Scilab Textbook Companion for
Modern Physics for Scientists and Engineers
by S. T. Thornton and A. Rex[1]

Created by
Juhi Rajput
Modern Physics
Physics
Shri Mata Vaishno Devi University
College Teacher
None
Cross-Checked by
Bhavani Jalkrish

August 14, 2014

[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: Modern Physics for Scientists and Engineers

Author: S. T. Thornton and A. Rex

Publisher: Cengage Learning, USA

Edition: 4

Year: 2013

ISBN: 978-1-133-10372-1

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

List of Figures

Chapter 2

Special Theory of Relativity

Scilab code Exa 2.2 Speed of the aircraft

```
// Scilab Code Ex2.2 : Page-34 (2013)
clc; clear;
ly = 9.46e+015;    // Distance travelled by light in an year, m
c = 3e+008;    // Speed of light, m/s
L = 4.30*ly;    // Distance of Alpha Centauri from earth, m
T0 = 16*365.25*24*60*60;    // Proper time in system K_prime, s
// As time measured on earth, T = 2*L/v = T0_prime/sqrt(1-(v/c)^2), solving for v
v = sqrt(4*L^2/(T0^2+4*L^2/c^2));    // Speed of the aircraft, m/s
gama = 1/sqrt(1-(v/c)^2);    // Relativistic factor
T = gama*T0/(365.25*24*60*60);    // Time interval as measured on Earth, y
printf("\nThe speed of the aircraft = %4.2e m/s", v);
printf("\nThe time interval as measured on earth = %4.1f y", T);

```

```
// Result
// The speed of the aircraft = 1.42e+008 m/s
// The time interval as measured on earth = 18.2 y
```

Scilab code Exa 2.3 Speed of the aircraft from the standpoint of length contraction

```
// Scilab Code Ex2.3 : Page-38 (2013)
clc; clear;
L0 = 4.30;    // Distance of Alpha Centauri from earth, ly
c = 3e+008;    // Speed of light, m/s
T = 8;    // Proper time in system K_prime, y
// As v/c = L0*sqrt(1-(v/c)^2)/(c*T) or bita = L0*sqrt(1-bita^2)/(c*T), solving for bita
bita = sqrt(L0^2/(T^2 + L0^2));    // Boost parameter
v = L0*sqrt(1-bita^2)/T;    // Speed of the aircraft, c units
printf("\nThe boost parameter = %5.3f", bita);
printf("\nThe speed of the aircraft = %5.3f c units", v);

// Result
// The boost parameter = 0.473
// The speed of the aircraft = 0.473 c units
```

Scilab code Exa 2.4 Speed of the aircraft from the standpoint of length contraction

```
// Scilab Code Ex2.4 : Page-40 (2013)
clc; clear;
```

```
c = 1;     // For simplicity assume speed of light to be unity, m/s
bita = 0.600;     // Boost parameter
gama = 1/sqrt(1-bita^2);     // Relativistic factor
u_x_prime = 0;     // Speed of the protons in spaceship frame along x-axis, m/s
u_y_prime = 0.99*c;     // Speed of the protons in spaceship frame along y-axis, m/s
u_z_prime = 0;     // Speed of the protons in spaceship frame along z-axis, m/s
v = 0.60*c;     // Speed of the spaceship w.r.t. space station, m/s
u_x = (u_x_prime + v)/(1 + v/c^2*u_x_prime);     // Speed of protons in space station frame along x-axis, m/s
u_y = u_y_prime/(gama*(1 + v/c^2*u_x_prime));     // Speed of protons in space station frame along y-axis, m/s
u_z = u_z_prime/(gama*(1 + v/c^2*u_x_prime));     // Speed of protons in space station frame along y-axis, m/s
u = sqrt(u_x^2 + u_y^2 + u_z^2);     // The speed of the protons measured by an observer in the space station, m/s
printf("\nThe speed of the protons measured by an observer in the space station = %5.3f c units", u);

// Result
// The speed of the protons measured by an observer in the space station = 0.994 c units
```

Scilab code Exa 2.6 Time loss of an atomic clock

```
// Scilab Code Ex2.6 : Page-45 (2013)
```

```
clc; clear;
c = 2.998e+008;    // Speed of light in free space, m/s
v = 7712;    // Speed of the space shuttle, m/s
bita = v/c;    // Boost parameter
T_loss = 295.02;    // Total measured loss in time, ps/sec
Add_T_loss = 35.0;    // Additional time loss for moving clock from general relativity prediction, ps/s
// From time dilation relation, T0_prime = T*sqrt(1 - bita^2), solving for (T - T0_prime)/T
Calc_T_loss = bita^2/2*1e+012;    // Expected time lost per sec by the moving clock, ps/sec
Measured_T_loss = T_loss + Add_T_loss;    // Total measured loss in time per sec as per special relativity, ps/s
percent_T_loss = (Calc_T_loss - Measured_T_loss)/Calc_T_loss*100;    // Percentage deviation of measured and the calculated time loss per sec
T = 6.05e+05;    // Total time of the seven-day mission, s
delta_T = Calc_T_loss*1e-012*T;    // The total time difference between the moving and stationary clocks during the entire shuttle flight, s
printf("\nThe expected time lost per second for the moving clock = %6.2f ps", Calc_T_loss);
printf("\nThe percentage deviation of measured and the calculated time loss per sec for moving clock = %3.1f percent", percent_T_loss);
printf("\nThe total time difference between the moving and stationary clocks during the entire shuttle flight = %3.1f ms", delta_T/1e-003);

// Result
// The expected time lost per second for the moving clock = 330.86 ps
// The percentage deviation of measured and the
```

```
   calculated time loss per sec for moving clock =
   0.3 percent
// The total time difference between the moving and
   stationary clocks during the entire shuttle
   flight = 0.2 ms
```

Scilab code Exa 2.8 Relativistic doppler effect in twin paradox

```
// Scilab Code Ex2.8 : Page-57 (2013)
clc; clear;
f0 = 1;    // For simplicity assume frequency of the
    signals sent by Frank, Hz
// Outbound trip
bita = -0.8;    // Boost parameter for receding
   frames
f = sqrt(1+bita)/sqrt(1-bita)*f0;    // The
   frequency of the signals received by Mary in
   outbound trip, Hz
printf("\nThe frequency of the signals received by
   Mary in outbound trip = f0/%d", ceil(f*9));
// Return trip
bita = +0.8;    // Boost parameter for approaching
   frames
f = sqrt(1+bita)/sqrt(1-bita)*f0;    // The
   frequency of the signals received by Mary in
   return trip, Hz
printf("\nThe frequency of the signals received by
   Mary in return trip = %df0", f);

// Result
// The frequency of the signals received by Mary in
   outbound trip = f0/3
// The frequency of the signals received by Mary in
   return trip = 3f0
```

Scilab code Exa 2.11 Accelerating electrons to produce X rays

```
// Scilab Code Ex2.11: Page-64 (2013)
clc; clear;
q = 1.6e-019;     // Charge on an electron, C
V = 25e+003;    // Accelerating potential, volt
K = q*V;    // Kinetic energy of electrons, J
m = 9.11e-031;    // Rest mass of an electron, kg
c = 3.00e+08;    // Speed of light, m/s
// From relativistic kinetic energy formula, K = (gama - 1)*m*C^2, solving for gama
gama = 1 + K/(m*c^2);    // Relativistic factor
bita = sqrt((gama^2-1)/gama^2);    // Boost parameter
u = bita*c;    // Speed of the electrons, m/s
// From non-relativistic expression, K = 1/2*m*u^2, solving for u
u_classical = sqrt(2*K/m);    // Non-relativistic speed of the electrons, m/s
u_error = (u_classical - u)/u_classical*100;    // Percentage error in speed of electrons, m/s
printf("\nThe relativistic speed of the accelerated electrons = %4.2e m/s", u);
printf("\nThe classical speed is about %d percent greater than the relativistic speed", ceil(u_error));

// Result
// The relativistic speed of the accelerated electrons = 9.04e+007 m/s
// The classical speed is about 4 percent greater than the relativistic speed
```

Scilab code Exa 2.13 Head on collision of two protons

```
// Scilab Code Ex2.13: Page-69 (2013)
clc; clear;
c = 1;    // For simplicity assume peed of light to
    be unity, m/s
K = 2.00;    // Kinetic energy of protons, GeV
E0 = 0.938;    // Rest mass of a proton, GeV
E = K + E0;    // Total energy of the proton, GeV
// From relativistic mass energy relation, E^2 = (p*
    c)^2+E0^2, solving for p
p = sqrt(E^2 - E0^2)/c;    // Momentum of the
    protons, GeV/c
// As E = gama*E0, solving for gama
gama = E/E0;    // Relativistic factor
bita = sqrt((gama^2-1)/gama^2);    // Boost
    parameter
v = bita*3.00e+08;    // Speed of 2 GeV proton, m/s
printf("\nThe energy of each initial proton = %5.3f
    GeV", E);
printf("\nThe momentum of each initial proton = %4.2
    f GeV/c", p);
printf("\nThe speed of each initial proton = %3.1e m
    /s", v);
printf("\nThe relativistic factor, gama = %4.2f",
    gama);
printf("\nThe boost parameter, beta = %5.3f", bita);

// Result
// The energy of each initial proton = 2.938 GeV
// The momentum of each initial proton = 2.78 GeV/c
// The speed of each initial proton = 2.8e+008 m/s
// The relativistic factor, gama = 3.13
// The boost parameter, beta = 0.948
```

Scilab code Exa 2.15 Minimum kinetic energy of the protons in head on collision

```
// Scilab Code Ex2.15: Page-71 (2013)
clc; clear;
E_d = 1875.6;     // Rest mass energy of the deuterium, MeV
E_pi = 139.6;     // Rest mass energy of the pion, MeV
E_p = 938.3;     // Rest mass energy of the proton, MeV
K = 1/2*(E_d + E_pi - 2*E_p);     // Minimum kinetic energy of the protons, MeV
printf("\nThe minimum kinetic energy of the protons = %2d MeV", K);

// Result
// The minimum kinetic energy of the protons = 69 MeV
```

Scilab code Exa 2.16 Binding energy of He nucleus

```
// Scilab Code Ex2.16: Page-72 (2013)
clc; clear
u = 931.5;     // Energy equivalent of 1 amu, MeV
c = 1;     // Speed of light in vacuum, m/s
m_e = 0.000549*u;     // Rest mass of an electron, MeV/c^2
m_p = 1.007276*u;     // Rest mass of a proton, MeV/c^2
m_n = 1.008665*u;     // Rest mass of a neutron, MeV/c^2
```

```
m_He = 4.002603*u;     // Rest mass of a helium, MeV/c^2
M_He = m_He - 2*m_e;     // Mass of He nucleus, MeV/c^2
E_B_He = (2*m_p + 2*m_n - M_He)*c^2;     // Binding energy of the He nucleus, MeV
printf("\nThe binding energy of the He nucleus = %4.1f MeV", E_B_He);

// Result
// The binding energy of the He nucleus = 28.3 MeV
```

Scilab code Exa 2.17 Fractional mass increase of the Na and Cl atoms

```
// Scilab Code Ex2.17: Page-72 (2013)
clc; clear
u = 931.5;     // Energy equivalent of 1 amu, MeV/u
c = 1;     // For simplicity assume speed of light in vacuum to be unity, m/s
E_B = 4.24;     // The dissociationenergy of the NaCl molecule, MeV
M = 58.44*u;     // Energy corresponding to molecular mass of NaCl, MeV
f_r = E_B/M;     // The fractional mass increase of the Na and Cl atoms
printf("\nThe fractional mass increase of the Na and Cl atoms when they are not bound together in NaCl = %4.1e", f_r/1e+006);

// Result
// The fractional mass increase of the Na and Cl atoms when they are not bound together in NaCl = 7.8e-011
```

Scilab code Exa 2.18 Kinetic energy and the mass of sigma particle

```
// Scilab Code Ex2.18: Page-72 (2013)
clc; clear
c = 1;    // For simplicity assume speed of light in vacuum to be unity, m/s
E0_n = 940;    // Rest energy of a neutron, MeV
E0_pi = 140;    // Rest energy of a pion, MeV
p_n = 4702;    // Momentum of the neutron, MeV/c
p_pi = 169;    // Momentum of the pion, MeV/c
E_n = sqrt((p_n*c)^2+E0_n^2);    // Total energy of the neutron, MeV
E_pi = sqrt((p_pi*c)^2+E0_pi^2);    // Total energy of the pion, MeV
E = E_n + E_pi;    // Total energy of the reaction, MeV
p_sigma = p_n + p_pi;    // Momentum of the sigma particle, MeV/c
E0_sigma = sqrt(E^2 - (p_sigma*c)^2);    // Rest mass energy of the sigma particle, MeV
m_sigma = E0_sigma/c^2;    // Rest mass of sigma particle, MeV/c^2;
K = E - E0_sigma;    // Kinetic energy of sigma particle, MeV
printf("\nThe rest mass of sigma particle = %4d MeV/c^2", ceil(m_sigma));
printf("\nThe kinetic energy of sigma particle = %4d MeV", ceil(K));

// Result
// The rest mass of sigma particle = 1192 MeV/c^2
// The kinetic energy of sigma particle = 3824 MeV
// The answers are given wrongly in the textbook
```

Chapter 3

The Experimental Basis of Quantum Physics

Scilab code Exa 3.1 A moving electron subjected to electric and magnetic fields

```
// Scilab Code Ex3.1: Page-87 (2013)
clc; clear
E = 1.2e+004;     // Electric field, V/m
B = 8.8e-004;     // Magnetic field, T
l = 0.05;     // Length of the deflection plates, m
v0 = E/B;     // Initial velocity of the electron, m/s
theta = 30;     // Angular deflection of the electron, degrees
q_ratio_m = E*tand(theta)/(B^2*l);     // Specific charge of the electron, C/kg
printf("\nThe initial velocity of the electron = %3.1e m/s", v0);
printf("\nThe specific charge of the electron = %3.1e C/kg", q_ratio_m);

// Result
// The initial velocity of the electron = 1.4e+007 m
```

```
/s
// The specific charge of the electron = 1.8e+011 C/
   kg
```

