Scilab Textbook Companion for Engineering Physics by D. C. Ghosh, N. C. Ghosh and P. K. Haldar[1]

Created by
Pankaj Biswas
Physics
Physics
Shri Mata Vaishno Devi University
College Teacher
Dr. Kamni
Cross-Checked by
Chaya

September 23, 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: Engineering Physics

Author: D. C. Ghosh, N. C. Ghosh and P. K. Haldar

Publisher: University Science, New Delhi

Edition: 1

Year: 2008

ISBN: 978-81-318-0366-0

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

Chapter 1

Classical Mechanics

Scilab code Exa 1.5 Force of contact between two masses

```
// Scilab Code Ex1.5: Page-11 (2008)
clc; clear;
m1 = 2;     // Mass of first body, kg
m2 = 1;     // Mass of second body, kg
F = 3;     // The horizontal force applied to the
    mass m1, N
F_prime = m2/(m1 + m2)*F;     // Force of contact
    between m1 and m2, N
printf("\nThe force of contact between m1 and m2 =
    %3.1f N", F_prime);
F_prime = m1/(m1 + m2)*F;     // Force of contact
    when F is applied to m2, N
printf("\nThe force of contact when F is applied to
    m2 = %3.1f N", F_prime);

// Result
// The force of contact between m1 and m2 = 1.0 N
// The force of contact when F is applied to m2 =
    2.0 N
```

Scilab code Exa 1.6 Direction of motion of a ball after momentum conservation during collision

```
// Scilab Code Ex1.6: Page-12 (2008)
clc; clear;
v = 1;      // Let the speed of the ball B be unity, unit
v_prime = v/2;      // Speed of the ball after the collision, unit
theta = atand(v_prime/v);      // The direction of motion of the ball A after collision, degree
printf("\nThe direction of motion of the ball after collision = %2.0f degree", theta);

// Result
// The direction of motion of the ball after collision = 27 degree
```

Scilab code Exa 1.9 Angular velocity of the combination of two wheels

```
// Scilab Code Ex1.9: Page-14 (2008)
clc; clear;
omega1 = 500;      // Angular speed of rotating shaft, r.p.m.
omega2 = 0;      // Initial angular speed of the second wheel, r.p.m.
I = 1;      // For simplicity assume moment of ineria of the wheels to be unity
I1 = I, I2 = I;      // Moment of inertia of wheels A and B, kg-Sq.m
// As I1*omega1 + I2*omega2 = (I1 + I2)*omega, solving for omega
```

```
omega = (I1*omega1 + I2*omega2)/(I1 + I2);    // Angular speed of the combination of two wheels, r.p.m.
printf("\nThe angular speed of the combination of two wheels = %3.0f r.p.m.", omega);

// Result
// The angular speed of the combination of two wheels = 250 r.p.m.
```

Scilab code Exa 1.10 Common velocity of a car truck system

```
// Scilab Code Ex1.10: : Page-14 (2008)
clc; clear;
m1 = 1200;    // Mass of the car, kg
m2 = 3600;    // Mass of the truck, kg
u1 = 30;    // Speed of the car, m/s
u2 = 20;    // Speed of the truck, m/s
theta = 60;    // Direction of motion of the truck w.r.t. that of car, degree
// As m1*u1 + m2*u2 = (m1 + m2)*v, solving for v along x and y directions
v_x = (m1*u1 + m2*u2*cosd(theta))/(m1 + m2);    // Common speed along x-direction, m/s
u1 = 0;    // The speed of the car after interlocking with the truck, m/s
v_y = (m1*u1 + m2*u2*sind(theta))/(m1 + m2);    // Common speed along y-direction, m/s
v = sqrt(v_x^2 + v_y^2);    // Common speed of the car-truck system, m/s
theta = atand(v_y/v_x);    // Direction of common velocity w.r.t. that of car, degree
printf("\nThe common speed of the car-truck system = %4.1f m/s", v);
printf("\nThe direction of common velocity = %4.1f
```

```
    degree north of east", theta);

// Result
// The common speed of the car-truck system = 19.8 m
    /s
```

Scilab code Exa 1.11 Velocity of third piece of the exploded object

```
// Scilab Code Ex1.11: Page-14 (2008)
clc; clear;
v1 = 20;    // Velocity of first piece, m/s
v2 = 30;    // Velocity of second piece, m/s
// From conservation of momentum, in x-direction
// m*v1*cosd(0)+m*v2*cosd(45)+m*v3*cosd(theta) = 0,
    solving for v3*cosd(theta)
v3_cos_theta = -(v1*cosd(0)+v2*cosd(45));    // x-
    component of v3 along theta, m/s
// From conservation of momentum, in y-direction
// m*v1*sind(0)-m*v2*sind(45)+m*v3*sind(theta) = 0,
    solving for v3*sind(theta)
v3_sin_theta = -(v1*sind(0)-v2*sind(45));    // y-
    component of v3 along theta, m/s
theta = atand(v3_sin_theta/v3_cos_theta);    //
    Direction of velocity of third piece, degree
v3 = -(v1*cosd(0)+v2*cosd(45))/cosd(theta+180);
    // Velocity of third piece, m/s
printf("\nThe velocity of third piece is %4.1f m/s
    towards %d degree north of west", v3, ceil(theta
    +180));

// Result
// The velocity of third piece is 46.4 m/s towards
    153 degree north of west
```

Chapter 2

Electricity and Magnetism

Scilab code Exa 2.12 Work done in moving a particle in force field

```
// Scilab Code Ex2.12: Page-80 (2008)
clc; clear;
t = poly(0, 't');
x = t^2 + 1;
y = 2*t^2;
z = t^3;
F = [3*x*y -5*z 10*x];    // Force acting on the
    particle, N
t1 = 1;    // lower limit
t2 = 2;    // upper limit
dr = [derivat(x); derivat(y); derivat(z)];    //
    Infinitesimal displacement, m
dW = F*dr;    // Work done or infinitesimally small
    displcement, J
work_exp = sci2exp(dW);    // Convert the polynomial
     to the expression
W = integrate(work_exp, 't', t1, t2);    // Total
    work done in moving the particle in a force field
    , J
printf("\nThe total work done in moving the particle
     in a force field = %d J", W);
```

```
// Result
```

