.3 Multiplication factor for uranium graphite moderated assembly

```
// Scilab code Exa15.3 : : Page-652 (2011)
clc; clear;
// For graphite
sigma_a_g = 0.0032;          // Absorption cross
    section for graphite, barns
sigma_s_g = 4.8;          // Scattered cross section
    for graphite, barns
zeta = 0.158;          // Average number of collisions
N_m = 50;          // Number of molecules of graphite
    per uranium molecule
// For uranium
sigma_f = 590;      // Fissioning cross section,
    barns
sigma_a_u = 698;          // Absorption cross section
    for U-235, barns
sigma_a_238 = 2.75;          // Absorption cross
    section for U-238, barns
v = 2.46;              // Number of fast neutrons
    emitted
N_u = 1              // Number of uranium atoms
f = N_u*sigma_a_u/(N_u*sigma_a_u+N_m*sigma_a_g );
            // Thermal utilization factor
N_0 = N_u*(75/76);          // Number of U-238 atoms
    per unit volume
```

```
sigma_s = N_m*76/75*sigma_s_g/N_u;          //
    Scattered cross section , barns
sigma_eff = 3.85*(sigma_s/N_0)^0.415;         //
    Effective cross section , barns
p = exp(-sigma_eff/sigma_s);          // Resonance
    escape probability , barns
eps = 1;         // Fast fission factor
eta = 1.34;          // Number of fast fission neutron
     produced
k_inf = eps*eta*p*f;          // Multiplication factor
printf("\nThe required multiplication factor = %3.1f
    ", k_inf);

// Result
// The required multiplication factor = 1.1
```

Scilab code Exa 15.4 Ratio of number of uranium atoms to graphite atoms

```
// Scilab code Exa15.4 : : Page-653 (2011)
clc; clear;
eta = 2.07;          // Number of fast fission neutron
     produced
x = 1/(eta-1);
sigma_a_u = 687;    // Absorption cross section for
    uranium , barns
sigma_a_g = 0.0045; // Absorption cross section for
    graphite , barns
N_ratio = x*sigma_a_g/sigma_a_u;     // Ratio of
    number of uranium atoms to graphite atoms
printf("\nThe ratio of number of uranium atoms to
    graphite atoms = %4.2e ", N_ratio);

// Result
// The ratio of number of uranium atoms to graphite
    atoms = 6.12e-006
```

Scilab code Exa 15.5 `Multiplication factor for LOPO nuclear reactor`

```
// Scilab code Exa15.5 : : Page-653 (2011)
clc; clear;
f = 0.754;          // Thermal utilization factor
sigma_s_o = 4.2;          // Scattered cross section for oxygen, barns
sigma_s_H = 20;          // Scattered cross section for hydrogen, barns
N_O = 879.25;          // Number of oxygen atoms
N_238 = 14.19;          // Number of uranium atoms
N_H = 1573;               // Number of hydrogen atoms
sigma_s = N_O/N_238*sigma_s_o+N_H/N_238*sigma_s_H;          // Scattered cross section, barns
N_0 = 14.19;          // Number of U-238 per unit volume
zeta_o = 0.120;     // Number of collision for oxygen
zeta_H = 1;          // Number of collision for hydrogen
sigma_eff = (N_0/(zeta_o*sigma_s_o*N_O+zeta_H*sigma_s_H*N_H));          // Effective cross section, barns
p = exp(-sigma_eff/sigma_s);          // Resonance escape probablity
eta = 2.08;          // Number of fission neutron produced.
eps = 1;          // Fission factor
K_inf = eps*eta*p*f;     // Multiplication factor
printf("\nThe multiplication factor for LOPO reactor = %3.1f ", K_inf);

// Result
// The multiplication factor for LOPO reactor = 1.6
```