Scilab code Exa 3.3 Hydrogen series of spectral lines

```
// Scilab Code Ex3.3: Page-94 (2013)
clc; clear
function flag = check_visible(lambda)
    if lambda >= 400 & lambda < 700 then
        flag = 1;
        else flag = 0;
    end
endfunction
R_H = 1.0968e+007;     // Rydberg constanr, per metre
f = zeros(7);
// Lyman series
printf("\nFor Lyman series, the wavelengths are:\n")
n = 1;     // The lowest level of Lyman series
for k = 2:1:3
    lambda = 1/(R_H*(1/n^2-1/k^2))/1e-009;
    printf("k = %d, %5.1f nm", k, lambda);
    f(k) = check_visible(lambda);
    if f(k) == 1 then
            printf(" (Visible) \n");
        else
            printf(" (Ultraviolet)\n");
    end
end
if f(1)  == 1 | f(2)  == 1 | f(3) == 1 then
        printf("Some wavelengths of Lyman series
           fall in the visible region.\n")
    else
        printf("All the wavelengths of Lyman series
           fall in the UV-region.\n")
```

```
    end

// Balmer series
printf("\nFor Balmer series, the wavelengths are:\n"
   )
n = 2;     // The lowest level of Balmer series
for k = 3:1:7
    lambda = 1/(R_H*(1/n^2-1/k^2))/1e-009;
    printf("k = %d, %5.1f nm", k, lambda);
    f(k) = check_visible(lambda);
        if f(k) == 1 then
            printf(" (Visible) \n");
        else
            printf(" (Ultraviolet)\n");
    end
end

// Paschen series
printf("\nFor Paschen series, the wavelengths are:\n
   ")
n = 3;     // The lowest level of Lyman series
for k = 4:1:5
    lambda = 1/(R_H*(1/n^2-1/k^2))/1e-009;
    printf("k = %d, %5.1f nm", k, lambda);
    f(k) = check_visible(lambda);
    if f(k) == 1 then
            printf(" (Visible) \n");
        else
            printf(" (Infrared)\n");
    end
end
// For limiting member
k = %inf;
lambda = 1/(R_H*(1/n^2-1/k^2))/1e-009;
printf("k = %d, %5.1f nm", %inf, lambda);
f(6) = check_visible(lambda);
if f(6) == 1 then
            printf(" (Visible) \n");
```

```
        else
            printf(" (Infrared)\n");
    end
if f(4)  == 1 | f(5)  == 1 | f(6) == 1 then
        printf("Some wavelengths of Paschen series fall in the visible region.")
    else
        printf("All the wavelengths of Paschen series fall in the IR-region.")
    end

// Result
// For Lyman series, the wavelengths are:
// k = 2, 121.6 nm (Ultraviolet)
// k = 3, 102.6 nm (Ultraviolet)
// All the wavelengths of Lyman series fall in the UV-region.

// For Balmer series, the wavelengths are:
// k = 3, 656.5 nm (Visible)
// k = 4, 486.3 nm (Visible)
// k = 5, 434.2 nm (Visible)
// k = 6, 410.3 nm (Visible)
// k = 7, 397.1 nm (Ultraviolet)

// For Paschen series, the wavelengths are:
// k = 4, 1875.6 nm (Infrared)
// k = 5, 1282.1 nm (Infrared)
// k = Inf, 820.6 nm (Infrared)
// All the wavelengths of Paschen series fall in the IR-region.
```

Scilab code Exa 3.4 Maximum wavelength emitted from a heated furnace

```
// Scilab Code Ex3.4: Page-98 (2013)
```

```
clc; clear
T = 1600 + 273;     // Temperature of the furnace, K
b = 2.898e-003;     // Wein's constant, m-K
lambda_max = ceil(b/(T*1e-009));     // Maximum wavelength from Wein's Displacement Law, nm
printf("\nThe maximum wavelength emitted from the heated furnace = %4d nm", lambda_max);
printf("\nThis wavelength falls in the IR-region.");

// Result
// The maximum wavelength emitted from the heated furnace = 1548 nm
// This wavelength falls in the IR-region.
```

Scilab code Exa 3.5 Sun as a blackbody

```

// Scilab Code Ex3.5: Page-98 (2013)
clc; clear
lambda_max = 500e-009;     // Maximum intensity wavelength emitted by the sun, m
b = 2.898e-003;     // Wein's constant, m-K
sigma = 5.67e-008;     // Stefan's constant, W/Sq.m-K^4
r = 6.96e+008;     // Radius of the sun, m
r_E = 6.37e+006;     // Radius of the earth, m
R_E = 1.49e+011;     // Mean-earth sun distance, m
S = 4*%pi*r^2;     // Surface area of the sun, Sq.m
T_sun = b/lambda_max;     // The temperature of the sun's surface, K
R_T = sigma*T_sun^4;     // Power per unit area radiated by the sun, W/Sq.m
P_sun = R_T*S;     // The total power radiated from the sun's surface, W
F = r_E^2/(4*R_E^2);     // Fraction of sun's
```

```
    radiation received by Earth
P_Earth_received = P_sun*F;    // The radiation received by the Earth from the sun, W
U_Earth = P_Earth_received*60*60*24;    // The radiation received by the Earth from the sun in one day, J
R_Earth = P_Earth_received/(%pi*r_E^2);    // Power received by the Earth per unit of exposed area, W/Sq.m
printf("\nThe surface temperature of the sun = %4d K", ceil(T_sun));
printf("\nThe power per unit area emitted from the surface of the sun = %4.2e W/Sq.m", R_T);
printf("\nThe energy received by the Earth each day from the radiation of sun = %4.2e J", U_Earth);
printf("\nThe power received by the Earth per unit of exposed area = %4d W/Sq.m", ceil(R_Earth));

// Result
// The surface temperature of the sun = 5796 K
// The power per unit area emitted from the surface of the sun = 6.40e+007 W/Sq.m
// The energy received by the Earth each day from the radiation of sun = 1.54e+022 J
// The power received by the Earth per unit of exposed area = 1397 W/Sq.m
// The answers are given wrong in the textbook
```

Scilab code Exa 3.10 Exposure time of light to produce a photoelectron of given kinetic energy

```
// Scilab Code Ex3.10: Page-106 (2013)
clc; clear
phi = 2.36;    // Work function of sodium, eV
N_A = 6.02e+023;    // Avogadro's number
```

```
e = 1.6e-019;     // Energy equivalent of 1 eV, J
I = 1e-008;     // Intensity of incident radiation, W/Sq.m
K = 1.00;     // Kinetic energy of the ejected photoelectron, eV
rho = 0.97;     // Density of Na atoms, g/cc
M = 23;     // Gram atomic mass of Na, g/mol
n = N_A*1e+006/M*rho;     // Number of Na atoms per unit volume, atoms/metre-cube
// Assume a cubic structure, then 1/d^3 = n, solving for d
d = (1/n)^(1/3);     // Thickness of one layer of sodium atoms, m
N = n*d;     // Number of exposed atoms per Sq.m
R = I/(N*e);     // Rate of energy received by each atom, eV/s
t = (phi+K)/R;     // Time needed to absorb 3.36 eV energy
printf("\nThe exposure time of light to produce the photoelectron of %4.2f kinetic energy = %4.1f years", K, t/(60*60*24*365.25));

// Result
// The exposure time of light to produce the photoelectron of 1.00 kinetic energy = 14.7 years
```

Scilab code Exa 3.11 Photoelectric effect for lithium

```
// Scilab Code Ex3.11: Page-109 (2013)
clc; clear
phi = 2.93;     // Work function of lithium, eV
lambda = 400e-009;     // Wavelength of incident light, m
e = 1.6e-019;     // Energy equivalent of 1 eV, J
c = 2.998e+008;     // Speed of light in vacuum, m/s
```

```
h = 6.626e-034;     // Planck's constant, Js
E = h*c/(lambda*e);     // Energy of incident light, eV
V0 = E - phi;     // Stopping potential, V
printf("\nThe energy of incident photons = %4.2f eV", E);
printf("\nThe stopping potential = %4.2f V", V0);

// Result
// The energy of incident photons = 3.10 eV
// The stopping potential = 0.17 V
```

Scilab code Exa 3.12 Lithium exposed to light radiation

```

// Scilab Code Ex3.12: Page-109 (2013)
clc; clear
phi = 2.93;     // Work function of lithium, eV
c = 2.998e+008;     // Speed of light in vacuum, m/s
K = 3.00;     // Kinetic energy of photoelectron, eV
E = phi + K;     // Total energy of the incident light, eV
h = 6.626e-034;     // Planck's constant, Js
e = 1.6e-019;     // Energy equivalent of 1 eV, J
f = E*e/h;     // Frequency of incident light, Hz
lambda = c/f;     // Wavelength of the incident light, m
printf("\nThe frequency of incident light = %4.2e Hz", f);
printf("\nThe wavelength of the incident light = %4.2f nm", lambda/1e-009);

// Result
// The frequency of incident light = 1.43e+015 Hz
// The wavelength of the incident light = 209.37 nm
```

Scilab code Exa 3.13 Number of photons in the light beam of given wavelength and intensity

```
// Scilab Code Ex3.13: Page-110 (2013)
clc; clear
lambda = 350;     // Wavelength of incident light, nm
e = 1.6e-019;     // Energy equivalent of 1 eV, J
E = 1.250e+003/lambda;     // Total energy of the
    incident light, eV
I = 1e-008;     // Intensity of incident light, W/Sq.
    m
// As Intensity, I = N*E, solving for N
N = I/(E*e);     // The number of photons in the
    light beam
printf("\nThe number of photons in the light beam =
    %3.1e photons/Sq.m/s", N);

// Result
// The number of photons in the light beam = 1.8e
    +010 photons/Sq.m/s
```

Scilab code Exa 3.15 Minimum wavelength of the X rays

```
// Scilab Code Ex3.15: Page-113 (2013)
clc; clear
e = 1.6e-019;     // Energy equivalent of 1 eV, J
c = 2.998e+008;     // Speed of light in vacuum, m/s
h = 6.626e-034;     // Planck's constant, Js
V0 = 35e+003;     // Electron acceleration voltage, V
lambda_min = h*c/(e*V0);     // Duane-Hunt rule for
    wavelength, m
```

```
printf("\nThe minimum wavelength of X-rays = %4.2e m", lambda_min);

// Result
// The minimum wavelength of X-rays = 3.55e-011 m
```

Scilab code Exa 3.16 X ray scattering from the gold target

```
// Scilab Code Ex3.16: Page-116 (2013)
clc; clear
c = 2.998e+008;    // Speed of light in vacuum, m/s
h = 6.626e-034;    // Planck's constant, Js
m_e = 9.11e-031;    // Rest mass of an electron, kg
lambda = 0.050;    // Wavelength of the X-ray, nm
theta = 180;    // The angle at which the recoil electron Ke becomes the largest, degree
E_x_ray = 1.240e+003/lambda;    // Energy of the X-ray, eV
lambda_prime = lambda + (1-cosd(theta))*h/(m_e*c*1e-009);    // The largest wavelength of the scattered photon, nm
E_prime_x_ray = 1.240e+003/lambda_prime;    // Energy of the scattered photon, eV
K = (E_x_ray - E_prime_x_ray)/1e+003;    // Kinetic energy of the most energetic recoil electron, keV
if (E_prime_x_ray < E_x_ray) then
    printf("\nThe X-ray is Compton-scattered by the electron.");
else
    printf("\nThe X-ray is not Compton-scattered by the electron.");
end
printf("\nThe largest wavelength of the scattered photon = %5.3f nm", lambda_prime);
printf("\nThe kinetic energy of the most energetic
```

```
    recoil electron = %3.1f keV", K);
printf("\nThe angle at which the recoil electron
    energy is the largest = %d degrees", theta);

// Result
// The X-ray is Compton-scattered by the electron.
// The largest wavelength of the scattered photon =
    0.055 nm
// The kinetic energy of the most energetic recoil
    electron = 2.2 keV
// The angle at which the recoil electron energy is
    the largest = 180 degrees
```

Chapter 4

Structure of the Atom

Scilab code Exa 4.1 Maximum scattering angle in Geiger and Marsden experiment

```
// Scilab Code Ex4.1: Page-129 (2013)
clc; clear
m_e = 0.000549;     // Rest mass of an electron, u
m_He = 4.002603;     // Rest mass of a helium, u
M_alpha = m_He - 2*m_e;     // Mass of alpha particle, u
theta_max = 2*m_e/M_alpha;     // Maximum sctttering angle for aplha particle, rad
printf("\nThe maximum sctttering angle for aplha particle = %5.3f degrees", theta_max*180/%pi);

// Result
// The maximum sctttering angle for aplha particle = 0.016 degrees
```

Scilab code Exa 4.2 Fraction of alpha particles deflected from a gold foil

```
// Scilab Code Ex4.2: Page-137 (2013)
clc; clear
rho = 19.3;     // Density of gold, g/cc
N_A = 6.02e+023;     // Avogadro's number
N_M = 1;     // Number of atoms per molecule
M_g = 197;     // Gram atomic mass of gold, g/mol
n = rho*N_A*N_M/(M_g*1e-006);     // Number density of gold atoms, atoms/metre-cube
Z1 = 79;     // Atomic number of gold
Z2 = 2;     // Atomic number of He nucleus
t = 1e-006;     // Thickness of the gold foil, m
e = 1.602e-019;     // Charge on an electron, C
k = 9e+009;     // Coulomb constant, N-Sq.m/C^2
theta = 90;     // Angle of deflection of alpha particle, degrees
K = 7.7;     // Kinetic energy of alpha particles, MeV
f = %pi*n*t*(Z1*Z2*e^2*k/(2*1.6e-013*K))^2*cotd(theta/2)^2; // The fraction of alpha particles deflected
printf("\nThe fraction of alpha particles deflected = %1.0e", f);

// Result
// The fraction of alpha particles deflected = 4e-005
```

Scilab code Exa 4.3 Fraction of alpha particles deflected from goil foil at a given angle

```
// Scilab Code Ex4.3: Page-138 (2013)
clc; clear
rho = 19.3;     // Density of gold, g/cc
N_A = 6.02e+023;     // Avogadro's number
N_M = 1;     // Number of atoms per molecule
```

```
M_g = 197;    // Gram atomic mass of gold, g/mol
n = rho*N_A*N_M/(M_g*1e-006);    // Number density of gold atoms, atoms/metre-cube
Z1 = 79;    // Atomic number of gold
Z2 = 2;    // Atomic number of He nucleus
t = 2.1e-007;    // Thickness of the gold foil, m
e = 1.602e-019;    // Charge on an electron, C
k = 9e+009;    // Coulomb constant, N-Sq.m/C^2
r = 1e-002;    // Distance of the alpha particles from the target, m
theta = 45;    // Angle of deflection of alpha particle, degrees
K = 7.7;    // Kinetic energy of alpha particles, MeV
f = n*t*(Z1*Z2*e^2*k)^2/((r*1.6e-013*K)^2*sind(theta/2)^4*16); // The fraction of alpha particles deflected
printf("\nThe fraction of alpha particles deflected at %d degrees = %3.1e per mm square", theta, f/1e+006);

// Result
// The fraction of alpha particles deflected at 45 degrees = 3.2e-007 per mm square
```

Scilab code Exa 4.5 Size of the nucleus

```
// Scilab Code Ex4.5:Page-139 (2013)
clc; clear
Z1 = 2;    // Atomic number of He nucleus
Z2 = 13;    // Atomic number of aluminium
e = 1.602e-019;    // Charge on an electron, C
k = 9e+009;    // Coulomb constant, N-Sq.m/C^2
K = 7.7;    // Kinetic energy of alpha particles, MeV
```

```
r_min = Z1*Z2*e^2*k/(K*1.6e-013);    // Size of the aluminium nucleus, m
printf("\nThe size of the aluminium nucleus = %3.1e m", r_min);

// Result
// The size of the aluminium nucleus = 4.9e-015 m


// Result
// The maximum scttering angle for aplha particle = 0.016 degrees
```

Scilab code Exa 4.6 Nonrelativistic justification for the spped of the electron

```
// Scilab Code Ex4.6: Page-140 (2013)
clc; clear
c = 3.00e+008;    // Speed of light, m/s
r = 0.5e-010;    // Radius of the atom, m
e = 1.6e-019;    // Charge on an electron, C
m_e = 9.11e-031;    // Mass of the electron, kg
k = 9e+009;    // Coulomb constant, N-Sq.m/C^2
v = e*k^(1/2)/(m_e*r)^(1/2);    // Speed of the electron, m/s
if v < 0.01*c then
    printf("\nThe nonrelativistic treatment for calculating speed of the electron = %3.1e m/s is justified", v);
end

// Result
// The nonrelativistic treatment for calculating speed of the electron = 2.2e+006 m/s is justified
```