Scilab code Exa 2.13 Evaluation of force integral

```
// Scilab Code Ex2.13: Page-80 (2008)
clc; clear;
x = poly(0, 'x');
y = x^2-4;
F = [x*y (x^2 + y^2)];     // Force acting on the
    particle, N
x1 = 2;     // lower limit
x2 = 4;     // upper limit
dr = [derivat(x); derivat(y);];     // Infinitesimal
    displacement, m
dW = F*dr;     // Work done or infinitesimally small
    displcement, J
work_exp = sci2exp(dW);     // Convert the polynomial
     to the expression
W = integrate(work_exp, 'x', x1, x2);     // Total
    work done in moving the particle in a force field
    , J
printf("\nThe total work done in moving the particle
     in the x-y plane = %d J", W);

// Result
// The total work done in moving the particle in the
     x-y plane = 732 J
```

Scilab code Exa 2.31 Electric flux through a surface area

```
// Scilab Code Ex2.31: Page-93 (2008)
clc; clear;
```

```
E = [3 4 8];     // Coefficients of i, j and k in the
    electric field, N/C
S = [0; 0; 100];     // Coefficients of i, j and k in
    the area vector, Sq. m
phi_E = E*S;     // Electric flux through the surface
    , N-Sq.m/C
printf("\nThe electric flux through the surface = %d
    N-Sq.m/C", phi_E);

// Result
// The electric flux through the surface = 800 N-Sq.
   m/C
```

Scilab code Exa 2.32 Electric flux through an area in XY plane

```
// Scilab Code Ex2.32: Page-93 (2008)
clc; clear;
E = [8 4 3];     // Coefficients of i, j and k in the
    electric field, N/C
S = [0; 0; 100];     // Coefficients of i, j and k in
    the area vector, Sq. m
phi_E = E*S;     // Electric flux through the surface
    , N-Sq.m/C
printf("\nThe electric flux through the area in XY
    plane = %d N-Sq.m/C", phi_E);

// Result
// The electric flux through the area in XY plane =
    300 N-Sq.m/C
```

Scilab code Exa 2.33 Electric flux through a surface in YZ plane

```
// Scilab Code Ex2.33: Page-93 (2008)
```

```
clc; clear;
E = [2 3 4];     // Coefficients of i, j and k in the electric field, N/C
S = [10; 0; 0];     // Coefficients of i, j and k in the area vector, Sq. m
phi_E = E*S;     // Electric flux through the surface, N-Sq.m/C
printf("\nThe electric flux through the surface in YZ plane = %d N-Sq.m/C", phi_E);

// Result
// The electric flux through the surface in YZ plane = 20 N-Sq.m/C
```

Scilab code Exa 2.39 Magnetic field due to a straight conductor carrying current

```
// Scilab Code Ex2.39: Page-96 (2008)
clc; clear;
mu_0 = 4*%pi*1e-007;     // Absolute magnetic permeability of free space, N/ampere-square
I = 15;     // Current through the wire, A
x = 1e-002;     // Distance of observation point from the wire, m
B = mu_0/(4*%pi)*2*I/x;     // Magnetic field at 1 cm distance, T
printf("\nThe magnetic field due to the current carrying wire at %d cm distance = %1.0e tesla", x/1e-002, B);
x = 5;     // Distance of observation point from the infinite straight conductor, m
I = 100;     // Current through the straight conductor, A
B = mu_0/(4*%pi)*2*I/x;     // Magnetic field at 1 cm distance, T
```

```
printf("\nThe magnetic field due to the current
    carrying infinite straight conductor at %d m
    distance = %1.0e tesla", x, B);

// Result
// The magnetic field due to the current carrying
    wire at 1 cm distance = 3e-004 tesla
// The magnetic field due to the current carrying
    infinite straight conductor at 5 m distance = 4e
    -006 tesla
```

Scilab code Exa 2.40 Force between two current carrying straight wires

```
// Scilab Code Ex2.40: Page-96 (2008)
clc; clear;
mu_0 = 4*%pi*1e-007;    // Absolute magnetic
    permeability of free space, N/ampere-square
I1 = 30;    // Current through the first wire, A
I2 = 40;    // Current through the second wire, A
x = 2;    // Separation distance between two wires,
    m
F = mu_0/(4*%pi)*2*I1*I2/x;    // Force between two
    current carrying straight wires, N
printf("\nThe force between two current carrying
    straight wires = %3.1e N", F);

// Result
// The force between two current carrying straight
    wires = 1.2e-004 N
```

Chapter 3

Vibration Waves and Light

Scilab code Exa 3.a.02 Frequency of particle executing SHM

```
// Scilab Code Ex3a.a.2: Page-132 (2008)
clc; clear;
m = 10;    // Mass of the particle, g
x = poly(0, 'x');
V = 50*x^2 + 100;    // Potential field surrounding
    the particle, erg/g
U = m*V;    // Potential energy of the particle
    field system, erg
F = -derivat(U);    // Force acting on the particle,
     dyne
// As F = -m*a = -m*omega^2*x = -m*(2%pi*f)^2*x,
    solving for f
f = sqrt(eval(pol2str(-pdiv(F,x)/m)))/(2*%pi);    //
     Frequency of oscillations of the particle
    executing SHM, Hz
printf("\nThe frequency of oscillations of the
    particle executing SHM = %4.2f Hz", f);

// Result
// The frequency of oscillations of the particle
    executing SHM = 1.59 Hz
```

Scilab code Exa 3.a.03 A body executing SHM

```
// Scilab Code Ex3a.a.3:Page-133 (2008)
clc; clear;
v1 = 80;    // Velocity of the body at 3 cm
    displacement, cm/s
v2 = 60;    // Velocity of the body at 4 cm
    displacement, cm/s
x1 = 3;    // Displacement of the body at velocity
    of 80 cm/s
x2 = 4;    // Displacement of the body at velocity
    of 60 cm/s
// As v = omega*sqrt(a^2 - x^2), solving for a
a = poly(0, 'a');
a = roots(v1^2*(a^2-16) - v2^2*(a^2 - 9));
omega = v1/sqrt(a(1)^2 - x1^2);    // Angular
    ferquency of the oscillations, rad/s
x = a(1);    // Maximum displacement, cm
// As x = a*sin(omega*t), solving for t
t_ex = asin(x/a(1))/omega;    // Time taken to reach
     the +ve extremity, s
d = a(1) - 2.5;    // Distance of the point from the
     mean position, cm
t = asin(d/a(1))/omega;    // Time taken to travel
    from mean position to positive extremity, s
printf("\nThe time taken to travel from 2.5 cm from
    +ve extremity = %5.3f s", t_ex - t);

// Result
// The time taken to travel from 2.5 cm from +ve
    extremity = 0.052 s
```