Scilab code Exa 15.6 Control poison required to maintain the criticality of U235

```
// Scilab code Exa15.6 : : Page-654 (2011)
clc; clear;
r = 35;          // Radius of the reactor, centi metre
B_sqr = (%pi/r)^2;     // Geometrical buckling, per
    square centi metre
D = 0.220;          // Diffusion coefficient, centi
    metre
sigma_a_f = 0.057;     // Rate of absorption of
    thermal neutrons
v = 2.5;          // Number of fast neutrons emitted
tau = 50;          // Age of the neutron
sigma_f = 0.048;     // Rate of fission
sigma_a_c = -1/(1+tau*B_sqr)*(-v*sigma_f+sigma_a_f+
    B_sqr*D+tau*B_sqr*sigma_a_f);         //
    Controlled cross section
printf("\nThe required controlled cross section = %6
    .4f ", sigma_a_c);

// Result
// The required controlled cross section = 0.0273
```

Scilab code Exa 15.7 Dimensions of a reactor

```
// Scilab code Exa15.7 : : Page-655 (2011)
clc; clear;
B_sqr = 65;          // Geometrical buckling
a = sqrt(3*%pi^2/B_sqr)*100;     // Side of the
    cubical reactor, centi metre
R = round(%pi/sqrt(B_sqr)*100); // Radius of the
    cubical reactor,centi metre
```

```
printf("\nThe side of the cubical reactor = %4.1f cm\nThe critical radius of the reactor = %d cm", a, R);

// Result
// The side of the cubical reactor = 67.5 cm
// The critical radius of the reactor = 39 cm
```

Scilab code Exa 15.8 Critical volume of the spherical reactor

```
// Scilab code Exa15.8 : : Page-655 (2011)
clc; clear;
sigma_a_u = 698;          // Absorption cross section for uranium, barns
sigma_a_M = 0.00092;          // Absorption cross section for heavy water, barns
N_m = 10^5;          // Number of atoms of heavy water
N_u = 1;          // Number of atoms of uranium
f = sigma_a_u/(sigma_a_u+sigma_a_M*N_m/N_u);     // Thermal utilization factor
eta = 2.08;          // Number of fast fission neutron produced
k_inf = eta*f;          // Multiplication factor
L_m_sqr = 1.70;          // Material length, metre
L_sqr = L_m_sqr*(1-f);     // Diffusion length, metre
B_sqr = 1.819/0.30381*exp(-1/12)-1/0.3038;     // Geometrical buckling, per square metre
V_c = 120/(B_sqr*sqrt(B_sqr));          // Volume of the reactor, cubic metre
printf("\nThe critical volume of the reactor = %4.1f cubic metre", V_c);

// Result
// The critical volume of the reactor = 36.4 cubic metre
```

Chapter 16

Chemical and Biological Effects of Radiation

Scilab code Exa 16.1 Radiation dosimetry

```
// Scilab code Exa16.1 : : Page-672 (2011)
clc; clear;
R_d = 25;         // Radiation dose, milli rad
R_c_gy = 25e-03; // Dose in centigray
R_Sv = 25*10^-2  // Dose in milli sieverts
printf("\n25 mrad = %2.0e cGy = %4.2f mSv", R_c_gy,
   R_Sv);

// Results
// 25 mrad = 3e-002 cGy = 0.25 mSv
```

Scilab code Exa 16.2 Conversion of becquerel into curie

```
// Scilab code Exa16.2 : : Page-673 (2011)
clc; clear;
BC_conv = 100*1e+009/3.7e+10;        // Becquerel
   curie conversion, milli curie
```

```
printf("\n100 mega becquerel = %3.1f milli curie ",
    BC_conv)

// Results
// 100 mega becquerel = 2.7 milli curie
```