Scilab code Exa 4.7 Longest and shortest wavelengths observed in Paschen series for hydrogen

```
// Scilab Code Ex4.7: Page-146(2013)
clc; clear
function region = check_region(lambda)
    if lambda >= 400 & lambda < 700 then
        region = "visible";
        else region = "infrared";
    end
endfunction
n_l = 3;    // Lower electron orbit in Paschen series
n_u = [4, %inf];    // First and limiting upper orbits in Paschen series
R_inf = 1.0974e+007;    // Rydberg constant, per metre
lambda_max = 1/(R_inf*(1/n_l^2-1/n_u(1)^2)*1e-009);    // The longest wavelength in Paschen series, nm
region = check_region(lambda_max);    // Check for the region
printf("\nThe maximum wavelength is %d nm and is in the %s region", ceil(lambda_max), region);
lambda_min = 1/(R_inf*(1/n_l^2-1/n_u(2)^2)*1e-009);    // The shortest wavelength in Paschen series, nm
region = check_region(lambda_min);    // Check for the region
printf("\nThe minimum wavelength is %d nm and is also in the %s region", lambda_min, region);

// Result
// The maximum wavelength is 1875 nm and is in the
```

```
infrared region
// The minimum wavelength is 820 nm and is also in the infrared region
```

Scilab code Exa 4.8 Wavelengts of H alpha lines for three isotopes of hydrogen

```
// Scilab Code Ex4.8: Page-149(2013)
clc; clear
m_e = 0.0005486;    // Mass of an electron u
m_p = 1.007276;    // Mass of a proton, u
m_d = 2.013553;    // Mass of a deutron, u
m_t = 3.015500;    // Mass of a triton, u
R_inf = 1.0974e+007;    // Rydberg constant, per metre
R_H = 1/(1+m_e/m_p)*R_inf;    // Rydberg constant for hydrogen
R_D = 1/(1+m_e/m_d)*R_inf;    // Rydberg constant for deuterium
R_T = 1/(1+m_e/m_t)*R_inf;    // Rydberg constant for tritium
lambda_H = 1/(R_H*(1/2^2-1/3^2)*1e-009);    // Wavelength of H_alpha line for hydrogen, nm
lambda_D = 1/(R_D*(1/2^2-1/3^2)*1e-009);    // Wavelength of H_alpha line for deuterium, nm
lambda_T = 1/(R_T*(1/2^2-1/3^2)*1e-009);    // Wavelength of H_alpha line for tritium, nm
printf("\nThe wavelength of H_alpha line for hydrogen = %6.2f nm", lambda_H);
printf("\nThe wavelength of H_alpha line for deutruim = %6.2f nm", lambda_D);
printf("\nThe wavelength of H_alpha line for tritium = %6.2f nm", lambda_T);

// Result
```

```
// The wavelength of H_alpha line for hydrogen =
   656.45 nm
// The wavelength of H_alpha line for deutruim =
   656.27 nm
// The wavelength of H_alpha line for tritium =
   656.22 nm
```

Scilab code Exa 4.9 Shortest wavelength emitted by doubly positive Li ion

```
// Scilab Code Ex4.9: Page-150 (2013)
clc; clear
R = 1.0974e+007;     // Rydberg constant, per metre
Z = 3;     // Atomic number of Li
n_l = 1;     // Lower orbit of Li++ ion
n_u = %inf;     // Limiting orbit of Li++ ion
lambda = 1/(Z^2*R*(1/n_l^2-1/n_u^2)*1e-009);     //
   The shortest wavelentgh emitted by Li++ ion, nm
printf("\nThe shortest wavelentgh emitted by Li++
   ion = %4.1f nm", lambda);

// Result
// The shortest wavelentgh emitted by Li++ ion =
   10.1 nm
```

Chapter 5

Wave Properties of Matter and Quantum Mechanics I

Scilab code Exa 5.1 The wavelength of the X rays incident on rock salt

```
// Scilab Code Ex5.1 :Page-167 (2013)
clc; clear;
N_A = 6.022e+23;     // Avogdaro's No., per mole
n = 1;     // Order of diffraction
M = 58.5;     // Molecular mass of NaCl, g/mol
rho = 2.16;     // Density of rock salt, g/cc
two_theta = 20;    // Scattering angle, degree
theta = two_theta/2;     // Diffraction angle, degree
N = N_A*rho*2/(M*1e-006); // Number of atoms per unit volume, per metre cube
d = (1/N)^(1/3);  // Interplanar spacing of NaCl crystal, m
lambda = 2*d*sind(theta)/n ;     // Wavelength of X-rays using Bragg's law, m
printf("\nThe wavelength of the incident X rays = %5.3f nm", lambda/1e-009);

// Result
// The wavelength of the incident X rays = 0.098 nm
```

Scilab code Exa 5.2 de Broglie wavelength of a tennis ball and an electron

```
// Scilab Code Ex5.2 : Page-168 (2013)
clc; clear;
h = 6.63e-034;      // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
// For a moving ball
m = 0.057;      // Mass of the ball, kg
v = 25;     // Velocity of ball, m/s
p = m*v;     // Momentum of the ball, kgm/s
lambda = h/p;     // Lambda is the wavelength of ball
    , nm
printf("\nThe wavelength of ball = %3.1e m", lambda)
    ;
// For a moving electron
m = 0.511e+006;      // Rest mass of an electron, eV
K = 50;     // Kinetic energy of the electron, eV
p = sqrt(2*m*K);     // Momentum of the electron, kgm
    /s
lambda = h*c/(1.602e-019*p*1e-009);     // Wavelength
     of the electron, nm
printf("\nThe wavelength of the electron = %4.2f nm"
    , lambda);

// Result
// The wavelength of ball = 4.7e-034 m
// The wavelength of the electron = 0.17 nm
```

Scilab code Exa 5.3 de Broglie wavelength of an electron used by Davisson and Germer

```
// Scilab Code Ex5.3 : Page-173 (2013)
clc; clear;
m = 9.1e-31;    // Mass of the electron, kg
h = 6.63e-34;    // Planck's constant, Js
c = 3e+008;    // Speed of light, m/s
e = 1.6e-19;    // Energy equivalent of 1 eV, J/eV
V0 = 54;    // Potential difference between
    electrodes, V
lambda = h*c/(sqrt(2*m*c^2/e*V0)*e*1e-009);    // de
    Broglie wavelength of the electron, nm
printf("\nThe de Broglie wavelength of the electron
    used by Davisson and Germer = %5.3f nm", lambda);

// Result
// The de Broglie wavelength of the electron used by
    Davisson and Germer = 0.167 nm
```

Scilab code Exa 5.4 Wavelength of a neutron at different temperatures

```
// Scilab Code Ex5.4 : Page-174 (2013)
clc; clear;
h = 6.63e-34;    // Planck's constant, Js
c = 3e+008;    // Speed of light, m/s
e = 1.6e-19;    // Energy equivalent of 1 eV, J/eV
m = 1.67e-27;    // Mass of a neutron, kg
k = 1.38e-23;    // Boltzmann constant, J/mol/K
T = [300 77];    // Temperatures, K
lambda = h*c/(sqrt(3*m*c^2/e*k/e*T(1))*e);    // The
    wavelength of the neutron at 300 K, nm
printf("\nThe wavelength of the neutron at %d K = %5
    .3f nm", T(1), lambda/1e-09);
lambda = h*c/(sqrt(3*m*c^2/e*k/e*T(2))*e);    // The
    wavelength of the neutron at 77 K, nm
printf("\nThe wavelength of the neutron at %d K = %5
    .3f nm", T(2), lambda/1e-09);
```

```
// Result
// The wavelength of the neutron at 300 K = 0.146 nm
// The wavelength of the neutron at 77 K = 0.287 nm
```

Scilab code Exa 5.7 Distance between first two maxima in fringe pattern

```
// Scilab Code Ex5.7 :Page-184 (2013)
clc; clear;
h = 6.626e-34;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J/
    ev
d = 2000;     // Distance between slit centres, nm
K = 50e+003;     // Kinetic energy of electrons, eV
l = 350e+006;     // Distance of screen from the
    slits, nm
lambda = 1.226/sqrt(K);     // Non-relativistic value
     of de Broglie wavelength of the electrons, nm
E0 = 0.511e+006;     // Rest energy of the electron,
    J
E = K + E0;     // Total energy of the electron, J
p_c = sqrt(E^2 - E0^2);     // Relativistic mass
    energy relation, eV
lambda_r = h*c/(p_c*e*1e-009);     // Relativistic
    value of de Broglie wavelength, nm
percent_d = (lambda - lambda_r)/lambda*100;     //
    Percentage decrease in relativistic value
    relative to non-relavistic value
sin_theta = lambda_r/d;     // Bragg's law
y = l*sin_theta;     // The distance of first maximum
     from the screen, nm
printf("\nThe percentage decrease in relativistic
    value relative to non-relavistic value = %1.0f
    percent", percent_d);
```

```
printf("\nThe distance between first two maxima = %3
    .0f nm", y);

// Result
// The percentage decrease in relativistic value
    relative to non-relavistic value = 2 percent
// The distance between first two maxima = 938 nm
```

Scilab code Exa 5.8 Momentum uncertainty fo a tennis ball and an electron

```
// Scilab Code Ex5.8 : Page-187 (2013)
clc; clear;
dx = 17.5;    // The uncertainty in position, m
h = 1.05e-034;    // Reduced Planck's constant, Js
dp_x = h/(2*dx);    // The uncertainty in momentum,
    kgm/s
printf("\nThe unecrtainty in momentum of the ball =
    %1.0e kg-m/s", dp_x);
dx = 0.529e-010;    // The uncertainty in position,
    m
dp_x = h/(2*dx);    // The uncertainty in momentum,
    kgm/s
printf("\nThe uncertainty in momentum of the
    electron = %1.0e kg-m/s", dp_x);

// Result
// The unecrtainty in momentum of the ball = 3e-036
    kg-m/s
// The uncertainty in momentum of the electron = 1e
    -024 kg-m/s
```

Scilab code Exa 5.9 Minimum kinetic energy of an electron in hydrogen atom

```
// Scilab Code Ex5.9 : Page-188 (2013)
clc; clear;
a_0 = 5.29e-11;     // Radius of H-atom, m
l = 2*a_0;     // Length, m
h = 6.63e-34;     // Planck's constant, Js
m = 9.1e-31;     // Mass of electron, kg
K_min = h^2/(8*(%pi)^2*m*l^2);     // Minimum kinetic
    energy possesed, J
printf("\nThe minimum kinetic energy of the electron
    = %3.1f eV", K_min/1.6e-19);

// Result
// The minimum kinetic energy of the electron = 3.4
   eV
```

Scilab code Exa 5.10 Minimum kinetic energy of electron localized within a typical nuclear radius

```
// Scilab Code 5.10: : Page-190 (2013)
clc; clear;
dx = 6e-015;     // The uncertainty in position of
   the electron, m
h_bar = 1.054e-034;     // PReduced Planck's constant
   , Js
e = 1.602e-019;     // Energy equivalnet of 1 eV, J/
   eV
c = 3e+008;     // Speed of light, m/s
E0 = 0.511e+006;     // Rest mass energy of the
   electron, J
dp = h_bar*c/(2*dx*e);     // Minimum electron
   momentum, eV/c
p = dp;     // Momentum of the electron at least
```

```
    equal to the uncertainty in momentum, eV/c
E = sqrt(p^2+E0^2)/1e+006;     // Relativistic energy
     of the electron , MeV
K = E - E0/1e+006;     // Minimum kinetic energy of
    the electron , MeV
printf("\nThe minimum kinetic energy of the electron
     = %4.1f MeV", K);

// Result
// The minimum kinetic energy of  the electron =
    15.9 MeV
```

Scilab code Exa 5.12 Energy width of excited state of atom and uncertainty ratio of frequency of emitted photon

```
// Scilab Code Ex5.12 : Page-190 (2014)
clc; clear;
c = 3e+8;     // Speed of light , m/s
dt = 1e-08;     // Relaxation time of atom, s
h = 6.6e-34;     // Planck's constant , Js
dE = h/(4*%pi*dt);     // Energy width of excited
    state of atom, J
lambda = 300e-009;     // Wavelegth of emitted photon
    , m
f = c/lambda;     // Frequency of emitted photon, per
     sec
printf("\nThe energy width of excited state of the
    atom = %3.1e eV", dE/1.6e-019);
df = dE/h;     // Uncertainty in frequency, per sec
printf("\nThe uncertainty ratio of the frequency =
    %1.0e",  df/f);

// Result
// The energy width of excited state of the atom =
    3.3e-008 eV
```

```
// The uncertainty ratio of the frequency = 8e-009
```

Scilab code Exa 5.13 Energy at diffrent levels

```
// Scilab Code Ex5.13 : Page-195 (2014)
clc; clear;
n = [1 2 3];     // First three energy levels
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
c = 3e008;    // Speed of light, m/s
h = 6.63e-034;     // Planck's constant, Js
m = 9.1e-031;     // Mass of the proton, kg
l = 0.1;     // Length of one-dimensional box, nm
E_n = n^2*(h*c/(e*1e-009))^2/(8*m*c^2/e*l^2);     //
    Energy of nth level, eV
printf("\nThe first three energy level are:\nE1 = %2
    .0f eV, E2 = %3.0f eV and E3 = %3.0f eV", E_n(1),
     E_n(2), E_n(3));

// Result
// The first three energy level are:
// E1 = 38 eV, E2 = 151 eV and E3 = 340 eV
```

Chapter 6

Quantum Mechanics II

Scilab code Exa 6.4 Probabilities of a particle in the given regions

```
// Scilab Code Ex6.4 : Page-205 (2014)
clc; clear;
alpha = 1;    // For simplicity assume alpha to be
    unity
P = integrate('sqrt(alpha)*exp(-2*alpha*x)', 'x', 0,
     1/alpha);    // Probability that the particle
    lies between 0 and 1/alpha
printf("\nThe probability that the particle lies
    between 0 and 1/alpha = %5.3f", P);
P = integrate('sqrt(alpha)*exp(-2*alpha*x)', 'x', 1/
    alpha, 2/alpha);    // Probability that the
    particle lies between 1/alpha and 2/alpha
printf("\nThe probability that the particle lies
    between 1/alpha and 2/alpha = %5.3f", P);

// Result
// The probability that the particle lies between 0
    and 1/alpha = 0.432
// The probability that the particle lies between 1/
    alpha and 2/alpha = 0.059
```

Scilab code Exa 6.9 Transition energy for a proton confined to a nucleus

```
// Scilab Code Ex6.9: Page-215 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
c = 3.00e+008;    // Speed of light, m/s
e = 1.602e-019;    // Energy equivalent of 1 eV, J
m = 938.3e+006;    // Energy equivalent of proton mass, eV
L = 1e-005;    // Diameter of the nucleus, nm
E1 = %pi^2*(h_bar*c/(e*1e-009))^2/(2*L^2*m*1e+006);    // Energy of the ground state of proton, MeV
E2 = 4*E1;    // Energy of first excited state of proton, MeV
delta_E = E2 -E1;    // Transition energy of the proton inside the nucleus, MeV
printf("\nThe transition energy of the proton inside the nucleus = %1d MeV", delta_E);

// Result
// The transition energy of the proton inside the nucleus = 6 MeV
```