Scilab code Exa 3.a.04 Time period of a body oscillating in the tunnel across the earth

```
// Scilab Code Ex3a.a.4: Page-134 (2008)
clc; clear;
R = 6.4e+006;     // Radius of the earth, m
g = 10;     // Acceleration due to gravity, m/sec-
    square
T = 2*%pi*sqrt(R/g);     // Time period of
    oscillations of the body, s
printf("\nThe time period of oscillations of the
    body = %4.1f min", T/60);

// Result
// The time period of oscillations of the body =
    83.8 min
```

Scilab code Exa 3.a.05 Resultant amplitude and phase angle relative to the first SHM

```
// Scilab Code Ex3a.a.5: Page-135 (2008)
clc; clear;
phi1 = 0;     // Phase of the first SHM, degree
phi2 = 60;     // Phase of the second SHM, degree
phi3 = 90;     // Phase of the third SHM, degree
a1 = 1.0;     // Amplitude of the first SHM, cm
a2 = 1.5;     // Amplitude of the second SHM, cm
a3 = 2.0;     // Amplitude of the third SHM, cm
A = sqrt((a1 + a2*cosd(phi2)+a3*cosd(phi3))^2 + (a2*
    sind(phi2)+a3*sind(phi3))^2);     // Resultant
    amplitude relative to the first SHM, cm
phi = atand((a2*sind(phi2)+a3*sind(phi3))/(a1 + a2*
    cosd(phi2)+a3*cosd(phi3)));     // Resultant phase
     angle relative to the first SHM, degree
printf("\nThe resultant amplitude and phase angle
```

```
    relative to the first SHM = %4.2f cm and %2d
    degrees respectively", A, phi);

// Result
// The resultant amplitude and phase angle relative
    to the first SHM are 3.73 cm and 62 degrees
    respectively
```

Scilab code Exa 3.a.07 Two SHMs acting in the same direction

```
// Scilab Code Ex3a.a.7:Page-136 (2008)
clc; clear;
phi1 = 0;     // Phase of the first SHM, degree
phi2 = 45;     // Phase of the second SHM, degree
a1 = 0.005;     // Amplitude of the first SHM, m
a2 = 0.002;     // Amplitude of the second SHM, m
A = sqrt((a1 + a2*cosd(phi2))^2 + (a2*sind(phi2))^2)
    ;    // Resultant amplitude relative to the first
     SHM, m
phi = atand(a2*sind(phi2)/(a1 + a2*cosd(phi2)));
    // Resultant phase angle relative to the first
   SHM, degree
printf("\nThe amplitude of the resultant
    displacement and phase angle relative to the
    first SHM are %7.5f m and %5.2f degrees
    respectively", A, phi);

// Result
// The amplitude of the resultant displacement and
    phase angle relative to the first SHM are 0.00657
     m and 12.43 degrees respectively
```

Scilab code Exa 3.a.11 A spring disc system undergoing damped oscillation

```
// Scilab Code Ex3a.b.1: Page-138 (2008)
clc; clear;
m = 100;    // Mass of the horizontal disc, g
t = 60;    // Time during which the amplitude
    reduces to half of its undamped value, s
f = 10;    // Frequency of oscillations of the
    system, Hz
omega_prime = 2*%pi*f;    // Angular frequency of
    the oscillations, rad/s
A0 = 1;    // Assume the amplitude of undamped
    oscillations to be unity, cm
// As A = A0*exp(-k*t), solving for k
A = A0/2;    // Amplitude of damped oscillations
    after 1 min, cm
k = log(A0/A)/t;    // Resisting force per unit mass
     per unit velocity, nepers/sec
r = 2*k*m;    // Resistive force constant, sec/cm
tau = 1/k;    // Relaxation time, sec
Q = m*omega_prime/r;    // Quality factor
s = m*(omega_prime^2 + k^2);    // Force constant of
     the spring, dynes/Sq.cm
printf("\nThe resistive force constant = %4.2f dyne-
    sec/cm", r);
printf("\nThe relaxation time of the system = %4.2f
    sec", tau);
printf("\nThe quality factor, Q = %4.2f", Q);
printf("\nThe force constant of the spring = %4.2e
    dyne/Sq.cm", s);

// Result
// The resistive force constant = 2.31 dyne-sec/cm
// The relaxation time of the system = 86.56 sec
// The quality factor, Q = 2719.42
// The force constant of the spring = 3.95e+005 dyne
    /Sq.cm
```

Scilab code Exa 3.a.12 A mass executing damped oscillations in one dimension

```
// Scilab Code Ex3a.b.2: Page-139 (2008)
clc; clear;
function m = check_motion_type(k, omega0)
    if k > omega0 then
        m = 'aperiodic';
    else if  k == omega0 then
            m = 'criticallydamped';
        else if  k < omega0 then
                m = 'oscillatory';
            end
        end
    end
endfunction
m = 10;     // Mass of the body, g
s = 10;     // Restoring force, dyne/cm
r = 2;     // Resistive force constant, dyne.sec/cm
k = r/(2*m);     // Resisting force, nepers/sec
// As omega0^2 = s/m, solving for omega0
omega0 = sqrt(s/m);     // Angular frequency, rad/s
motion = check_motion_type(k, omega0);     // Check for the type of motion
r_new = 2*sqrt(m*s);     // Resistive force constant, dyne-sec/cm
m = r^2/(4*s);     // Mass for which the given forces makes the motion critically damped, g
printf("\nThe motion is %s in nature", motion);
printf("\nThe resistive force constant = %d dyne-sec/cm", r_new);
printf("\nThe mass for which the given forces makes the motion critically damped = %3.1f g", m);
```

```
// Result
// The motion is oscillatory in nature
// The resistive force constant = 20 dyne-sec/cm
// The mass for which the given forces makes the motion critically damped = 0.1 g
```

Scilab code Exa 3.a.14 A mass executing damped oscillations in one dimension

```
// Scilab Code Ex3a.b.4: Page-140 (2008)
clc; clear;
m = 1;     // Mass of the suspended body, kg
s = 25;     // Stifness constant of the spring, N/m
r = poly(0, 'r');
// As f0/f_prime = 2/sqrt(3), solving for r
r = roots(4*(s/m-r^2/(4*m^2))-3*s/m);     // Damping factor, kg/sec
printf("\nThe damping factor of damped oscillations = %d kg/sec", r(1));

// Result
// The damping factor of damped oscillations = 5 kg/sec
```