Scilab code Exa 16.4 Amount of liver dose for a liver scan

```
// Scilab code Exa16.4 : : Page-673 (2011)
clc; clear;
A = 80*10^6;          // Activity , becquerel
t_half = 6*3600;      // Half life , s
N = A*t_half/0.693;      // Number of surviving
    radionuclei
E_released = 0.9*N*(140e+03)*1.6e-19;     // Energy
    released , joule
m_l = 1.8;                          // Mass of liver of
    average man, Kg
liv_dose = E_released*10^2/m_l;     // Liver dose ,
    centigray
printf("\nThe requiresd liver dose = %3.1f cGy",
    liv_dose);

// Result
// The requiresd liver dose = 2.8 cGy
```

Chapter 18

Elementary Particles

Scilab code Exa 18.1 Root mean square radius of charge distribution

```
// Scilab code Exa18.1 : : Page-770 (2011)
clc; clear;
m_sqr = 0.71;          // For proton, (GeV/c-square)^2
R_rms = sqrt(12)/(sqrt(m_sqr)*5.1);     // Root mean
    square radius, femto metre
printf("\nThe root mean square radius of charge
    distribution: %4.2f fermi", R_rms);

// Result
// The root mean square radius of charge
    distribution: 0.81 fermi
```

Scilab code Exa 18.3 Isospin of the strange particles

```
// Scilab code Ex18.3 : : Page-763 (2011)
clc; clear;
p = rand(1,2);              // proton
pi_minus = rand(1,2);       //pi minus meson
```

```
pi_plus = rand(1,2);          // pi plus meson
n = rand(1,2);                // neutron
lamda_0 = rand(1,2);          // lamda hyperon
K_0 = rand(1,2);              // K zero (Kaons)
K_plus = rand(1,2);           // K plus (Kaons)
sigma_plus = rand(1,2);       // hyperon
sigma_minus = rand(1,2)       // hyperon
ksi_minus = rand(1,2);        // hyperon
// Allocate the value of Isospins (T and T3)
p(1,1) = 1/2;
p(1,2) = 1/2;
pi_minus(1,1) = 1;
pi_minus(1,2) = -1;
pi_plus(1,1) = 1;
pi_plus(1,2) = +1;
n(1,1) = 1/2;
n(1,2) = -1/2;
lambda_0(1,1) = 0;
lambda_0(1,2) = 0;
K_0(1,1) = pi_minus(1,1)+p(1,1);
K_0(1,2) = pi_minus(1,2)+p(1,2) ;
K_plus(1,1) = p(1,1)+p(1,1)-lambda_0(1,1)-p(1,1);
K_plus(1,2) = p(1,2)+p(1,2)-lambda_0(1,2)-p(1,2) ;
sigma_plus(1,1) = pi_plus(1,1)+p(1,1)-K_plus(1,1);
sigma_plus(1,2) =  pi_plus(1,2)+p(1,2)-K_plus(1,2);
sigma_minus(1,1) =  pi_minus(1,1)+p(1,1)-K_plus(1,1)
   ;
sigma_minus(1,2) =  pi_minus(1,2)+p(1,2)-K_plus(1,2)
   ;
ksi_minus(1,1) = pi_plus(1,1)+n(1,1)-K_plus(1,1)-
   K_plus(1,1);