Scilab code Exa 6.14 Fraction of electrons tunneling through a barrier

```
// Scilab Code Ex6.14: Page-229 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
```

```
c = 3.00e+008;    // Speed of light, m/s
e = 1.602e-019;    // Energy equivalent of 1 eV, J
m = 0.511e+006;    // Energy equivalent of electron rest mass, eV
V0 = 10;    // Height of potential barrier, eV
E = 5;    // Energy of the incident electrons, eV
L = 0.8e-009;    // Width of the potential barrier, m
k = sqrt(2*m*(V0 - E))*e/(h_bar*c);    // Schrodinger's constant, per m
T = (1 + V0^2*sinh(k*L)^2/(4*E*(V0 - E)))^(-1);    // Transmission electron probability
printf("\nThe fraction of electrons tunneled through the barrier = %3.1e", T);

// Result
// The fraction of electrons tunneled through the barrier = 4.4e-008
```

Scilab code Exa 6.15 Probability of electron tunneling through the barrier

```
// Scilab Code Ex6.15: Page-229 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
c = 3.00e+008;    // Speed of light, m/s
e = 1.602e-019;    // Energy equivalent of 1 eV, J
m = 0.511e+006;    // Energy equivalent of electron rest mass, eV
V0 = 10;    // Height of potential barrier, eV
Sum_M = 0;
i = 1;
for E = 5:-1:1    // Range of energies of the
```

```
incident electrons, eV
    M = 16*E/V0*(1-E/V0);    // All the factors multiplying the exponential term
    Sum_M = Sum_M + M;    // Accumulator
    i = i + 1;
end
E = 5;    // Given energy of the incident electrons, eV
M = int(Sum_M/i);    // Avearge value of M
L = 0.8e-009;    // Width of the potential barrier, m
k = sqrt(2*m*(V0 - E))*e/(h_bar*c);    // Schrodinger's constant, per m
T = M*exp(-2*k*L);    // Transmission electron probability
printf("\nThe fraction of electrons tunneled through the barrier = %3.1e", T);

// Result
// The fraction of electrons tunneled through the barrier = 2.2e-008
```

Scilab code Exa 6.16 A particle penetrating through a potential step

```
// Scilab Code Ex6.16: Page-230 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
c = 3.00e+008;    // Speed of light, m/s
e = 1.602e-019;    // Energy equivalent of 1 eV, J
m = 0.511e+006;    // Energy equivalent of electron rest mass, eV
V0 = 10;    // Height of potential barrier, eV
E = 5;    // Given energy of the incident electrons,
```

```
eV
l = h_bar*c/(2*sqrt(2*m*(V0 - E))*1e-009*e);    // Penetration distance into the barrier when the probability of the particle penetration drops to 1/e, nm
printf("\nThe penetration distance for a %d eV electron approaching a step barrier of %d eV = %5.3f nm", E, V0, l);

// Result
// The penetration distance for a 5 eV electron approaching a step barrier of 10 eV = 0.044 nm
```

Scilab code Exa 6.17 An alpha particle tunnelling through a nucleus

```

// Scilab Code Ex6.17: Page-234 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
c = 3.00e+008;    // Speed of light, m/s
e = 1.602e-019;    // Charge on an electron, C
k = 9e+009;    // Coulomb constant, N-Sq.m./C^2
m = 3727;    // Energy equivalent of alpha particle rest mass, MeV
E = 5;    // Given energy of the incident electrons, eV
Z1 = 2;    // Atomic number of an alpha particle
Z2 = 92;    // Atomic number of the U-238 nucleus
r_N = 7e-015;    // Nuclear radius, m
K = 4.2;    // Kinetic energy of alpha particle, MeV
V_C = Z1*Z2*e^2*k/(r_N*e*1e+006);    // Coulomb Potential, MeV
r_prime = V_C*r_N/(K*1e-015);    // Distance through
```

```
   which the alpha particle must tunnel, fm
kapa = sqrt(2*m*(V_C-E))*e/(h_bar*c*1e-006);    // Schronginger's Constant, per m
L = r_prime - r_N/1e-015;    // Barrier width, fm
T = 16*(K/V_C)*(1-K/V_C)*exp(-2*kapa*L*1e-015);    // Tunnelling probability of alpha particle
V_C_new = V_C/2;    // Potential equal to half the Coulomb potential, MeV
L = L/2;    // Width equal to half the barrier width, fm
kapa = sqrt(2*m*(V_C_new-E))*e/(h_bar*c*1e-006);    // Schronginger's Constant, per m
T_a = 16*(K/V_C_new)*(1-K/V_C_new)*exp(-2*kapa*L*1e-015);    // Approximated tunnelling probability of alpha particle
v = sqrt(2*K/m)*c;    // Speed of the alpha particle, m/s
t = r_N/v;    // Time taken by alpha particle to cross the U-238 nucleus, s
printf("\nThe barrier height = %2d MeV", ceil(V_C));
printf("\nThe distance that alpha particle must tunnel = %2d fm", r_prime);
printf("\nThe tunneling potential assuming square-top potential = %1.0e", T);
printf("\nThe approximated tunneling potential = %3.1e", T_a);
printf("\nThe speed of the alpha particle = %3.1e m/s", v);
printf("\nThe time taken by alpha particle to cross the U-238 nucleus = %1.0e s", t);

// Result
// The barrier height = 38 MeV
// The distance that alpha particle must tunnel = 63 fm
// The answers are given wrongly in the textbook
// The tunneling potential assuming square-top potential = 6e-123
```

```
// The approximated tunneling potential = 3.8e-040
// The speed of the alpha particle = 1.4e+007 m/s
// The time taken by alpha particle to cross the U
    -238 nucleus = 5e-022 s
// Some answers are given wrong in the textbook for
    this problem
```

Chapter 7

The Hydrogen Atom

Scilab code Exa 7.2 Normalization of hydrogen wave function

```
// Scilab Code Ex7.2: Page-248 (2014)
clc; clear;
a0 = 1;    // For simplicity assume Bohr radius to
    be unity, m
NE = 1/(64*%pi*a0^5)*integrate('r^4*exp(-r/a0)', 'r'
    , 0, 15)*integrate('sin(t)^3', 't', 0, %pi)*
    integrate('p^0', 'p', 0, 2*%pi);
if round(NE) == 1 then
    printf("\nThe hydrogen wave function <211| is
        normalized");
else
    printf("\nThe hydrogen wave function <211| is
        not normalized");
end

// Result
// The hydrogen wave function <211| is normalized
```

Scilab code Exa 7.4 Degeneracy of M level in hydrogen atom

```
// Scilab Code Ex7.4: Page-252 (2014)
clc; clear;
n = 3;    // Principal quantum number
Total = 0;
printf("\nn     l     m_l                    2(l + 1)");
printf("\n-------------------------------------------");
for l = 0:1:n-1
    printf("\n%d", n);
    printf("     %d     ", l);
        if l > 0 then
            count = 0;
            for m_l = -l:1:l
               printf("%2d ", m_l);
               count = count + 1;
            end
            if  l == 1 then
                printf("       ");
            else
                printf("");
            end
        else
               m_l = 0;
               count = 0;
               printf("%2d                ", m_l);
               count = count + 1;
        end
        printf("     %d", count);
        Total = Total + count;
end
printf("\n                          Total = %d", Total);

// Result
// n     l     m_l                    2(l + 1)
// -------------------------------------------
// 3     0      0                        1
// 3     1     -1   0   1                3
// 3     2     -2  -1   0   1   2        5
//                          Total = 9
```

Scilab code Exa 7.7 Energy difference between components of p states of atomic hydrogen placed in an external field

```
// Scilab Code Ex7.7: Page-255 (2014)
clc; clear;
e = 1.602e-019;     // Charge on an electron, C
h = 6.62e-034;     // Planck's constant, Js
h_bar = h/(2*%pi);     // Reduced Planck's constant, Js
m = 9.11e-031;     // Electron mass, kg
B = 2.00;     // External magnetic field, T
m_l1 = 0;     // Lower orbial magnetic quantum number
m_l2 = 1;     // Upper orbial magnetic quantum number
delta_m_l = m_l2 - m_l1;     // Change in m_l
mu_B = e*h_bar/(2*m);     // Bohr's magneton, J/T
delta_E = mu_B*B*delta_m_l/e;     // Energy difference between components of p states of atomic hydrogen placed in the external field, eV
printf("\nThe value of Bohr magneton = %4.2e J/T", mu_B);
printf("\nThe energy difference between components of p states of atomic hydrogen placed in the external field = %4.2e eV", delta_E);

// Result
// The value of Bohr magneton = 9.26e-024 J/T
// The energy difference between components of p states of atomic hydrogen placed in the external field = 1.16e-004 eV
```

Scilab code Exa 7.8 Separation of the atomic beam

```
// Scilab Code Ex7.8: Page-257 (2014)
clc; clear;
m = 1.67e-027;     // Mass of the proton, kg
k = 1.38e-023;     // Boltzmann constant, J/K
T = 663;     // Temperature of the discharge tube, K
v_x = sqrt(3*k*T/m);     // Average speed of the
    hydrogen atom
mu_z = 9.27e-024;     // Bohr's magneton, J/T
B_grad = 1240;     // Magnetic field gradient, T/m
delta_x = 0.03;     // Length of the homogeneous
    magnetic field, m
d = 1/(2*m)*(mu_z*B_grad)*(delta_x/v_x)^2;     //
    Separation of the atomic beam, m
printf("\nThe separation of the atomic beam = %4.2f
    mm", d/1e-003);

// Result
// The separation of the atomic beam = 0.19 mm
```

Scilab code Exa 7.9 Number of distinct states for the 4d level of atomic hydrogen

```
// Scilab Code Ex7.9: Page-259 (2014)
clc; clear;
n = 4;     // Principal quantum number
l = 2;     // For 4d-state
printf("\nn     l          m_l       m_s");
printf("\n-----------------------------------");
     count = 0;
     for m_l = -l:1:l
         if (m_l == 0) then
             printf("\n%d", n);
             printf("     %d     ", l);
             printf("      %2d", m_l);
             printf("     +1/2, -1/2");
```

```
            else
                printf("\n                    %2d", m_l);
                printf("    +1/2, -1/2");
            end
            count = count + 2;
        end
printf("\nTotal No. of different states for 4d level of atomic hydrogen = %d", count);

// Result
// n    l          m_l       m_s
// ----------------------------------
//                  -2     +1/2, -1/2
//                  -1     +1/2, -1/2
// 4    2            0     +1/2, -1/2
//                   1     +1/2, -1/2
//                   2     +1/2, -1/2
// Total No. of different states for 4d level of atomic hydrogen = 10
```

Scilab code Exa 7.10 Energy of allowed transitions for the hydrogen atom

```
// Scilab Code Ex7.10: Page-263 (2014)
clc; clear;
function flag = check_allowance(dn, dl, dml, dms)
    if (dl == -1 | dl == 1 | dml == -1 | dml == 0 | dml == 1 | dms == -1 | dms == 0 | dms == 1) & dl <> 0 then
        flag = 1;
    else
        flag = 0;
    end
endfunction
state = [2 0 0 1/2; 3 1 1 1/2; 2 0 0 1/2; 3 0 0 1/2; 4 2 -1 -1/2; 2 1 0 1/2];
```

```
for i = 1:2:5
    flag = 0;
    d_n = state(i,1) - state(i+1,1);
    d_l = state(i,2) - state(i+1,2);
    d_m_l = state(i,3) - state(i+1,3);
    d_m_s = state(i,4) - state(i+1,4);
    flag = check_allowance(d_n, d_l, d_m_l, d_m_s);
    if flag == 1 then
        printf("\n\nThe transition (%d,%d,%d,1/%d)
           —> (%d,%d,%d,1/%d) is allowed", state(i
           ,1), state(i,2), state(i,3), state(i,4)
           *4, state(i+1,1), state(i+1,2), state(i
           +1,3), state(i+1,4)*4);
        delta_E = -13.6*(1/state(i+1)^2-1/state(i)
           ^2);
        printf("\nThe energy of this transition is
           %4.2f eV", delta_E);
    else
        printf("\n\nThe transition (%d,%d,%d, %d)—>
            (%d,%d,%d,%d) is not allowed", state(i
           ,1), state(i,2), state(i,3), state(i,4),
           state(i+1,1), state(i+1,2), state(i+1,3),
            state(i+1,4));
    end
end

// Result
// The transition (2,0,0,1/2) —> (3,1,1,1/2) is
   allowed
// The energy of this transition is 1.89 eV

// The transition (2,0,0, 0)—> (3,0,0,0) is not
   allowed

// The transition (4,2,-1,1/-2) —> (2,1,0,1/2) is
   allowed
// The energy of this transition is -2.55 eV
```

Scilab code Exa 7.13 Probability of the electron in the 1s state of the hydrogen atom

```
// Scilab Code Ex7.13: Page-265 (2014)
clc; clear;
a0 = 1;     // For simplicity assume bohr radius to
    be unity
P = integrate('4/a0^3*exp(-2*r/a0)*r^2', 'r', a0,
    10);
printf("\nThe probability of the electron in the 1s
    state of the hydrogen atom = %4.2f", P);

// Result
// The probability of the electron in the 1s state
    of the hydrogen atom = 0.68
```

Chapter 8

Atomic Physics

Scilab code Exa 8.3 Splitting of 3p subshell of sodium

```
// Scilab Code Ex8.3: Page-285 (2014)
clc; clear;
delta_E = 2e-003;    // Energy difference for the 3p
    subshell of sodium, eV
h = 6.62e-034;    // Planck's constant, Js
e = 1.602e-019;    // Energy equivalent of 1 eV, J
c = 3.00e+008;    // Speed of light in vacuum, m/s
lambda = 589.3;    // Wavelength of spectral line,
    nm
// As delta_E = h*c/lambda^2*delta_lambda, solving
    for delta_lambda
delta_lambda = lambda^2*e/(h*c*1e+009)*delta_E;
    // Splitting of 3p subshell of sodium, nm
printf("\nThe splitting of 3p subshell of sodium =
    %3.1f nm", delta_lambda);

// Result
// The splitting of 3p subshell of sodium = 0.6 nm
```

Scilab code Exa 8.5 LS coupling of two electrons in an atom

```
// Scilab Code Ex8.5: Page-289 (2014)
clc; clear;
l1 = 1;     // Orbital angular momentum quantum
    number for first electron
l2 = 2;     // Orbital angular momentum quantum
    number for second electron
s1 = 1/2;     // Spin angular momentum quantum number
     for first electron
s2 = 1/2;     // Spin angular momentum quantum number
     for second electron
temp_j = zeros(15);
cnt = 1;
printf("\nThe all possibe values of the total
    angular momentum quantum number of J are:\n");
for L = abs(l1 - l2):1:abs(l1 + l2)
    for S = abs(s1 - s2):1:abs(s1 + s2)
        for j = abs(L - S):1:abs(L + S)
            temp_j(cnt) = j;
            cnt = cnt + 1;
        end
    end
end
J = -1;
temp_J = gsort(temp_j, 'g', 'i');
for i = 1:1:cnt-1
    if temp_J(i) > J  then
       J = temp_J(i);
       printf("%d  ", J);
    end
end

// Result
// The all possibe values of the total angular
    momentum quantum number of J are:
// 0  1  2  3  4
```