Scilab code Exa 3.a.15 Resisting force for critically damped motion

```

// Scilab Code Ex3a.b.5: Page-141 (2008)
clc; clear;
function m = check_motion_type(k, omega0)
    if k > omega0 then
        m = 'aperiodic';
    else if  k == omega0 then
```

```
                m = 'criticallydamped';
            else if  k < omega0 then
                    m = 'oscillatory';
                end
            end
        end
endfunction
m = 10;     // Mass of the oscillating body, g
r = 2;     // Resisting force, dyne-sec/cm
s = 5;     // Restoring force, dyne/cm
k = r/(2*m);     // Resisting force, nepers/sec
// As omega0^2 = s/m, solving for omega0
omega0 = sqrt(s/m);     // Angular frequency, rad/s
motion = check_motion_type(k, omega0);     // Check
    for the type of motion
r = 2*sqrt(m*s);     // Resistive force constant for
    critical damping, dyne-sec/cm
printf("\nThe motion is %s in nature", motion);
printf("\nThe resistive force constant for critical
    damping = %4.1f dyne-sec/cm", r);

// Result
// The motion is oscillatory in nature
// The resistive force constant for critical damping
    = 14.1 dyne-sec/cm
```

Scilab code Exa 3.a.16 Damped oscillatory motion

```
// Scilab Code Ex3a.b.6: Page-141 (2008)
clc; clear;
m = 0.1;     // Mass of the oscillating body, kg
t = 50;     // Time during which the energy of system
    decays to 1/e of its undamped value, s
s = 10;     // Spring constant, N/m
E0 = 1;     // Assume the energy of undamped
```

```
oscillations to be unity, erg
// As E = E0*exp(-k*t) and E/E0 = 1/e,  solving for k
E = E0/%e;     // Energy of damped oscillations after 50 sec, erg
k = log(E0/E)/t;     // Resisting force per unit mass per unit velocity, nepers/sec
p = m*k;     // A resistive force constant, N-s/m
omega0 = sqrt(s/m);     // Angular frequency in the absence of damping, rad/sec
omega_prime = sqrt(omega0^2 - k^2/4);     // Angular frequency when damping takes place, rad/sec
Q = omega_prime/k;     // Quality factor
printf("\nThe resistive force constant, p = %1.0e N-s/m", p);
printf("\nThe quality factor, Q = %d", ceil(Q));

// Result
// The resistive force constant, p = 2e-003 N-s/m
// The quality factor, Q = 500
```

Scilab code Exa 3.a.17 Damped simple harmonic motion

```
// Scilab Code Ex3a.b.7: Page-142 (2008)
clc; clear;
t = 10;     // Time during which the amplitude reduces to 1/10th of its undamped value, s
f = 200;     // Frequency of oscillations of the system, Hz
omega0 = 2*%pi*f;     // Angular frequency of the oscillations, rad/s
A0 = 1;     // Assume the amplitude of undamped oscillations to be unity, cm
// As A = A0*exp(-k*t), solving for k
A = A0/10;     // Amplitude of damped oscillations
```

```
after 10 sec, cm
k = log(A0/A)/t;    // Resisting force per unit mass per unit velocity, nepers/sec
tau = 1/(2*k);    // Relaxation time, sec
Q = omega0*tau;    // Quality factor
E0 = 1;    // Assume energy of undamped oscillations to be unity, erg
E = E0/10;    // Energy of damped oscillations after t sec, erg
// As E = E0*exp(-2*k*t), solving for t
t = 1/(2*k)*log(E0/E);    // Time during which the energy falls to 1/10 of its initial value, sec
printf("\nThe relaxation time = %4.2f sec", tau);
printf("\nThe quality factor, Q = %d", Q);
printf("\nThe time during which the energy falls to 1/10 of its initial value = %d sec", t);
printf("\nThe damping constant, k = %4.2f", k);

// Result
// The relaxation time = 2.17 sec
// The quality factor, Q = 2728
// The time during which the energy falls to 1/10 of its initial value = 5 sec
// The damping constant, k = 0.23
// The answer for Q is given wrongly in the textbook
```

Scilab code Exa 3.a.21 Characteristics of progressive waves

```
// Scilab Code Ex3a.c.1: Page-143 (2008)
clc; clear;
// Comparing with the standard progressive wave equation, we have
a = 0.5;    // Amplitude of the wave, m
lambda = 2*%pi/12.56;    // Wavelength of the wave, m
```

```
v = 314/12.56;     // Wave velocity , m/s
nu = v/lambda;     // Frequency of the wave , Hz
printf("\nThe amplitude of the wave = %3.1f m", a);
printf("\nThe wavelength of the wave = %3.1f m", lambda);
printf("\nThe velocity of the wave = %d m/s", v);
printf("\nThe frequency of the wave = %d Hz", ceil(nu));

// Result
// The amplitude of the wave = 0.5 m
// The wavelength of the wave = 0.5 m
// The velocity of the wave = 25 m/s
// The frequency of the wave = 50 Hz
```

Scilab code Exa 3.a.22 A simple harmonic wave travelling along X axis

```
// Scilab Code Ex3a.c.2: Page-144 (2008)
clc; clear;
// Comparing with the standard progressive wave equation , we have
a = 5;     // Amplitude of the wave , m
nu = 0.2;     // Frequency of the wave , Hz
lambda = 1/0.5;     // Wavelength of the wave , m
v = nu*lambda;     // Wave velocity , m/s
printf("\nThe amplitude of the wave = %3.1f m", a);
printf("\nThe wavelength of the wave = %3.1f m", lambda);
printf("\nThe velocity of the wave = %3.1f m/s", v);
printf("\nThe frequency of the wave = %3.1f Hz", nu);

// Result
// The amplitude of the wave = 5.0 m
// The wavelength of the wave = 2.0 m
```

```
// The velocity of the wave = 0.4 m/s
// The frequency of the wave = 0.2 Hz
```

Scilab code Exa 3.a.23 Travelling wave characteristics and phase difference

```
// Scilab Code Ex3a.c.3: Page-144 (2008)
clc; clear;
// Comparing with the standard progressive wave equation, we have
a = 8;      // Amplitude of the wave, cm
nu = 4/2;      // Frequency of the wave, Hz
lambda = 2/0.02;      // Wavelength of the wave, cm
v = nu*lambda;      // Wave velocity, cm/s
delta_x = 20;      // Path difference between two particles, cm
delta_phi = delta_x*2*%pi/lambda*180/%pi;      // Phase difference between two particles, degree
printf("\nThe amplitude of the wave = %3.1f cm", a);
printf("\nThe wavelength of the wave = %3.1f cm", lambda);
printf("\nThe velocity of the wave = %3.1f cm/s", v);
printf("\nThe frequency of the wave = %d Hz", nu);
printf("\nThe phase difference between two particles = %d degree", delta_phi);

// Result
// The amplitude of the wave = 8.0 cm
// The wavelength of the wave = 100.0 cm
// The velocity of the wave = 200.0 cm/s
// The frequency of the wave = 2 Hz
// The phase difference between two particles = 72 degree
```