ksi_minus(1,2) = pi_plus(1,2)+n(1,2)-K_plus(1,2)-
   K_plus(1,2);
printf("\n  Reaction I \n          pi_minus + p
   ......> lambda_0  + K_0");
printf("\n The value of T for K_0 is : %3.1f ",K_0
   (1,1));
printf("\n The value of T3 for K_0 is : %3.1f ",K_0
```

```
(1,2));
printf("\n  Reaction II \n          pi_plus + p ->
   lambda_0  + K_plus");
printf("\n The value of T for K_plus is : %3.1f ",
   K_plus(1,1));
printf("\n The value of T3 for K_plus is : %3.1f ",
   K_plus(1,2));
printf("\n  Reaction III \n          pi_plus + n ->
   lambda_0 + K_plus");
printf("\n The value of T for K_plus is : %3.1f ",
   K_plus(1,1));
printf("\n The value of T3 for K_plus is : %3.1f ",
   K_plus(1,2));
printf("\n  Reaction VI \n          pi_minus + p ->
   sigma_minus  + K_plus");
printf("\n The value of T for sigma_minus is : %3.1f
    ",sigma_minus(1,1));
printf("\n The value of T3 for sigma_minus is : %3.1
   f ",sigma_minus(1,2));
printf("\n  Reaction V \n          pi_plus + p ->
   sigma_plus  + K_plus");
printf("\n The value of T for sigma_plus is : %3.1f
   ",sigma_plus(1,1));
printf("\n The value of T3 for sigma_plus is : %3.1f
    ",sigma_plus(1,2));
printf("\n  Reaction VI \n          pi_plus + n ->
   ksi_minus  + K_plus + K_plus");
printf("\n The value of T for Ksi_minus is : %3.1f "
   ,ksi_minus(1,1));
printf("\n The value of T3 for Ksi_minus is : %3.1f
   ",ksi_minus(1,2));

// Result
//
//  Reaction I
//         pi_minus + p -> lambda_0  + K_0
// The value of T for K_0 is : 1.5
// The value of T3 for K_0 is : -0.5
```

```
//  Reaction  II
//           pi_plus + p -> lambda_0   + K_plus
// The value of T for K_plus is : 0.5
// The value of T3 for K_plus is : 0.5
//  Reaction  III
//           pi_plus + n -> lambda_0   + K_plus
// The value of T for K_plus is : 0.5
// The value of T3 for K_plus is : 0.5
//  Reaction  VI
 //          pi_minus + p -> sigma_minus   + K_plus
// The value of T for sigma_minus is : 1.0
// The value of T3 for sigma_minus is : -1.0
//  Reaction  V
//           pi_plus + p -> sigma_plus   + K_plus
// The value of T for sigma_plus is : 1.0
// The value of T3 for sigma_plus is : 1.0
//  Reaction  VI
 //          pi_plus + n -> ksi_minus   + K_plus + K_plus
// The value of T for Ksi_minus is : 0.5
// The value of T3 for Ksi_minus is : -0.5
```