Scilab code Exa 8.8 Internal magnetic field causing spin orbit splitting

```
// Scilab Code Ex8.8: Page-291 (2014)
clc; clear;
delta_E = 0.0021;    // Energy difference for the 3p
     subshell of sodium, eV
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant,
    Js
e = 1.602e-019;    // Energy equivalent of 1 eV, J
m = 9.11e-031;    // Rest of an an electron, kg
g_s = 2;    // Gyromagnetic ratio due to spin
    splitting
// As delta_E = g_s*e*h_bar/(2*m)*B, solving for B
B = m*delta_E/h_bar;    // Internal magnetic field
    causing the LS splitting, T
printf("\nThe internal magnetic field causing the LS
     splitting = %2d T", B);

// Result
// The internal magnetic field causing the LS
    splitting = 18 T
```

Chapter 9

Statistical Physics

Scilab code Exa 9.1 Mean translational kinetic energy of gas molecules at room temperature

```
// Scilab Code Ex9.1: Page-303 (2014)
clc; clear;
k = 1.38e-023;    // Boltzmann constant, J/K
N_A = 6.023e+023;    // Avogadro's number
T = 293;    // Room temperature, K
e = 1.6e-019;    // Energy equivalent of 1 eV, J
// For a single molecule
K_bar_single = 3/2*k*T/e;    // Mean translational
    kinetic energy of a single gas molecule, J
// For a 1 mole of molecules
K_bar_mole = K_bar_single*N_A*e;    // Mean
    translational kinetic energy of 1 mole of gas
    molecules, J
printf("\nThe mean translational kinetic energy of
    the single idela gas molecule = %5.3f eV",
    K_bar_single);
printf("\nThe mean translational kinetic energy of
    the one mole of ideal gas molecules = %4d J",
    ceil(K_bar_mole));

```

```
// Result
// The mean translational kinetic energy of the single idela gas molecule = 0.038 eV
// The mean translational kinetic energy of the one mole of ideal gas molecules = 3654 J
```

Scilab code Exa 9.3 Mean molecular speed in light gas hydrogen and heavy radon gas

```
// Scilab Code Ex9.3: Page-310 (2014)
clc; clear;
k = 1.38e-023;     // Boltzmann constant, J/K
u = 1.67e-027;     // Mass equivalent of one atomic mass unit, kg
T = 293;    // Room temperature, K
m_H = 1.008*u;     // Gram atomic mass of hydrogen, kg
m = 2*m_H;     // Gram molecular mass of hydrogen molecule, kg
v_bar = 4/sqrt(2*%pi)*sqrt(k*T/m);     // Mean molecular speed in the light gas hydrogen, m/s
printf("\nThe mean molecular speed in the light gas hydrogen = %4d m/s", ceil(v_bar));
m = 222*u;     // Gram atomic mass of Radon, kg
v_bar = 4/sqrt(2*%pi)*sqrt(k*T/m);     // Mean molecular speed in the heavy radon gas, m/s
printf("\nThe mean molecular speed in the heavy radon gas = %3d m/s", ceil(v_bar));

// Result
// The mean molecular speed in the light gas hydrogen = 1749 m/s
// The mean molecular speed in the heavy radon gas = 167 m/s
```

Scilab code Exa 9.4 Fraction of molecules in an ideal gas having speed near to the most probable speed

```
// Scilab Code Ex9.4: Page-310 (2014)
clc; clear;
m = 1;     // For simplicity assume mass of gas
    molecule to be unity, kg
k = 1.38e-023;     // Boltzmann constant, J/K
T = 293;     // Room temperature, K
bita = k*T;     // Energy associated with three
    degrees of freedom, J
v_mps = sqrt(2/(bita*m));     // For simplcity assume
     most probable speed to be unity, m/s
C = (bita*m/(2*%pi))^(3/2);     // Constant in the
    distribution function
P = integrate('4*%pi*C*exp(-1/2*bita*m*v^2)*v^2', 'v
    ', 0.99*v_mps, 1.01*v_mps);
printf("\nThe fraction of molecules in an ideal gas
    in equilibrium which have speeds within 1 percent
     above and below the most probable speed = %5.3f"
    , P);

// Result
// The fraction of molecules in an ideal gas in
    equilibrium which have speeds within 1 percent
    above and below the most probable speed = 0.017
```

Scilab code Exa 9.6 Relative number of atoms in the ground and first excited states in atomix hydrogen

```
// Scilab Code Ex9.6: Page-315 (2014)
clc; clear;
```

```
k = 1.38e-023;    // Boltzmann constant, J/K
T = [293 5000 1e+006];    // Room temperature, temperature at the surface of the star and temperature at the star interior respectively, K
e = 1.6e-019;    // Energy equivalent of 1 eV, J
g_E1 = 2;    // Possible configuration of the electrons in ground state of H-atom
g_E2 = 8;    // Possible configuration of the electrons in the first excited state of H-atom
E1 = -13.6;    // Energy of the ground state, eV
E2 = -3.4;    // Energy of the first excited state state, eV
n_ratio = zeros(3);
for i = 1:1:3
    n_ratio(i) = g_E2/g_E1*exp(1/(k*T(i))*(E1 - E2)*e);
    printf("\nFor T = %4.2e K, n_E2/n_E1 = %4.2e", T(i), n_ratio(i));
end


// Result
// For T = 2.93e+002 K, n_E2/n_E1 = 2.05e-175
// For T = 5.00e+003 K, n_E2/n_E1 = 2.14e-010
// For T = 1.00e+006 K, n_E2/n_E1 = 3.55e+000
```

Scilab code Exa 9.7 Fermi energy and Fermi temperature for copper

```
// Scilab Code Ex9.7: Page-320 (2014)
clc; clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, J
n = 8.47e+028;    // Number density of conduction electrons in copper, per metre cube
k = 1.38e-023;    // Boltzmann constant, J/K
h = 6.626e-034;    // Planck's constant, Js
```

```
m = 9.11e-031;     // Mass of an electron, kg
E_F = h^2/(8*m*e)*(3*n/%pi)^(2/3);     // Fermi energy for copper, eV
T_F = E_F*e/k;     // Fermi temperature for copper, K
printf("\nThe Fermi energy for copper = %4.2f eV", E_F);
printf("\nThe Fermi temperature for copper = %4.2e K", T_F);

// Result
//  The Fermi energy for copper = 7.04 eV
// The Fermi temperature for copper = 8.16e+004 K
```

Scilab code Exa 9.8 Electronic contribution to the molar heat capacity of metals

```
// Scilab Code Ex9.8: Page-322 (2014)
clc; clear;
R = 1;     // For simplicity assume the molar gas constant to be unity, J/mol/K
T = 293;     // Room temperature, K
T_F = 8.16e+004;     // The Fermi temperature for copper
C_V = %pi^2*T/(2*T_F)*R;     // Electronic contribution to the molar heat capacity for copper, J/mol/K
printf("\nThe electronic contribution to the molar heat capacity for copper = %6.4fR", C_V);
T_F = 6.38e+004;     // The Fermi temperature for silver
C_V = %pi^2*T/(2*T_F)*R;     // Electronic contribution to the molar heat capacity for silver, J/mol/K
printf("\nThe electronic contribution to the molar heat capacity for silver = %6.4fR", C_V);
```

```
// Result
// The electronic contribution to the molar heat
   capacity for copper = 0.0177R
// The electronic contribution to the molar heat
   capacity for silver = 0.0227R
```

Chapter 10

Molecules Lasers and Solids

Scilab code Exa 10.1 Energy of lowest rotational state of nitrogen gas

```
// Scilab Code Ex10.1 : Page-342 (2014)
clc; clear;
m = 2.33e-026;    // Mass of a nitrogen atom, kg
R = 1.1e-010;    // Interatomic separtion between two nitrogen atoms, m
h = 6.626e-034;    // Planck's constant, Js
e = 1.6e-019;    // Energy equivalent of 1 eV, J
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
I = m*R^2/2;    // Momemt of rotational inertia of nitrogen gas molecule, kg-Sq.m
l = 1;    // Rotational angular momentum quantum number
E_rot = h_bar^2*l*(l+1)/(2*I);    // The energy for lowest rotational state of the nitrogen molecule, J
printf("\nThe energy for lowest rotational state of the nitrogen molecule = %4.2e eV", E_rot/e);

// Result
// The energy for lowest rotational state of the
```

```
nitrogen molecule = 4.93e-004 eV
```

Scilab code Exa 10.2 Vibrational energy levels of HCl molecule

```

// Scilab Code Ex10.2 : Page-343 (2014)
clc; clear;
h = 6.626e-034;     // Planck's constant, Js
e = 1.6e-019;     // Energy equivalent of 1 eV, J
h_bar = h/(2*%pi);     // Reduced Planck's constant, Js
k = 1.38e-023;     // Boltzmann constant, J/K
u = 1.67e-027;     // Mass equivalent of 1 amu, kg
m1 = 34.97*u;     // Atomic mass of chlorine atom, kg
m2 = 1.008*u;     // Atomic mass of hydrogen atom, kg
mu = m1*m2/(m1 + m2);     // Reduced mass of the HCl system, kg
delta_E = 0.36;     // Spacing between vibrational energy levels of the HCl molecule, eV
omega = delta_E*e/h_bar;     // Angular frequency of vibration, rad/s
kapa = mu*omega^2;     // Effective force constant for HCl molecule, N/m
T = delta_E*e/k;     // Classical temperature associated with the rotational energy spacing, K

printf("\nThe effective force constant for HCl molecule = %3d N/m", ceil(kapa));
printf("\nThe classical temperature associated with the rotational energy spacing = %4d K", ceil(T));

// Result
// The effective force constant for HCl molecule = 489 N/m
// The classical temperature associated with the
```

```
    rotational energy spacing = 4174 K
// The answers are given wrong in the textbook
```

Scilab code Exa 10.4 Range parameter for NaCl

```
// Scilab Code Ex10.4 : Page-358 (2014)
clc; clear;
e = 1.602e-019;     // Charge on an electron , C
N_A = 6.023e+023;     // Avogadro's number
alpha = 1.7476;     // Madelung constant
E = -764.4e+003;     // Dissociation energy of NaCl
    molecule , J/mol
V = E/N_A;     // Repulsive potential energy , J
k = 8.988e+009;     // Coulomb's constant , N-Sq.m/C^2
r0 = 0.282e-009;     // Equilibrium separation for
    nearest neighbour in NaCl , m
rho = r0*(1+r0*V/(k*alpha*e^2));     // Range
    parameter for NaCl , nm
printf("\nThe range parameter for NaCl = %6.4f nm",
    rho/1e-009);

// Result
// The range parameter for NaCl = 0.0316 nm
```

Scilab code Exa 10.5 Induced diamagnetism in an atom

```
// Scilab Code Ex10.5 :Page-364 (2014)
clc; clear;
e = 1.602e-019;     // Charge on an electron , C
r = 5.29e-011;     // Orbital radius of electron
    equal to Bohr radius , m
B = 2.0;     // Applied magnetic field , T
m = 9.11e-031;     // Mass of an electron , kg
```

```
delta_mu = e^2*r^2*B/(4*m);     // Induced diamagnetic moment in the atom, J/T
printf("\nThe induced diamagnetic moment in the atom = %3.1e J/T", delta_mu);

// Result
// The induced diamagnetic moment in the atom = 3.9e-029 J/T
```

Scilab code Exa 10.6 Paramagnetism in a typical material

```
// Scilab Code Ex10.6 : Page-366 (2014)
clc; clear;
mu_B = 9.27e-024;     // Bohr's magneton, J/T
B = 0.50;     // Applied magnetic field, T
k = 1.38e-023;     // Boltzmann constant, J/K
T = 10*mu_B*B/k;     // Temperature at which mu*B = 0.1k*T, K
b_muB = mu_B*B/(k*T);
ratio = b_muB/tanh(b_muB);     // Ratio of b_muB and tanh(b_muB)
if (ratio - 1) < 0.01 then
    printf("\nThe value of T = %4.2f K is suiable as a classical temperature.", T);
else
    printf("\nThe value of T = %4.2f K is not suiable as a classical temperature.", T);
end
// For higher temperature
T = 100;     // Given temperature
b_muB = mu_B*B/(k*T);
ratio = b_muB/tanh(b_muB);     // Ratio of b_muB and tanh(b_muB)
if (ratio - 1) < 0.001 then
    printf("\nAt the value of T = %4.2f K, the
```

```
        approximation is an excellent one.", T);
else
    printf("\nAt the value of T = %4.2f K, the
        approximation is not an excellent.", T);
end

// Result
// The value of T = 3.36 K is suiable as a classical
    temperature.
// At the value of T = 100.00 K, the approximation
    is an excellent one.
```

Scilab code Exa 10.7 Superconductivity in niobium

```
// Scilab Code Ex10.7 : Page-374 (2014)
clc; clear;
k = 1.38e-023;    // Boltzmann constant, J/K
e = 1.602e-019;    // Energy equivalent of 1 eV, J
h = 6.62e-034;    // Planck's constant, Js
c = 3.00e+008;    // Speed of light, m/s
T_c = 9.25;    // Critical temperature for niobium,
    K
E_g = 3.54*k*T_c;    // Energy gap for niobium from
    BCS theory, J
lambda = h*c/E_g;    // Minimum photon wavelength
    needed to break the Cooper pair, m
printf("\nThe energy gap for niobium = %4.2f meV",
    E_g/(e*1e-003));
printf("\nThe minimum photon wavelength needed to
    break the Cooper pair = %4.2e m", lambda);

// Result
// The energy gap for niobium from BCS theory = 2.82
    meV
// The minimum photon wavelength needed to break the
```

```
Cooper pair = 4.39e-004 m
```

Scilab code Exa 10.8 Magnetic field perpendicular to the loop

```
// Scilab Code Ex10.8 : Page-382 (2014)
clc; clear;
r = 1e-002;    // Radius of the loop, m
phi0 = 2.068e-015;    // Magnetic flux penetrating to the loop, T-Sq.m
A = %pi*r^2;    // Area of the loop, Sq.m
B = phi0/A;    // Magnetic field perpendicular to the loop, T
printf("\nThe magnetic field perpendicular to the loop = %4.2e T", B);

// Result
// The magnetic field perpendicular to the loop = 6.58e-012 T
```

Chapter 11

Semiconductor Theory and Devices

Scilab code Exa 11.1 Relative number of electrons with given energies above the valence band

```
// Scilab Code Ex11.1 : Page-399 (2014)
clc; clear;
e = 1.6e-019;     // Energy equivalent of 1 eV, J
k = 1.38e-023;     // Boltzmann constant, J/K
T = 293;     // Room temperature, K
dE = [0.10, 1.0, 10.0];     // Energies above the
    valence band, eV
F_FD = zeros(3);
for i = 1:1:3
    F_FD(i) = 1/(exp(dE(i)*e/(k*T)) + 1);
    printf("\nFor E - E_F = %4.2f eV, F_FD = %4.2e",
        dE(i), F_FD(i));
end

// Result
// For E - E_F = 0.10 eV, F_FD = 1.88e-002
// For E - E_F = 1.00 eV, F_FD = 6.53e-018
// For E - E_F = 10.00 eV, F_FD = 1.40e-172
```