Scilab code Exa 3.b.101 Brewster angle and angle of refraction for glass

```
// Scilab Code Ex3b.1: Page-163 (2008)
clc; clear;
mu = 1.5;     // Refractive indexof glass
i_p = atand(mu);     // Angle of polarization from
   Brewster's law, degree
r = 90 - i_p;     // Angale of refraction, degree
printf("\nThe Brewster angle for glass = %4.1f
   degree", i_p);
printf("\nThe angle of refraction for glass = %4.1f
   degree", r);

// Result
// The Brewster angle for glass = 56.3 degree
// The angle of refraction for glass = 33.7 degree
```

Scilab code Exa 3.b.102 Polarizing angles for various pair of media

```
// Scilab Code Ex3b.2: Page-163 (2008)
clc; clear;
// Function to convert degree to degree-minute
function [d,m]= deg2deg_min(deg)
    d = int(deg);
    m = (deg - d)*60;
endfunction
mu_air = 1;     // Refractive index fo air
mu_glass = 1.54;     // Refractive index of glass
mu_water = 1.33;     // Refractive index of water
// Air to glass incidence
i_p = atand(mu_glass/mu_air);     // Angle of
   polarization for air to glass incidence, degree
```

```
printf("\nFor air to glass, i_p = %d degree", i_p);
// glass to air incidence
i_p = atand(mu_air/mu_glass);    // Angle of polarization for glass to air incidence, degree
printf("\nFor glass to air, i_p = %d degree", ceil(i_p));
// Water to glass incidence
i_p = atand(mu_glass/mu_water);    // Angle of polarization for water to glass incidence, degree
[d,m] = deg2deg_min(i_p);    // Call function to convert to deg-min
printf("\nFor water to glass, i_p = %d degree %d min", d, m);
// Glass to water incidence
i_p = atand(mu_water/mu_glass);    // Angle of polarization for glass to water incidence, degree
[d,m] = deg2deg_min(i_p);    // Call function to convert to deg-min
printf("\nFor glass to water, i_p = %d degree %d min", d, m);
// Air to water incidence
i_p = atand(mu_water/mu_air);    // Angle of polarization for air to water incidence, degree
[d,m] = deg2deg_min(i_p);    // Call function to convert to deg-min
printf("\nFor air to water, i_p = %d degree %d min", d, m);
// Water to air incidence
i_p = atand(mu_air/mu_water);    // Angle of polarization for water to airincidence, degree
[d,m] = deg2deg_min(i_p);    // Call function to convert to deg-min
printf("\nFor water to air, i_p = %d degree %d min", d, m);

// Result
// For air to glass, i_p = 57 degree
// For glass to air, i_p = 33 degree
```

```
// For water to glass, i_p = 49 degree 11 min
// For glass to water, i_p = 40 degree 48 min
// For air to water, i_p = 53 degree 3 min
// For water to air, i_p = 36 degree 56 min
```

Scilab code Exa 3.b.103 Polarizing angle for glass

```
// Scilab Code Ex3b.3: Page-163 (2008)
clc; clear;
C = 40;    // Critical angle for glass to air
mu = 1/sind(C);    // Refractive index of glass w.r.
    t. air
i_p = atand(mu);    // Polarizing angle for glass,
    degree
printf("\nThe polarizing angle for glass = %4.1f
    degree", i_p);

// Result
// The polarizing angle for glass = 57.3 degree
```

Scilab code Exa 3.b.104 Polarization by reflection

```
// Scilab Code Ex3b.4: Page-164 (2008)
clc; clear;
i = 60;    // Angle of incidence, degree
i_p = i;    // Angle of polarization, degree
mu = tand(i_p);    // Refractive index of the medium
r = 90 - i;    // Angle of refraction, degree
printf("\nThe refractive index of transparent medium
     = %5.3f", mu);
printf("\nThe angle of refraction, r = %d degree", r
    );
```

```
printf("\nThe reflected and transmitted components
   are at right angles to each other.");

// Result
// The refractive index of transparent medium =
   1.732
// The angle of refraction , r = 30 degree
// The reflected and transmitted components are at
   right angles to each other.
```

Scilab code Exa 3.b.105 Intensity ratio of two beams through analyser

```
// Scilab Code Ex3b.5: Page-164 (2008)
clc; clear;
theta_A = 30;     // Angle between principal sections
    of polariser and  analyser for beam A, degree
theta_B = 60;     // Angle between principal sections
    of polariser and  analyser for beam B, degree
// As I_A*cosd(theta_A)^2 = I_B*cosd(theta_B)^2,
   solving for I ratio
I_ratio = cosd(theta_B)^2/cosd(theta_A)^2;     // The
    intensity ratio of the two beams
printf("\nThe intensity ratio of the two beams = %4
   .2f", I_ratio);

// Result
// The intensity ratio of the two beams = 0.33
```

Scilab code Exa 3.b.106 Percentage reduction in the intensity of the inident light

```
// Scilab Code Ex3b.6: Page-165 (2008)
clc; clear;
```

```
theta = [30 45 60 90];      // Angles between principal sections of polariser and analyser , degree
for i = 1:1:4
    P_red = (1-cosd(theta(i))^2)*100;     // Percentage reduction in intensity of incident light
    printf("\nFor theta = %d degree , percentage reduction = %1.0f percent", theta(i), P_red);
end

// Result
// For theta = 30 degree , percentage reduction = 25 percent
// For theta = 45 degree , percentage reduction = 50 percent
// For theta = 60 degree , percentage reduction = 75 percent
// For theta = 90 degree , percentage reduction = 100 percent
```

Scilab code Exa 3.b.107 Angle of rotation of polaroid to reduce the intensity

```
// Scilab Code Ex3b.7: Page-165 (2008)
clc; clear;
// For half reduction in intensity
I_ratio = 1/2;     // Intensity ratio
theta = acosd(sqrt(I_ratio));     // Angle of rotation of polaroid , degree
printf("\nFor half reduction in intensity , the angle of rotation = %d degree", theta);
// For one-fourth reduction in intensity
I_ratio = 1/4;     // Intensity ratio
theta = acosd(sqrt(I_ratio));     // Angle of
```

```
rotation of polaroid, degree
printf("\nFor one−fourth reduction in intensity, the angle of rotation = %d degree", theta);

// Result
// For half reduction in intensity, the angle of rotation = 45 degree
// For one−fourth reduction in intensity, the angle of rotation = 60 degree
```