Scilab code Exa 18.4 `Allowed and forbidden reactions under conservation laws`

```
// Scilab code Exa18.4 : : Page-764 (2011)
clc;clear;
p = rand(1,3);              // proton
pi_minus = rand(1,3);       // pi minus meson
pi_plus = rand(1,3);        // pi plus meson
pi_0 = rand(1,3);           // pi zero meson
n = rand(1,3);              // neutron
lambda_0 = rand(1,3);       // lambda zero hyperon
K_0 = rand(1,3);            // k zero meson
K_plus = rand(1,3);         // k plus meson
K_0_bar = rand(1,3);        // anti particle of k zero
```

```
sigma_plus = rand(1,3);    // sigma hyperon
// Now in the following steps we allocated the value of charge(Q),baryon number(B) and strangeness number (S)
p(1,1) = 1;
p(1,2) = 1;
p(1,3) = 0;
pi_minus(1,1) = -1;
pi_minus(1,2) = 0;
pi_minus(1,3) = 0;
pi_plus(1,1) = 1;
pi_plus(1,2) = 0;
pi_plus(1,3) = 0;
n(1,1) = 0;
n(1,2) = 1;
n(1,3) = 0;
lambda_0(1,1) = 0;
lambda_0(1,2) = 1;
lambda_0(1,3) = -1;
K_0(1,1) =0 ;
K_0(1,2) = 0 ;
K_0(1,3) = 1;
K_plus(1,1) = 1;
K_plus(1,2) = 0 ;
K_plus(1,3) = 1;
sigma_plus(1,1) = 1;
sigma_plus(1,2) =  1;
sigma_plus(1,3) = -1;
K_0_bar(1,1) = 0;
K_0_bar(1,2) = 0;
K_0_bar(1,3) = -1;
pi_0(1,1)  = 0;
pi_0(1,2)  = 0;
pi_0(1,3)  = 0;
j = 0;
k = 0;
printf("\n  Reaction I \n          pi_plus + n ......> lambda_0 + K_plus")
```

```
for i = 1:3
    if pi_plus(1,i)+n(1,i) == lambda_0(1,i)+K_plus(1,i) then
     j = j+1;
    else
     printf("\n Reaction I is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero")
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
end

if j==3 then
    printf("\n Reaction I is allowed ");
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero")
end
printf("\n  Reaction II \n          pi_plus + n ......> K_0 + K_plus")
j = 0;
for i = 1:3
    if pi_plus(1,i)+n(1,i) == K_0(1,i)+K_plus(1,i) then
     j = j+1;
    else
     printf("\n Reaction II is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero")
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
```

```
end

if j==3 then
    printf("\n Reaction II is allowed ");
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero")
end
j = 0;
printf("\n  Reaction III \n          pi_plus + n ......> K_0_bar + sumison_plus")
for i = 1:3
    if pi_plus(1,i)+n(1,i) == K_0_bar(1,i)+sigma_plus(1,i) then
     j = j+1;
    else
     printf("\n Reaction III is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero")
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
end

if j==3 then
    printf("\n Reaction III is allowed ");
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero")
end
j = 0;
printf("\n  Reaction IV \n          pi_plus + n ......> pi_minus  + p")
for i = 1:3
    if pi_plus(1,i)+n(1,i) == pi_minus(1,i)+p(1,i) then
     j = j+1;
```

```
    else
     printf("\n Reaction IV is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero")
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
end

if j==3 then
    printf("\n Reaction IV is allowed ");
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero")
end
j = 0;
printf("\n  Reaction V \n          pi_minus + p ......> lambda_0  + K_0")
for i = 1:3
    if pi_minus(1,i)+p(1,i) == lambda_0(1,i)+K_0(1,i) then
     j = j+1;
    else
     printf("\n Reaction V is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero")
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
end

if j==3 then
    printf("\n Reaction V is allowed ");
```

```
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero")
end
j = 0;
printf("\n  Reaction VI \n         pi_plus + n ......> lambda_0  + K_plus")
for i = 1:3
    if pi_minus(1,i)+p(1,i) == pi_0(1,i)+lambda_0(1,i) then
     j = j+1;
    else
     printf("\n Reaction VI is forbidden")
     if i == 1 then
         printf("\n Delta Q is not zero");
      elseif i == 2 then
         printf("\n Delta B is not zero")
        elseif i == 3 then
         printf("\n Delta S is not zero")
      end
   end
end

if j==3 then
    printf("\n Reaction VI is allowed ");
   printf("\n Delta Q is zero \n Delta B is zero \n Delta S is zero");
end

// Result
// Reaction I
 //        pi_plus +   n ......> lambda_0  + K_plus
// Reaction I is allowed
// Delta Q is zero
// Delta B is zero
// Delta S is zero
//  Reaction II
 //        pi_plus +   n ......> K_0  + K_plus
//  Reaction II is forbidden
```

```
// Delta B is not zero
// Reaction II is forbidden
// Delta S is not zero
//  Reaction III
//         pi_plus +   n ......> K_0_bar  +
   sumison_plus
// Reaction III is forbidden
// Delta S is not zero
//  Reaction IV
//         pi_plus +   n ......> pi_minus  + p
// Reaction IV is forbidden
// Delta Q is not zero
//  Reaction V
//         pi_minus +   p ......> lambda_0  + K_0
// Reaction V is allowed
// Delta Q is zero
// Delta B is zero
// Delta S is zero
//  Reaction VI
//         pi_plus +   n ......> lambda_0  + K_plus
// Reaction VI is forbidden
// Delta S is not zero
```