Scilab code Exa 11.3 Hall effect in zinc strip

```

// Scilab Code Ex11.3: Page-402 (2014)
clc; clear;
e = 1.6e-019;     // Energy equivalent of 1 eV, J
rho = 5.92e-008;     // Resistivity of the zinc at
    room temperature, ohm-m
B = 0.25;     // Magnetic field applied perpendicular
     to the strip, T
x = 10.0e-002;     // Length of the zinc strip, m
y = 2.0e-002;     // Width of the zinc strip, m
V = 20e-003;     // Potential difference applied
    across the strip, V
I = 0.400;     // Current through the strip, A
V_H = 0.56e-006;     // Hall voltage that appeared
    across the strip, V
z = rho*x*I/(y*V);     // Thickness of the strip, m
n = I*B/(e*V_H*z);     // Number density of the
    charge carriers, per metre cube
printf("\nThe thickness of the zinc strip = %4.2e m"
    , z);
printf("\nThe number density of the charge carriers
    = %4.2e per metre cube", n);
printf("\nThe charge carries in zinc are positive.")
    ;

// Result
// The thickness of the zinc strip = 5.92e-006 m
// The number density of the charge carriers = 1.89e
    +029 per metre cube
// The charge carries in zinc are positive.
```

Scilab code Exa 11.4 Current through a reverse bias pn junction diode

```
// Scilab Code Ex11.4: Page-408 (2014)
clc; clear;
e = 1.602e-019;    // Energy equivalent of 1 eV, J
k = 1.38e-023;    // Boltzmann constant, J/K
T = 293;    // Room temperature, K
V_f = 0.200;    // Forward voltage, V
I_f = 50e-003;    // Forward current, A
V_r = -0.200;    // Reverse voltage, V
I_r = I_f*(exp(e*V_r/(k*T))-1)/(exp(e*V_f/(k*T)) - 1);    // Reverse current from diode equation, A
printf("\nThe reverse current through pn-juntion diode = %2d micro-ampere", I_r/1e-006);

// Result
// The reverse current through pn-juntion diode = -18 micro-ampere
```

Scilab code Exa 11.5 Energy produced by a solar cell per day

```
// Scilab Code Ex11.5: Page-412 (2014)
clc; clear;
A = 100*100;    // Area of solar cell, Sq.m
t = 12*60*60;    // Time for which the solar cell operates, s
phi = 680;    // Solar flux received by the solar cell, W/Sq.m
eta = 0.30    // Efficiency of the solar array
E_array = eta*phi*A*t;    // Energy produced by solar cell in one 12-hour day, J
```

```
printf("\nThe energy produced by solar cell in one
   12−hour day : %3.1e J", E_array);
P = 100e+006;     // Power output of power plant, W
t = 24*60*60;     // Time for which power plant
   operates, s
E_plant = P*t;     // Energy produced by power plant,
    J
printf("\nThe energy produced by power plant in one
   day : %3.1e J which is about %d times more than
   that produced by solar cell array..!", E_plant,
   ceil(E_plant/E_array));

// Result
// The energy produced by solar cell in one 12−hour
   day : 8.8e+010 J
// The energy produced by power plant in one day :
   8.6e+012 J which is about 99 times more than that
    produced by solar cell array..!
```

Scilab code Exa 11.6 Data bits stored into CD ROM

```

// Scilab Code Ex11.6: Page−418 (2014)
clc; clear;
r1 = 2.30e-002;     // Radius of inner edge of
   storing region of CD−ROM, m
r2 =5.80e-002;     // Radius of outer edge of storing
    region of CD−ROM, m
A = %pi*(r2^2 - r1^2);     // Area of the usable
   region of CD−ROM, Sq.m
N = 700e+006*8;     // Total number of bits in CD−ROM
APB = A/N;     // Area per bit of CD−ROM, Sq.m/bit
t = 1.6e-006;     // Track width of CD_ROM, m
l = APB/t;     // Bit length, m
printf("\nThe surface area of CD−ROM allowed for
```

```
each data bit = %3.1e Sq.m/bit", APB);
printf("\nThe approx. dimensions of each bit along the track = %1.0f micro-metre", l/1e-006);

// Result
// The surface area of CD-ROM allowed for each data bit = 1.6e-012 Sq.m/bit
// The approx. dimensions of each bit along the track = 1 micro-metre
```

Chapter 12

The Atomic Nucleus

Scilab code Exa 12.1 Minimum kinetic energy of a proton in a medium sized nucleus

```
// Scilab Code Ex12.1: Page-432 (2014)
clc; clear;
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant,
    Js
m = 1.67e-027;    // Rest mass of proton, kg
e = 1.602e-019;    // Energy equivalent of 1 eV, J
c = 3.00e+008;    // Speed of light, m/s
delta_x = 8.0e-015;    // Size of the nucleus, m
delta_p = h_bar/(2*delta_x*e);    // Uncertainty in
    momentum of proton from Heisenberg Uncertainty
    Principle, eV-s/m
p_min = delta_p;    // Minimum momentum of the
    proton inside the nucleus, eV-s/m
K = (p_min*c)^2*e/(2*m*c^2*1e+006);    // The
    minimum kinetic energy of the proton in a medium
    sized nuclecus, MeV
printf("\nThe minimum kinetic energy of the proton
    in a medium sized nuclecus = %4.2f MeV", K);
```

```
// Result
// The minimum kinetic energy of the proton in a medium sized nuclecus = 0.08 MeV
```

Scilab code Exa 12.2 Nuclear radius of calcium

```
// Scilab Code Ex12.2: Page-436 (2014)
clc; clear;
c = 3.00e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
m_e = 0.511;     // Rest mass energy of electron, MeV
m_p = 938.3;     // Rest mass energy of proton, MeV
h = 6.62e-034;     // Planck's constant, Js
A = 40;     // Mass number of Ca-40
r0 = 1.2;     // Nuclear radius constant, fm
R = r0*A^(1/3);     // Radius of Ca-40 nucleus, fm
printf("\nThe radius of Ca-40 nucleus = %3.1f fm", R);
lambda = 2.0;     // de-Broglie wavelength to distinguish a distance at least half the radius, fm
// Electron energy
E = ceil(sqrt(m_e^2+(h*c/(lambda*e*1e+006*1e-015))^2));     // Total energy of the probing electron, MeV
K = E - m_e;     // Kinetic energy of probing electron, MeV
printf("\nThe kinetic energy of probing electron = %3d MeV", ceil(K));
// Proton energy
E = ceil(sqrt(m_p^2+(h*c/(lambda*e*1e+006*1e-015))^2));     // Total energy of the probing electron, MeV
K = E - m_p;     // Kinetic energy of probing electron, MeV
```

```
printf("\nThe kinetic energy of probing proton = %3d MeV", ceil(K));

// Result
//The radius of Ca-40 nucleus = 4.1 fm
// The kinetic energy of probing electron = 620 MeV
// The kinetic energy of probing proton = 187 MeV
// The answers are deviated due to the approximations used in the textbook
```

Scilab code Exa 12.3 Radii of U238 and He4 nuclei

```
// Scilab Code Ex12.3: Page-437 (2014)
clc; clear;
A_U238 = 238;     // Mass number of U-238
A_He4 = 4;     // Mass number of He-4
r0 = 1.2;     // Nuclear radius constant, nm
R_U238 = r0*A_U238^(1/3);     // Radius of U-238 nucleus, fm
R_He4 = r0*A_He4^(1/3);     // Radius of He-4 nucleus, fm
printf("\nThe radii of U-238 and He-4 nuclei are %3.1f fm and %3.1f fm repectively", R_U238, R_He4);
ratio = R_U238/R_He4;     // Ratio of the two radii
printf("\nThe ratio of radius to U-238 to that of He-4 = %3.1f", ratio);

// Result
// The radii of U-238 and He-4 nuclei are 7.4 fm and 1.9 fm repectively
// The ratio of radius to U-238 to that of He-4 = 3.9
```

Scilab code Exa 12.4 A proton subjected to the magnetic field

```
// Scilab Code Ex12.4: Page-438 (2014)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
c = 3.00e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
B = 2.0;     // Applied magnetic field, T
mu_N = 3.15e-008;     // Nucleon magnetic moment, eV/T
mu_p = 2.79*mu_N;     // Proton magnetic moment, eV/T
delta_E = 2*mu_p*B;     // Energy difference between the up and down proton states, eV
f = delta_E*e/h;     // Frequency of electromagnetic radiation that flips the proton spins, Hz
lambda = c/f;     // Wavelength of electromagnetic radiation that flips the proton spins, m
printf("\nThe energy difference between the up and down proton states = %3.1e eV", delta_E);
printf("\nThe frequency of electromagnetic radiation that flips the proton spins = %2d MHz", f/1e+006);
printf("\nThe wavelength of electromagnetic radiation that flips the proton spins = %3.1f m", lambda);

// Result
// The energy difference between the up and down proton states = 3.5e-007 eV
// The frequency of electromagnetic radiation that flips the proton spins = 85 MHz
// The wavelength of electromagnetic radiation that flips the proton spins = 3.5 m
```

Scilab code Exa 12.5 Binding energy of Be

```
// Scilab Code Ex12.5: Page-443 (2014)
clc; clear;
c = 3.00e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
u = 931.5;     // Energy equivalent of 1 amu, MeV
m_n = 1.008665;     // Mass of a neutron, u
M_H = 1.007825;     // Mass of a proton, u
M_He = 4.002603;     // Mass of helium nucleus
M_Be = 8.005305;     // Mass of Be-8, u
B_Be = (4*m_n+4*M_H - M_Be)*u;
B_Be_2alpha = (2*M_He - M_Be)*u;
printf("\nThe binding energy of Be-8 = %4.1f MeV and is positive", B_Be);
if (B_Be_2alpha > 0) then
    printf("\nThe Be-4 is stable w.r.t. decay into two alpha particles.");
else
    printf("\nThe Be-4 is unstable w.r.t. decay into two alpha particles since the decay has binding energy of %5.3f MeV", B_Be_2alpha);
end

// Result
// The binding energy of Be-8 = 56.5 MeV and is positive
// The Be-4 is unstable w.r.t. decay into two alpha particles since the decay has binding energy of -0.092 MeV
```

Scilab code Exa 12.6 Total Coulomb energy of U238

```
// Scilab Code Ex12.6: Page-444 (2014)
clc; clear;
Z = 92;     // Atomic number of U-238
A = 238;     // Mass number of U-238
```

```
E_Coul = 0.72*Z*(Z-1)*A^(-1/3);    // Total Coulomb energy of U-238, MeV
printf("\nThe total Coulomb energy of U-238 = %3d MeV", E_Coul);

// Result
// The total Coulomb energy of U-238 = 972 MeV
```

Scilab code Exa 12.8 Binding energy per nucleon

```
// Scilab Code Ex12.8: Page-447 (2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
m_p = 1.007825;    // Mass of proton, amu
m_n = 1.008665;    // Mass of neutron, amu
M_Ne = 19.992440; // Mass of Ne-20 nucleus, amu
M_Fe = 55.934942;    // Mass of Fe-56 nucleus, amu
M_U = 238.050783;    // Mass of U-238 nucleus, amu
A_Ne = 20;    // Mass number of Ne-20 nucleus
A_Fe = 56;    // Mass number of Fe-56 nucleus
A_U = 238;    // Mass number of U-238 nucleus
BE_Ne = [10*m_p+10*m_n-M_Ne]*u;    // Binding energy of Ne-20 nucleus, MeV
BE_Fe = [26*m_p+30*m_n-M_Fe]*u;    // Binding energy of Fe-56 nucleus, MeV
BE_U = [92*m_p+146*m_n-M_U]*u;    // Binding energy of U-238 nucleus, MeV
printf("\nThe binding energy per nucleon for Ne-20 : %4.2f MeV/nucleon", BE_Ne/A_Ne);
printf("\nThe binding energy per nucleon for Fe-56 : %4.2f MeV/nucleon", BE_Fe/A_Fe);
printf("\nThe binding energy per nucleon for U-238 : %4.2f MeV/nucleon", BE_U/A_U);

// Result
```

```
// The binding energy per nucleon for Ne−20 : 8.03 MeV/nucleon
// The binding energy per nucleon for Fe−56 : 8.79 MeV/nucleon
// The binding energy per nucleon for U−238 : 7.57 MeV/nucleon
```

Scilab code Exa 12.10 Radiactive decay of Po210

```
// Scilab Code Ex12.10: Page−451 (2014)
clc; clear;
N_A = 6.023e+023;    // Avogadro's number
T = 138*24*3600;    // Half life of Po−210, s
R = 2000;    // Activity of Po−210, disintegrations/s
M = 0.210;    // Gram molecular mass of Po−210, kg
f = 1/3.7e+010;    // Conversion factor to convert from decays/s to Ci
A = R*f/1e-006;    // Activity of Po−210, micro−Ci
N = R*T/log(2);    // Number of radiactive nuclei of Po−210, nuclei
m = N*M/N_A;    // Mass of Po−210 sample, kg
printf("\nThe activity of Po−210 = %5.3f micro−Ci", A);
printf("\nThe mass of Po−210 sample = %3.1e kg", m);

// Result
//  The activity of Po−210 = 0.054 micro−Ci
// The mass of Po−210 sample = 1.2e−014 kg
```

Scilab code Exa 12.11 Time of decay of F18 isotope

```
// Scilab Code Ex12.11: Page−452 (2014)
```

```
clc; clear;
T = 110;     // Half life of F-18, min
f_remain = 0.01;     // Fraction of the F-18 sample
    remained
t = -log(0.01)/(log(2)*60)*T;     // Time taken by
    the F-18 sample to decay to 1 percent of its
    initial value, h
printf("\nThe time taken for 99 percent of the F-18
    sample to decay = %4.1f h", t);

// Result
// The time taken for 99 percent of the F-18 sample
    to decay = 12.2 h
```

Scilab code Exa 12.12 Alpha activity of 10 kg sample of U235

```
// Scilab Code Ex12.12: Page-452 (2014)
clc; clear;
N_A = 6.023e+023;     // Avogadro's number
M = 10e+03;     // Mass of the U-235, g
M_U235 = 235;     // Molecular mass of U-235, g
t_half = 7.04e+008;     // Half life of U-235, y
N = M*N_A/M_U235;     // Number of U-235 atoms in 10
    kg sample
R = log(2)*N/t_half;     // The alpha activity of 10
    kg sample of U-235, decays/y
printf("\nThe alpha activity of 10 kg sample of U
    -235 = %3.1e Bq", R/(365.25*24*60*60));

// Result
// The alpha activity of 10 kg sample of U-235 = 8.0
    e+008 Bq
```