Scilab code Exa 3.c.202 Ratio of maximum to minimum intensity in the interference fringe system

```
// Scilab Code Ex3c.2: Page−184 (2008)
clc; clear;
I2 = 1;     // Assume intensity of light beam from the second source to be unity
I1 = 81*I2;     // Intensity of light beam from the first source
a = sqrt(I1);     // Width of the first slit, mm
b = sqrt(I2);     // Width of the second slit, mm
I_max = (1+a/b)^2;     // Maximum intensity in the fringe pattern
I_min = (1-a/b)^2;     // Minimum intensity in the fringe pattern
fact = gcd([I_max,I_min]); // Find l.c.m. of I_max and I_min
printf("\nThe ratio of maximum to minimum intensity in the fringe system, I_max:I_min = %d:%d", I_max/4, I_min/4);

// Result
// The ratio of maximum to minimum intensity in the fringe system, I_max:I_min = 25:16
```

Scilab code Exa 3.c.203 Wavelength of light from interference of fringes

```
// Scilab Code Ex3c.3: Page-184 (2008)
clc; clear;
d = 0.1;    // Separation between the two slits, cm
D = 100;    // Distance between the source and the
    slit, cm
bita = 0.05;    // Fringe width, cm
lambda = bita*d/D;    // Wavelength of light, cm
printf("\nThe wavelength of light used = %4d
    angstrom", lambda/1e-008);

// Result
// The wavelength of light used = 5000 angstrom
```

Scilab code Exa 3.c.204 Fringe width from interference pattern

```
// Scilab Code Ex3c.4: Page-184 (2008)
clc; clear;
d = 0.3;    // Separation between the two slits, cm
D = 60;    // Distance between the source and the
    slit, cm
lambda = 59e-006;    // Wavelength of light, cm
bita = lambda*D/d;    // Fringe width, cm
printf("\nThe fringe width = %4.2e cm", bita);

// Result
// The fringe width = 1.18e-002 cm
```

Scilab code Exa 3.c.205 Distance between the two coherent sources

```
// Scilab Code Ex3c.5: Page-185 (2008)
clc; clear;
D = 80;      // Distance between the source and the
    slit, cm
lambda = 5890e-008;     // Wavelength of light, cm
bita = 9.424e-002;     // Fringe width, cm
d = lambda*D/bita;     // Separation between the two
    slits, cm
printf("\nThe distance between the two coherent
    sources = %4.2f cm", d);

// Result
// The distance between the two coherent sources =
    0.05 cm
```

Scilab code Exa 3.c.206 Distance between consecutive interference bands

```
// Scilab Code Ex3c.6: Page-185 (2008)
clc; clear;
D = 100;     // Distance between the source and the
    slit, cm
lambda = 5893e-008;     // Wavelength of light, cm
d1 = 4.05e-001;     // Distance between the images of
     the two slits in one position, cm
d2 = 2.90e-001;     // Distance between the images of
     the two slits in second position, cm
d = sqrt(d1*d2);     // Separation between the two
    slits, cm
bita = lambda*D/d;     // Fringe width, cm
printf("\nThe distance between consecutive
    interference bands = %6.4f cm", bita);

// Result
```

```
// The distance between consecutive interference bands = 0.0172 cm
```

Scilab code Exa 3.c.207 Wavelength of the light used in biprism experiment

```
// Scilab Code Ex3c.7: Page-185 (2008)
clc; clear;
D = 1.2;    // Distance between the source and the slit, m
d = 7.5e-004;    // Separation between the two slits, cm
n = 20;    // Number of fringes crossed in the field of view
bita = 1.888e-002/n;    // Fringe width, cm
lambda = bita*d/D;    // Wavelength of light, cm
printf("\nThe wavelength of the light used in biprism experiment = %4d angstrom", lambda/1e-010);

// Result
// The wavelength of the light used in biprism experiment = 5900 angstrom
```

Scilab code Exa 3.c.208 Number of fringes obtained with the given wavelength

```
// Scilab Code Ex3c.8: Page-186 (2008)
clc; clear;
lambda1 = 5893;    // First wavelength of light, angstrom
lambda2 = 4358;    // Second wavelength of light, angstrom
```

```
n = 40;    // Number of fringes obtained with first
    wavelength
// As bita1/bita2 = lambda1/lambda2, so
x = n*lambda1/lambda2;    // Number of fringes
    obtained with the seocond wavelength
printf("\nThe number of fringes obtained with the
    given wavelength = %d", x);

// Result
// The number of fringes obtained with the given
    wavelength = 54
```

Scilab code Exa 3.c.209 Wavelength of light from biprism interference pattern

```
// Scilab Code Ex3c.9: Page-186 (2008)
clc; clear;
D = 100;    // Distance between the source and the
    slit, cm
bita = 0.0135;    // Fringe width, cm
alpha = %pi/360;    // Angle of refracting face with
     the base of biprism, radian
mu = 1.5;    // Refractive index of the material of
    biprism
x = 50;    // Distance between slit and the biprism,
     cm
d = 2*(mu-1)*x*alpha;    // Separation between the
    two virtual slits, cm
lambda = bita*d/D;    // Wavelength of light, cm
printf("\nThe wavelength of light from biprism
    interference pattern = %4d angstrom", lambda/1e
    -008);

// Result
// The wavelength of light from biprism interference
```

```
pattern = 5890 angstrom
```

Scilab code Exa 3.c.210 Fringe width observed at one metre distance from biprism

```
// Scilab Code Ex3c.10: Page-187 (2008)
clc; clear;
mu = 1.5;     // Refractive index of the material of
    biprism
alpha = %pi/180;     // Base angle of biprism, radian
D = 110;     // Distance between the source and the
    slit, cm
x = 10;     // Distance between slit and the biprism,
    cm
d = 2*(mu-1)*x*alpha;    // Separation between the
    two virtual slits, cm
lambda = 5900e-008;     // Wavelength of light, cm
bita = lambda*D/d;     // Fringe width, cm
printf("\nThe fringe width observed at one metre
    distance from biprism = %6.4f cm", bita);

// Result
// The fringe width observed at one metre distance
    from biprism = 0.0372 cm
```