Scilab code Exa 18.9 Decay of sigma particle

```
// Scilab code Ex18.9 : : Page-766 (2011)
clc; clear;
h_cross = 6.62e-022;       // Reduceed planck's
   constant, MeV sec
p_width = 0.88*35;         // Partial width of the
   decay, MeV
tau = h_cross/p_width;     // Life time of sigma, sec
T_pi = 1;                  // Isospin of pi plus
   particle
T_lambda = 0;              // Isospin of lambda zero
```

```
particle
T_sigma  = T_pi+T_lambda; // Isospin of sigma particle
printf("\nThe lifetime of sigma particle = %4.2e s\nThe reaction is strong\nThe isospin of sigma particle is : %d",tau, T_sigma);

// Result
// The lifetime of sigma particle = 2.15e-023 s
// The reaction is strong
// The isospin of sigma particle is : 1
```

Scilab code Exa 18.10 Estimation of the mean life of tau plus

```
// Scilab code Exa18.10 : : Page-767 (2011)
clc; clear;
m_mew = 106;          // Mass of mew lepton, mega electron volts per square c
m_tau = 1784;         // Mass of tau lepton, mega electron volts per square c
tau_mew = 2.2e-06;    // Mean life of mew lepton, sec
R = 16/100;           // Branching factor
tau_plus = R*(m_mew/m_tau)^5*tau_mew;        // Mean life for tau plus, sec
printf("\nThe mean life for tau plus : %3.1e sec", tau_plus);

// Result
// The mean life for tau plus : 2.6e-013 sec
```

Scilab code Exa 18.13 Possible electric charge for a baryon and a meson

```
// Scilab code Exa18.13 : : Page-768(2011)
clc; clear;
function s = symbol(val)
    if val == 2 then
        s = '++';
    elseif val == 1 then
        s = '+';
    elseif val == 0 then
        s = '0';
    elseif val == -1 then
        s = '-';
    end
endfunction

B = 1;          // Baryon number
S = 0;         // Strangeness quantum number
Q = rand(1,4)     // Charge
I3 = 3/2;
printf("\nThe possible charge states are");
for i = 0:1:3
    Q = I3+(B+S)/2;
    sym = symbol(Q);
    printf(" %s", sym);
    I3 = I3 - 1;
end
printf(" respectively");

// Result
// The possible charge states are ++ + 0 -
   respectively
```

Scilab code Exa 18.15 `Branching ratio for resonant decay`

```
// Scilab code Exa18.15 : : Page-768 (2011)
clc; clear;
```

```
I_1 = 3/2;          // Isospin for delta(1232)
I_2 = 1/2;          // Isospin for delta 0
delta_ratio = sqrt((2/3)^2)/sqrt((1/3)^2);          // Branching ratio
printf("\nThe branching ratio for a resonance with I = 1/2 is %d", delta_ratio);

// Result
// The branching ratio for a resonance with I = 1/2 is 2
```

Scilab code Exa 18.16 Ratio of cross section for reactions

```
// Scilab code Exa18.16 : : Page-768 (2011)
clc; clear;
phi = 45*%pi/180;         // Phase difference
Cross_sec_ratio = 1/4*(5+4*cos(phi))/(1-cos(phi));          // Cross section ratio
printf("\nThe cross section ratio : %4.2f", Cross_sec_ratio);

// Result
```

Scilab code Exa 18.18 Root mean square radius of charge distribution

```
// Scilab code Exa18.18 : : Page-770 (2011)
clc; clear;
m_sqr = 0.71;         // For proton, (GeV/c-square)^2
R_rms = sqrt(12)/(sqrt(m_sqr)*5.1);     // Root mean square radius, femto metre
printf("\nThe root mean square radius of charge distribution: %4.2f fermi", R_rms);

```

```
// Result
// The root mean square radius of charge
   distribution: 0.81 fermi
```