Scilab code Exa 12.13 Non emission of a neutron by U230

```
// Scilab Code Ex12.13: Page-453 (2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 u, MeV
M_U230 = 230.033927;    // Atomic mass of U-230, u
m_n = 1.008665;    // Mass of a neutron, u
M_H = 1.007825;    // Mass of hydrogen, u
M_U229 = 229.033496;    // Gram atomic mass of U-230
Q = (M_U230 - M_U229 - m_n)*u;    // Q-value of the reaction emitting a neutron
if (Q < 0) then
    printf("\nThe neutron decay in this reaction is not possible.");
else
    printf("\nThe neutron decay in this reaction is possible.");
end
Q = (M_U230 - M_U229 - M_H)*u;    // Q-value of the reaction emitting a proton
if (Q < 0) then
    printf("\nThe proton decay in this reaction is not possible.");
else
    printf("\nThe proton decay in this reaction is possible.");
end

// Result
// The neutron decay in this reaction is not possible.
// The proton decay in this reaction is not possible.
```

Scilab code Exa 12.15 Possible reaction with Fe55 isotope

```
// Scilab Code Ex12.15: Page-461 (2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 u, MeV
M_Fe55 = 54.938298;    // Atomic mass of Fe-55, u
M_Mn55 = 54.938050;    // Atomic mass of Mn-55, u
m_e = 0.000549;    // Mass of the electron, u
Q = (M_Fe55 - M_Mn55 - 2*m_e)*u;    // Q-value of
    the reaction undergoing beta+ decay, MeV
if (Q < 0) then
    printf("\nThe beta+ decay is not allowed for Fe
        -55");
else
    printf("\nThe beta+ decay is allowed for Fe-55")
        ;
end
Q = (M_Fe55 - M_Mn55)*u;    // Q-value of the
    reaction undergoing electron capture, MeV
if (Q < 0) then
    printf("\nFe-55 may not undergo electron capture
        ");
else
    printf("\nFe-55 may undergo electron capture");
end

// Result
// The beta+ decay is not allowed for Fe-55
// Fe-55 may undergo electron capture
```

Scilab code Exa 12.16 Allowed decay modes for Ac226

```
// Scilab Code Ex12.16: : Page-461 (2014)
clc; clear;
function [] = check_allowance(Q, decay_type)
    if (Q < 0) then
    printf("\nThe %s is not allowed for Ac-226",
```

```
        decay_type);
else
    printf("\nThe %s is allowed for Ac-226", decay_type);
end
endfunction
u = 931.5;    // Energy equivalent of 1 u, MeV
M_Ac226 = 226.026090;    // Atomic mass of Ac-226, u
M_Fr222 = 222.017544;    // Atomic mass of Fr-222, u
M_He4 = 4.002603;    // Atomic mass of He-4, u
M_Th226 = 226.024891;    // Atomic mass of M_Th226, u
M_Ra226 = 226.025403;    // Atomic mass of M_Ra226, u
m_e = 0.000549;    // Mass of the electron, u
// Alpha Decay
Q = (M_Ac226 - M_Fr222 - M_He4)*u;    // Q-value of the reaction undergoing alpha decay, MeV
check_allowance(Q, "alpha decay");
// Beta- Decay
Q = (M_Ac226 - M_Th226)*u;    // Q-value of the reaction undergoing beta- decay, MeV
check_allowance(Q, "beta- decay");
// Beta+ Decay
Q = (M_Ac226 - M_Ra226 - 2*m_e)*u;    // Q-value of the reaction undergoing beta+ decay, MeV
check_allowance(Q, "beta+ decay");
// Electron Capture
Q = (M_Ac226 - M_Ra226)*u;    // Q-value of the reaction undergoing electron capture, MeV
check_allowance(Q, "electron capture");

// Result
// The alpha decay is allowed for Ac-226
// The beta- decay is allowed for Ac-226
// The beta+ decay is not allowed for Ac-226
// The electron capture is allowed for Ac-226
```

Scilab code Exa 12.17 Error introduced in the gamma ray energy

```
// Scilab Code Ex12.17: Page-463(2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 u, MeV
E_ex = 0.072;    // Energy of the excited state, MeV
M = 226*u*1e;    // Energy equivalent of atomic mass
    of Th-226, MeV
x = E_ex/(2*M);    // The error introduced in the
    gamma ray energy by approximation
printf("\nThe error introduced in the gamma ray
    energy by approximation = %3.1e", x);

// Result
// The error introduced in the gamma ray energy by
    approximation = 1.7e-007
```

Scilab code Exa 12.18 Age of the uranium ore

```
// Scilab Code Ex12.18: Page-467(2014)
clc; clear;
t_half = 4.47e+009;    // The half life of uranium
    ore, years
R_prime = 0.60;    // The ratio of Pb206 abundance
    to that of U238
t = t_half/log(2)*log(R_prime + 1);    // Age of the
    uranuim ore, years
printf("\nThe age of U-238 ore = %3.1e years", t);

// Result
// The age of U-238 ore = 3.0e+009 years
```

Scilab code Exa 12.19 C14 dating to determine age of bone

```
// Scilab Code Ex12.19: Page-469(2014)
clc; clear;
t_half = 5730;    // The half life of uranium ore, years
R0 = 1.2e-012;    // The initial ratio of C14 to C12 at the time of death
R = 1.10e-012;    // The final ratio of C14 to C12 t years after death
// As R = R0*exp(-lambda*t), solving for t
t = -log(R/R0)*t_half/log(2);    // Age of the bone, years
printf("\nThe %3d years age of bone does not date from the Roman Empire.", ceil(t));

// Result
// The 720 years age of bone does not date from the Roman Empire.
```

Chapter 13

Nuclear Interactions and Applications

Scilab code Exa 13.1 Number of neutrons produced in collision of alpha particle and carbon target

```
// Scilab Code Ex13.1: Page-479(2014)
clc; clear;
N_A = 6.02e+023;    // Avogadro's number
e = 1.6e-019;    // Charge on an electron, C
q = 2*e;    // Charge on the alpha particle, C
rho = 1.9;    // Density of carbon target, atoms/cc
N_M = 1;    // Number of atoms per molecule
M_g = 12;    // Gram atomic mass of C12 isotope, g/
   mol
sigma = 25e-031;    // Total cross section for the
   reaction, Sq.m
t = 1e-006;    // Thickness of carbon target, m
I_beam = 1e-006;    // Beam current of akpha
   particle, ampere
time = 3600;    // Time for which the alpha particle
    beam is incident on the target, s
n = rho*N_A*N_M/M_g;    // Number of nuclei per unit
    volume, per cc
```

```
P = n*t*sigma*1e+006;      // Probability of scattering of alpha particles
N_I = I_beam*time/q;     // Number of incident alpha particles
N_n = N_I*P;     // Number of neutrons produced in the reaction
printf("\nThe number of neutrons produced in the reaction = %3.1e neutrons", N_n);

// Result
// The number of neutrons produced in the reaction = 2.7e+009 neutrons
```

Scilab code Exa 13.2 Likelihood of a neutron production than a proton

```
// Scilab Code Ex13.2: Page-480(2014)
clc; clear;
sigma_n = 3;     // Differential cross setion of production of the neutron, mb/sr
sigma_p = 0.2;     // Differential cross setion of production of the proton, mb/sr
// As P_n = sigma_n and P_p = sigma_p, so
P_ratio = sigma_n/sigma_p;     // The likelihood of a neutron production than a proton
printf("\nThe likelihood of the neutron production than the proton = %2d", P_ratio);

// Result
// The likelihood of the neutron production than the proton = 15
```

Scilab code Exa 13.3 Nuclear reaction observed by Rutherford

```
// Scilab Code Ex13.3: Page-481(2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
M_He = 4.002603;    // Mass of He-4 nucleus, u
M_N = 14.003074;    // Mass of N-14 nucleus, u
M_H = 1.007825;    // Mass of hydrogen nucleus, u
M_O = 16.999132;    // Mass of O-16 nucleus, u
K_alpha = 7.7;    // The kinetic energy of alpha particle, MeV
Q = (M_He + M_N - M_H - M_O)*u;    // The ground state Q-value of the nuclear reaction, MeV
// As Q = K_p + K_O - K_alpha, solving for K_p + K_O
K = Q + K_alpha;    // The sum of kinetic energy of the products, MeV
printf("\nThe ground state Q-value of the endoergic nuclear reaction = %5.3f MeV", Q);
printf("\nThe sum of kinetic energy of the products = %3.1f MeV", K);

// Result
// The ground state Q-value of the endoergic nuclear reaction = -1.192 MeV
// The sum of kinetic energy of the products = 6.5 MeV
```

Scilab code Exa 13.4 Final energy of excitation of product nucleus in the nuclear reaction

```
// Scilab Code Ex13.4: Page-484(2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
K_lab = 14.6;    // Kinetic energy of incident aplha particle, MeV
Mx = 4;    // Mass number of projectile nucleus
MX = 12;    // Mass number of target nucleus
```

```
M_He = 4.002603;     // Mass of He nucleus, u
M_C = 12.0 // Mass of carbon nucleus, u
M_O = 15.994915;     // Mass of oxygen nucleus, u
K_cm = MX/(Mx + MX)*K_lab;     // Kinetic energy
    available in the centre of mass, MeV
E_O = (M_He + M_C - M_O)*u;     // The ground state
    excitation energy of O-16, MeV
E_final_O = K_cm + E_O;     // The final excitation
    energy of O-16, MeV
printf("\nThe final excitation energy of O-16 = %4.2
    f MeV", E_final_O);

// Result
// The final excitation energy of O-16 = 18.11 MeV
```

Scilab code Exa 13.5 Ground state Q value of the induced fission reaction

```

// Scilab Code Ex13.5: Page-487(2014)
clc; clear;
u = 931.5;     // Energy equivalent of 1 amu, MeV
M_U235 = 235.0439;     // Mass of U-235 nucleus, u
m_n = 1.0087;     // Mass of a neutron, u
M_Zr99 = 98.9165;     // Mass of Zr-99 nucleus, u
M_Te134 = 133.9115;     // Mass of Te-134 nucleus, u
Q = (M_U235 + m_n - M_Zr99 - M_Te134 - 3*m_n)*u;
    // Q-value of the reaction, MeV
printf("\nThe ground state Q-value of the induced
    fission reaction = %3d MeV", ceil(Q));

// Result
// The ground state Q-value of the induced fission
    reaction = 185 MeV
```

Scilab code Exa 13.6 Excitation energy of the compound nuclei

```
// Scilab Code Ex13.6: Page-488(2014)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
m_n = 1.0087;    // Mass of a neutron, u
M_U235 = 235.0439;    // Mass of U-235 nucleus, u
M_U236 = 236.0456;    // Mass of U-236 nucleus, u
M_U238 = 238.0508;    // Mass of U-238 nucleus, u
M_U239 = 239.0543;    // Mass of U-239 nucleus, u
E_U236 = (m_n + M_U235 - M_U236)*u;    // Excitation energy of U-236 nucleus, MeV
E_U239 = (m_n + M_U238 - M_U239)*u;    // Excitation energy of U-239 nucleus, MeV
printf("\nThe excitation energy of U-236 nucleus = %3.1f MeV", E_U236);
printf("\nThe excitation energy of U-239 nucleus = %3.1f MeV", E_U239);

// Result
// The excitation energy of U-236 nucleus = 6.5 MeV
// The excitation energy of U-239 nucleus = 4.8 MeV
```

Scilab code Exa 13.7 Nuclear fission through neutron capture

```

// Scilab Code Ex13.7: Page-490(2014)
clc; clear;
t = 30e-003;    // Time during which the number of fissions is to be calculated, s
E = 185;    // Energy produced for each fission, MeV
```

```
delta_t = 5e-006;     // Average time during which a neutron is captured, s
fs = t/delta_t;     // Number of fission cycles within 30 ms
N = (1.01)^fs;     // Number of fissions that occur in 30 ms
E_total = N*E;     // Total energy produced in 30 ms, MeV
printf("\nThe total number of fissions that occur in %d ms = %3.1e", t/1e-003, N);
printf("\nThe total energy produced = %3.1e MeV", E_total);

// Result
// The total number of fissions that occur in 30 ms = 8.5e+025
// The total energy produced = 1.6e+028 MeV
// The answers are given wrong in the textbook
```

Scilab code Exa 13.8 Fusion reaction in supergiant stars

```
// Scilab Code Ex13.8: Page-500(2014)
clc; clear;
u = 931.5;     // Energy equivalent of 1 amu, MeV
M_He = 4.002603;     // Mass of He nucleus, u
M_C = 12.0 // Mass of carbon nucleus, u
M_O = 15.994915;     // Mass of oxygen nucleus, u
Q = (M_He + M_C - M_O)*u;     // Q-value of the reaction, MeV
printf("\nThe energy expended in the fusion reaction inside supergiant star = %3.1f MeV", Q);

// Result
//The energy expended in the fusion reaction inside supergiant star = 7.2 MeV
```

Scilab code Exa 13.9 Ignition temperature needed for the fusion reaction between a deuterium and a tritium

```
// Scilab Code Ex13.9: Page-502(2014)
clc; clear;
k = 1.38e-023;     // Boltzmann constant, J/K
r = 3e-015;     // Distance at whcih the nuclear force becomes effective, m
e = 1.6e-019;   // Charge on an electron, C
K = 9e+009;     // Coulomb's constant, N-Sq.m/C^2
V = K*e^2/r;     // Coulomb potential energy, J
// As V = 3/2*k*T, solving for T
T = 2/3*V/k;     // The ignition temperature needed for the fusion reaction between deuterium and a tritium, K
printf("\nThe ignition temperature needed for the fusion reaction between a deuterium and a tritium = %3.1e K", T);

// Result
// The ignition temperature needed for the fusion reaction between a deuterium and a tritium = 3.7e+009 K
```

Scilab code Exa 13.10 Neutron beam study of atomic structures

```
// Scilab Code Ex13.10: Page-509(2014)
clc; clear;
k = 1.38e-023;     // Boltzmann constant, J/K
e = 1.6e-019;   // Energy equivalent of 1 eV, J
h = 6.62e-034;     // Planck's oconstant, Js
```

```
m = 1.67e-027;     // Mass of the neutron, kg
lambda = 0.060e-009;     // Wavelength of the neutron
     beam, m
p = h/lambda;     // Momentum of the neutron from de-
    Broglie relation, kg-m/s
K = p^2/(2*m*e);     // Kinetic energy of the neutron
     needed to study atomic structure, eV
// As K = 3/2*k*T, solving for T
T = 2/3*K*e/k;     // The temperature of the neutron
    needed to study atomic structure, K
printf("\nThe energy and temperature of the neutron
    needed to study the atomic structure of solids =
    %4.2f eV and %4d K respectively", K, T);

// Result
// The energy and temperature of the neutron needed
    to study the atomic structure of solids = 0.23 eV
     and 1760 K respectively
```

Chapter 14

Particle Physics

Scilab code Exa 14.1 Mass of the meson from Heisenberg uncertainty principle

```
// Scilab Code Ex14.1: Page-522(2014)
clc; clear;
e = 1.6e-019;   // Energy equivalent of 1 eV, J
h = 6.62e-034;     // Planck's oconstant, Js
c = 3.00e+008;     // Speed of light in vacuum, m/s
h_bar = h/(2*%pi);     // Reduced Planck's constant,
    Js
R_N = 1e-015;      // Range of nuclear force, m
// As delta_E*delta_t = h_bar/2 and delta_E = m_pion
    *c^2, solving for m_pion
m_pion = h_bar*c/(2*R_N*e*1e+006);     // Mass of the
     meson, MeV/c^2
printf("\nThe estimated mass of meson from
    Heisenberg uncertainty principle = %d MeV/c^2",
    round(m_pion));

// Result
// The estimated mass of meson from Heisenberg
    uncertainty principle = 99 MeV/c^2
// The answer is rounded off in the textbook
```