Scilab code Exa 3.c.211 Wavelength of light in Newton ring experiment

```
// Scilab Code Ex3c.11: Page-187 (2008)
clc; clear;
D_n = 0.42;     // Diameter of nth ring, cm
D_mplusn = 0.7;     // Diameter of (m+n)th ring, cm
m = 14;     // Difference between (m+n)th and nth
    rings
```

```
R = 100;     // Radius of curvature of the plano-
    convex lens, m
lambda = (D_mplusn^2 - D_n^2)/(4*m*R);     //
    Wavelength of the light, cm
printf("\nThe wavelength of the light used = %4d
    angstrom", lambda/1e-008);

// Result
// The wavelength of the light used = 5600 angstrom
```

Scilab code Exa 3.c.212 Radius of plano convex lens

```
// Scilab Code Ex3c.12: Page-187 (2008)
clc; clear;
D5 = 0.336;     // Diameter of 5th ring, cm
D10plus5 = 0.590;     // Diameter of (10+5)th ring,
    cm
m = 10;     // Difference between (10+5)th and 5th
    rings
lambda = 5890e-008;     // Wavelength of the light,
    cm
R = (D10plus5^2 - D5^2)/(4*m*lambda);     // Radius
    of curvature of the plano-convex lens, m
printf("\nThe radius of plano convex lens = %5.2f cm
    ", R);

// Result
// The radius of plano convex lens = 99.83 cm
```

Scilab code Exa 3.c.213 Wavelenght of light used in obtaining Newton rings

```
// Scilab Code Ex3c.13: Page-187 (2008)
```

```
clc; clear;
D3 = 0.181;     // Diameter of 3rd ring, cm
D23 = 0.501;     // Diameter of 23rd ring, cm
m = 23-3;     // Difference between (m+n)th and nth rings
R = 50;     // Radius of curvature of the plano-convex lens, m
lambda = (D23^2 - D3^2)/(4*m*R);     // Wavelength of the light, cm
printf("\nThe wavelength of the light used = %4d angstrom", lambda/1e-008);

// Result
// The wavelength of the light used = 5456 angstrom
```

Scilab code Exa 3.c.214 Diameter of the 20th dark ring

```
// Scilab Code Ex3c.14: Page-188 (2008)
clc; clear;
D4 = 0.4;     // Diameter of 4th ring, cm
D12 = 0.7;     // Diameter of 12th ring, cm
m = 12-4;     // Difference between (m+n)th and nth rings
lambda_R = (D12^2 - D4^2)/(4*m);     // Wavelength-Radius product, Sq.cm
D20 = sqrt(80*lambda_R);     // Diameter of the 20th dark ring, cm
printf("\nThe diameter of the 20th dark ring = %5.3f cm", D20);

// Result
// The diameter of the 20th dark ring = 0.908 cm
```

Scilab code Exa 3.c.215 Radius of curvature of the lens and the thickness of the air film

```
// Scilab Code Ex3c.15: Page-188 (2008)
clc; clear;
D10 = 0.50;     // Diameter of 10th ring, cm
n = 10;     // Number of dark fringe
lambda = 6250e-008;     // Wavelength of light used,
   cm
R = D10^2/(4*n*lambda);     // Radius of curvature of
    the lens, cm
t = D10^2/(8*R);     // Thickness of the air film, cm
printf("\nThe radius of curvature of the lens = %3d
   cm", R);
printf("\nThe thickness of the air film = %9.7f cm",
    t);

// Result
// The radius of curvature of the lens = 100 cm
// The thickness of the air film = 0.0003125 cm
```

Scilab code Exa 3.c.216 Newton rings observed in reflected light

```
// Scilab Code Ex3c.16: Page-188 (2008)
clc; clear;
D10 = 5e-003;     // Diameter of 10th ring, cm
n = 10;     // Number of dark fringe
lambda = 5.9e-007;     // Wavelength of reflected
   light, m
R = D10^2/(4*n*lambda);     // Radius of curvature of
    the lens, cm
t = D10^2/(8*R);     // Thickness of the air film, cm
printf("\nThe radius of curvature of the lens = %5.3
   f m", R);
printf("\nThe thickness of the air film = %4.2e m",
```

```
t);

// Result
// The radius of curvature of the lens = 1.059 m
// The thickness of the air film = 2.95e-006 m
```

Scilab code Exa 3.c.217 Smallest thickness of the glass film in which it appears dark

```
// Scilab Code Ex3c.17: Page-189 (2008)
clc; clear;
lambda = 5893e-010;    // Wavelength of light used,
    m
mu = 1.5;    // Refractive index of glass film
r = 60;    // Angle of reflection in the film,
    degree
t = lambda/(2*mu*cosd(r));    // Smallest thickness
    of the
printf("\nThe smallest thickness of the glass film
    when it appears dark = %6.1f angstrom", t/1e-010)
    ;

// Result
// The smallest thickness of the glass film when it
    appears dark = 3928.7 angstrom
```

Scilab code Exa 3.d.301 Wavelength of light used in diffraction due to narrow slit

```
// Scilab Code Ex3d.1: Page-205 (2008)
clc; clear;
D = 200;    // Distance between the source and the
    slit, cm
```

```
a = 0.02;     // Slit width, cm
x = 0.5;     // Position of first minimum, cm
n = 1;     // Order of diffraction
lambda = a*x/(D*n);     // Wavelength of light used, cm
printf("\nThe wavelength of light used = %4d angstrom", lambda/1e-008);

// Result
// The wavelength of light used = 5000 angstrom
```

Scilab code Exa 3.d.302 Separation between the second minima on either side of the central maximum

```
// Scilab Code Ex3d.2: Page-205 (2008)
clc; clear;
f = 20;     // Focal length of the lens, cm
a = 0.06;     // Slit width, cm
n = 2;     // Order of diffraction
lambda = 6e-005;     // Wavelength of light used, cm
x = 2*lambda*f/a;     // Separation between the second minima on either side of the central maximum, cm
printf("\nThe separation between the second minimum an central maximum = %4.2f cm", x);

// Result
// The separation between the second minimum an central maximum = 0.04 cm
```

Scilab code Exa 3.d.303 Distance of the first dark band from the axis

```
// Scilab Code Ex3d.3: Page-206 (2008)
```

```
clc; clear;
n = 1;     // Order of diffraction
f = 40;     // Focal length of the lens, cm
a = 0.03;     // Slit width, cm
lambda = 5890e-008;     // Wavelength of the light
    used, cm
// As a*sind(theta) = n*lambda, solving for theta
theta = asin(n*lambda/a);     // The angle of
    diffraction corresponding to the first minimum,
    radian
x = f*theta;     // The distance of the first dark
    band from the axis, cm
printf("\nThe distance of the first dark band from
    the axis = %6.4f cm", x);

// Result
// The distance of the first dark band from the axis
    = 0.0785 cm
```