Scilab code Exa 14.2 Range of the weak interaction

```
// Scilab Code Ex14.2: Page-525(2014)
clc; clear;
e = 1.6e-019;  // Energy equivalent of 1 eV, J
h = 6.62e-034;     // Planck's oconstant, Js
c = 3.00e+008;     // For simplicity assume speed of light to be unity
h_bar = h/(2*%pi);     // Reduced Planck's constant, Js
m_W = 80.4;     // Energy equivalent of mass of W- particle, MeV
R_W = h_bar*c/(2*m_W*e*1e+009);     // Range of W- particle, m
delta_t = h_bar/(2*m_W*e*1e+009);     // Time during which the energy conservation is violated, s
printf("\nThe range of W- particle = %3.1e m", R_W);
printf("\nThe time during which the energy conservation is violated = %1.0e s", delta_t);

// Result
// The range of W- particle = 1.2e-018 m
// The time during which the energy conservation is violated = 4e-027 s
```

Scilab code Exa 14.10 Fixed target accelerators

```
// Scilab Code Ex14.10: Page-548(2014)
clc; clear;
m_p = 0.938;     // Rest mass energy of the proton, GeV
```

```
K = 6.4;     // Kinetic energy of the proton projectile , GeV
E_cm = sqrt(2*m_p^2+2*m_p*K);     // Centre of mass energy of proton collsion with the fixed proton target , GeV
Q = 2*m_p - 4*m_p;     // Q value of the reaction , GeV
K_th = -3*Q;     // Threshold kinetic energy required to produce the antiprotons , GeV
K = 1000;     // Kinetic energy of the protons in Tevatron , GeV
E_cm_T = sqrt(2*m_p^2+2*m_p*K);     // Centre-of-mass energy available for the reaction for the Tevatron , GeV
printf("\nThe available energy in the center on mass = %4.2f GeV", E_cm);
printf("\nThe threshold kinetic energy required to produce the antiprotons = %3.1f GeV", K_th);
printf("\nThe centre-of-mass energy available for the reaction for the Tevatron = %d GeV", E_cm_T);

// Result
// The available energy in the center on mass = 3.71 GeV
// The threshold kinetic energy required to produce the antiprotons = 5.6 GeV
// The centre-of-mass energy available for the reaction for the Tevatron = 43 GeV
```

Scilab code Exa 14.11 Energy required by a fixed target accelerator to match with that available for colliding beams at LHC

```
// Scilab Code Ex14.11: Page-550(2014)
clc; clear;
m_p = 0.938;     // Rest mass energy of the proton ,
```

```
GeV
E_cm = 14000;     // Centre of mass energy of colliding proton beams at LHC, GeV
// As E_cm = sqrt(2*m_p^2+2*m_p*K), solving for K
K = E_cm^2*1e+009/(2*m_p);     // Approx. kinetic energy of the protons needed for fixed-target experiment, eV
printf("\nThe kinetic energy of the protons needed for fixed-target experiment = %3.1e eV", K);

// Result
// The kinetic energy of the protons needed for fixed-target experiment = 1.0e+017 eV
```

Chapter 15

General Relativity

Scilab code Exa 15.1 Gravitational time dilation effect

```
// Scilab Code Ex15.1: Page-562(2014)
clc; clear;
g = 9.8;    // Acceleration due to gravity, m/sec^2
H = 10000;    // Altitude of the aeroplane above the surface of earth, m
c = 3.00e+008;    // Speed of light in free space, m/s
T = 45*3600;    // Time taken by the airplane to from eastward to westward trip, s
delta_T_G = g*H*T/(c^2*1e-009);    // Time difference in the two clocks due to gravitational redshift, ns
C = 4e+007;    // Circumference of the earth, m
v = 300;    // Speed of the jet airplane, m/s
T0 = C/v;    // Time of flight of jet airplane very near the surface of the earth, s
bita = v/c;    // Boost parameter
// As from special relativity time dilation relation, T = T0*sqrt(1-bita^2), solving for T0 - T = delta_T_R, we have
delta_T_R = T0*(1-sqrt(1-bita^2))/1e-009;    // Time
```

```
difference in the two clocks due to special relativity, ns
printf("\nThe gravitational time dilation effect of %d ns is larger than the approximate %4.1f ns of that of special relativity.", ceil(delta_T_G), delta_T_R);

// Result
// The gravitational time dilation effect of 177 ns is larger than the approximate 66.7 ns of that of special relativity.
```

Scilab code Exa 15.2 Schwarzschild radius for the sun and the earth

```
// Scilab Code Ex15.2: Page-567(2014)
clc; clear;
c = 3.00e+008;     // Speed of light in free space, m/s
G = 6.67e-011;     // Newton's gravitational constant, N-Sq.m/per kg square
M_S = 2.0e+030;     // Mass of the sun, kg
M_E = 6.0e+024;     // Mass of the earth, kg
r_S = 2*G*M_S/(c^2*1e+003);     // Schwarzschild radius for sun, km
r_E = 2*G*M_E/(c^2*1e-003);     // Schwarzschild radius for earth, mm
printf("\nThe Schwarzschild radius for sun = %d km", ceil(r_S));
printf("\nThe Schwarzschild radius for earth = %d mm", ceil(r_E));

// Result
// The Schwarzschild radius for sun = 3 km
// The Schwarzschild radius for earth = 9 mm
```

Scilab code Exa 15.3 Time taken by a black hole to radiate its energy

```
// Scilab Code Ex15.3: Page-568(2014)
clc; clear;
c = 3.00e+008;     // Speed of light in free space, m/s
G = 6.67e-011;     // Newton's gravitational constant, N-Sq.m/per kg square
h = 6.62e-034;     // Planck's constant, Js
h_bar = h/(2*%pi);     // Reduced Planck's constant, Js
sigma = 5.67e-008;     // Stefan-Boltzmann constant, W per Sq.m per K^4
k = 1.38e-023;     // Boltzmann constant, J/K
M0 = 1.99e+030;     // Mass of the sun, kg
alpha = 2*sigma*h_bar^4*c^6/((8*%pi)^3*k^4*G^2);    // A constant, kg^3/s
t = integrate('1/alpha*M^2', 'M', 0, 3*M0);
printf("\nThe time required for the 3-solar-mass black hole to evaporate = %3.1e y", t/(365.25*24*60*60));

// Result
// The time required for the 3-solar-mass black hole to evaporate = 5.7e+068 y
```

Chapter 16

Cosmology and Modern Astrophysics

Scilab code Exa 16.1 Hubble constant determinantion

```
// Scilab Code Ex16.1: Page-581(2014)
clc; clear;
H0 = 22;    // Value of Hubble constant, km/s per
   million ly
parsec = 3.26;    // The value of 1 parsec, light
   years
printf("\nThe value of Hubble constant = %d km/s per
   Mpc", ceil(H0*parsec));

// Result
// The value of Hubble constant = 72 km/s per Mpc
```

Scilab code Exa 16.2 Current ratio of protons to neutrons in the universe

```
// Scilab Code Ex16.2: Page-583(2014)
clc; clear;
```

```
M = 1;    // Let the current mass of the universe be unity
m_u = 1;    // Mass equivalent of 1 amu, u
N_n = 2;    // Number of neutrons in helium
N_p = 2;    // Number of protons in helium
M_p = 0.75*M*m_u;    // Total mass of protons
M_He = 0.25*M*m_u;    // Total mass of helium
N_fp = M_p/M_He*(N_n + N_p);    // Total number of free protons for every He-4
N_P = N_fp + N_p;    // Total number of protons per He-4
ratio = N_P/N_n;    // Current ratio of protons to the neutrons in the universe
printf("\nThe current ratio of protons to the neutrons in the universe = %d", ratio);

// Result
// The current ratio of protons to the neutrons in the universe = 7
```

Scilab code Exa 16.3 Ratio of protons to neutrons at 10 billion kelvin temperature of the universe

```
// Scilab Code Ex16.3: Page-587(2014)
clc; clear;
m_n = 939.566;    // Rest mass of the neutron, MeV/c^2
m_p = 938.272;    // Rest mass of the proton, MeV/c^2
e = 1.6e-019;    // Energy equivalent of 1 eV, J
c = 1;    // For simplicity assume speed of light of light to be unity
T = 1e+010;    // Temperature of the universe, K
delta_m = m_n - m_p;    // Mass difference between a proton and a neutron, MeV/c^2
```

```
k = 1.38e-023;     // Boltzmann constant, J/k
// As from Maxwell-Boltzmann distribution from thermodynamics, N = exp(-m*c^2/(k*T)), so
ratio = exp(delta_m*c^2*1e+006*e/(k*T));     // Ratio of protons to neutrons in the universe at 10 billion kelvin
printf("\nThe ratio of protons to neutrons in the universe at 10 billion kelvin = %3.1f", ratio);

// Result
// The ratio of protons to neutrons in the universe at 10 billion kelvin = 4.5
```

Scilab code Exa 16.4 Mean temperature of the sun

```
// Scilab Code Ex16.4: Page-589(2014)
clc; clear;
M = 1.99e+030;     // Mass of the sun, kg
G = 6.67e-011;     // Universal gravitational constant, N-Sq.m/kg^2
k = 1.38e-023;     // Boltzmann constant, J/K
R = 6.96e+008;     // Radius of the sun, m
m = 1.67e-027;     // Rest mass of the proton, kg
PE = 3/5*(G*M^2/R);     // Self potential energy of the sun, J
// As KE = 1/3*(M/m_p)*m_p*v^2, solving for v
v = sqrt(2*PE/M);     // Velocity of a proton inside the sun, m/s
// From kinetic theory of gases, v = sqrt(3*k*T/m), solving for T
T = m*v^2/(3*k);     // The mean temperature of the sun, K
printf("\nThe mean temperature of the sun = %1.0e K", T);

```

```
// Result
// The mean temperature of the sun = 9e+006 K
```

Scilab code Exa 16.5 Radius of the neutron star

```
// Scilab Code Ex16.5: Page-590(2014)
clc; clear;
M_sun = 1.99e+030;    // Mass of the sun, kg
m_n = 1.675e-027;    // Rest mass of the neutron, kg
h = 6.62e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Planck's constant, Js
G = 6.67e-011;    // Universal gravitational
    constant, N-Sq.m/kg^2
N = 2*M_sun/m_n;    // Number of neutrons in the
    neutron star
V = (6.5*h_bar^2/(N^(1/3)*m_n^3*G))^3;    // Volume
    of the neutron star, metre cube
R = (3/(4*%pi)*V)^(1/3);    // The radius of neutron
    star, m
printf("\nThe radius of the neutron star of 2 solar
    masses = %d km", ceil(R/1e+003));

// Result
// The radius of the neutron star of 2 solar masses
    = 11 km
```

Scilab code Exa 16.6 Redshift versus recession velocity

```
// Scilab Code Ex16.6: Page-598(2014)
clc; clear;
c = 1;    // Assume speed of light to be unity
```

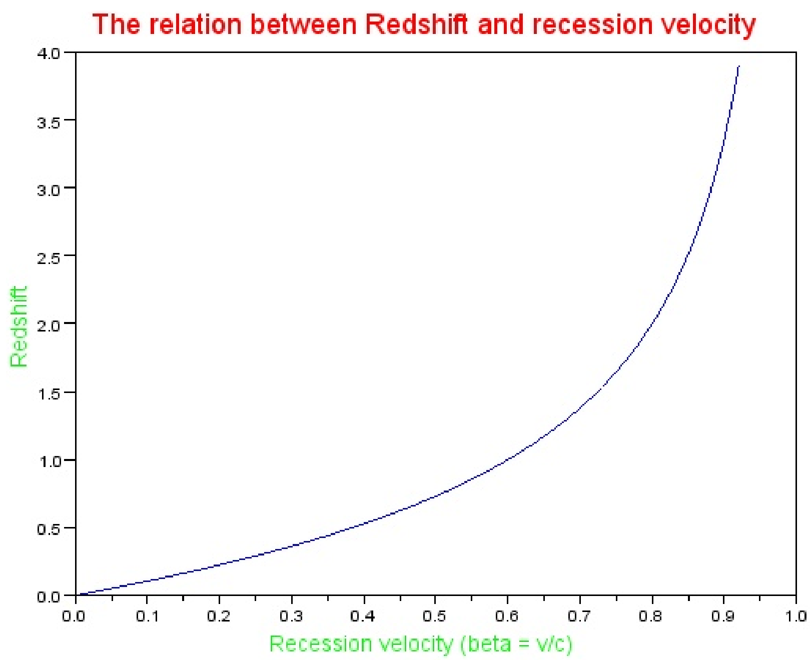

Figure 16.1: Redshift versus recession velocity

```
clf();
v = [0:0.01:0.92]';
bita = v/c;         // Recession velocity ratio
for i = 1:1:93
    red_shift(i) = sqrt((1+bita(i))/(1-bita(i)))-1;
end
plot(bita, red_shift);
title('The relation between Redshift and recession velocity', 'fontsize', 4, 'color','red', 'position', [0.02, 4.1]);
xlabel('Recession velocity (beta = v/c)', 'fontsize', 3, 'color', 'green');
ylabel('Redshift', 'fontsize', 3, 'color', 'green');

// Result
// The plot between Redshift vs recession velocity is as shown in the Fig.
```

Scilab code Exa 16.7 Difference in the travel times of different mass neutrinos

```
// Scilab Code Ex16.7: Page-598(2014)
clc; clear;
c = 1;     // For simplicity assume speed of light to be unity, m/s
d = 1.6e+005;     // Distance of the supernova 1987A from the earth, ly
m = 16;     // Mass of heavier neutrino, eV/c^2;
E = 20e+006;     // Energy of the neutrino, eV
delta_t = d/(2*c)*(m/E)^2;     // Difference between the travel times of the lighter and the massive neutrinos, y
printf("\nThe difference between the travel times of the lighter and the massive neutrinos = %3.1f s", delta_t*(365.25*24*60*60));
```

```

// Result
// The difference between the travel times of the lighter and the massive neutrinos = 1.6 s
```

Scilab code Exa 16.8 Critical density of the universe

```
// Scilab Code Ex16.8: Page-602(2014)
clc; clear;
c = 3.00e+008;     // Speed of light, m/s
H = 22;     // Hubble constant, km/s per million ly
G = 6.67e-011;     // Universal gravitational constant, N-Sq.m/kg^2
rho_c = 3/(8*%pi)*H^2/G*1e+003/(c*365.25*24*60*60*1e+006)^2;     // The critical density of the universe, g/cc
printf("\nThe critical density of the universe = %3.1e g/cc", rho_c);

// Result
//The critical density of the universe = 9.7e-030 g/cc
```

Scilab code Exa 16.9 Upper limit of the age of the universe

```
// Scilab Code Ex16.9: Page-604(2014)
clc; clear;
H0 = 71;     // Hubble cinstant, km/s per Mpc
tau = 1/H0*1e+006*3.26*9.46e+012/3.16e+007;     // The upper limit of the age of the universe, y
printf("\nThe upper limit of the age of the universe = %4.2e y", tau);

```

```
// Result
// The upper limit of the age of the universe = 1.37
   e+010 y
```