Scilab code Exa 3.d.304 Angle of diffraction for the principal maxima

```
// Scilab Code Ex3d.4: Page-206 (2008)
clc; clear;
lambda1 = 5890e-008;     // Wavelength of D1 line of
    sodium lamp, cm
lambda2 = 5896e-008;     // Wavelength of D2 line of
    sodium lamp, cm
d_lambda = lambda2 - lambda1;     // Wavelength
    difference, cm
w = 0.5;     // Width of the grating, cm
N = 2500;     // Total number of grating lines
N_prime = N/w;     // Number of lines per cm, lines/
    cm
a_plus_b = 1/N_prime;     // Grating element, cm
n = 1;     // Order of diffraction
```

```
// Case 1
theta = asind(n*lambda1/a_plus_b);    // Angle of
    diffraction for D1 line, degree
// Case 2
theta_prime = asind(n*lambda2/a_plus_b);    // Angle
    of diffraction for D2 line, degree
printf("\nThe angle of diffraction for D1 and D2
    lines of sodium are %5.2f dgree and %5.2f degree
    respectively.", theta, theta_prime);
// From the condition for just resolution, lambda/
    d_lambda = n*N, solving for N
N_min = lambda1/(d_lambda*n);    // Minimum number
    of lines required on the grating
if N_min < N then
    printf("\nThe two lines are well resolved.");
else
    printf("\nThe two lines are not resolved.");
end

// Result
// The angle of diffraction for D1 and D2 lines of
    sodium are 17.13 dgree and 17.15 degree
    respectively.
// The two lines are well resolved.
```

Scilab code Exa 3.d.305 Wavelength of the spectral line

```
// Scilab Code Ex3d.5: Page-207 (2008)
clc; clear;
N = 4250;    // Number of lines per cm of grating,
    lines/cm
a_plus_b = 1/N;    // Grating element, cm
n = 2;    // Order of diffraction
theta = 30;    // Angle of diffraction, degree
lambda = sind(theta)*a_plus_b/n;    // Wavelength of
```

```
    spectral line from diffraction condition , cm
printf("\nThe wavelength of spectral line from
    diffraction condition = %4d angstrom", lambda/1e
    -008);

// Result
// The wavelength of spectral line from diffraction
    condition = 5882 angstrom
```

Scilab code Exa 3.d.306 Number of lines in one centimeter of the grating surface

```
// Scilab Code Ex3d.6: Page-207 (2008)
clc; clear;
n = 2;     // Order of diffraction
lambda = 5e-005;     // Wavelength of light , cm
theta = 30;     // Angle of diffraction , degree
N = sind(theta)/(n*lambda);     // Number of lines
    per cm of grating , lines/cm
printf("\nThe number of lines per cm of grating =
    %4d per cm", ceil(N));

// Result
// The number of lines per cm of grating = 5000 per
    cm
```

Scilab code Exa 3.d.307 Highest order spectrum obtainable with the given diffraction grating

```
// Scilab Code Ex3d.7: Page-208 (2008)
clc; clear;
N = 5000;     // Number of lines per cm ruled on
    grating , lines/cm
```

```
lambda = 6e-005;    // Wavelength of light , m
a_plus_b = 1/N;    // Grating element , m
theta = 90;    // Maximum angle of diffraction ,
    degree
n = a_plus_b*sind(theta)/lambda;    // Order of
    diffraction
printf("\nIn highest order spectrum obtainable with
    the given diffraction grating = %4.2f", n);

// Result
// In highest order spectrum obtainable with the
    given diffraction grating = 3.33
```

Scilab code Exa 3.d.308 Invisible third and higher order principal maxima in a diffraction grating

```
// Scilab Code Ex3d.8: Page-208 (2008)
clc; clear;
lambda = 5.5e-007;    // Wavelength of light , m
a_plus_b = 1.5e-006;    // Grating element , m
theta = 90;    // Maximum angle of diffraction ,
    degree
n = a_plus_b*sind(theta)/lambda;    // Order of
    diffraction
printf("\nIn this diffraction grating only %dnd
    order will be visible while %drd and higher
    orders are not possible.", n, n+1);

// Result
// In this diffraction grating only 2nd order will
    be visible while 3rd and higher orders are not
    possible.
```

Scilab code Exa 3.d.309 Number of lines per cm on the grating

```
// Scilab Code Ex3d.9: Page-208 (2008)
clc; clear;
theta = 30;     // Maximum angle of diffraction ,
    degree
lambda1 = 5400e-010;     // Wavelength of light
    giving certain diffraction order , m
lambda2 = 4050e-010;     // Wavelength of light
    giving higher diffraction order , m
n = poly(0, 'n');
n = roots(lambda1*n-(n+1)*lambda2);     // Order of
    diffraction for first wavelength
a_plus_b = n*lambda1/sind(theta);     // Grating
    element , m
N = 1/a_plus_b;     // Number of lines per cm ruled
    on grating , lines/cm
printf("\nThe number of lines per cm on the
    diffraction grating = %d lines per cm", N/100);

// Result
// The number of lines per cm on the diffraction
    grating = 3086 lines per cm
```

Scilab code Exa 3.d.310 Minimum number of lines on the diffraction grating

```
// Scilab Code Ex3d.10: Page-209 (2008)
clc; clear;
lambda = 5890e-008;     // Wavelength of light , cm
n = 1;     // Order of diffraction
d_lambda = 6e-008;     // Difference in wavelengths
    of D1 and D2 lines , cm
N = lambda/(n*d_lambda);     // Number of lines on
    grating
```

```
printf("\nThe minimum number of lines on the diffraction grating = %d", ceil(N));

// Result
// The minimum number of lines on the diffraction grating = 982
```

Scilab code Exa 3.d.311 Design of a plane transmission diffraction grating

```
// Scilab Code Ex3d.11: Page-209 (2008)
clc; clear;
lambda = 6000e-008;     // Wavelength of light, cm
n = 2;     // Order of diffraction
d_lambda = 6e-008;     // Difference in wavelengths of D1 and D2 lines, cm
N = lambda/(n*d_lambda);     // Number of lines on grating
printf("\nThe minimum number of lines in the required diffraction grating = %d", N);

// Result
// The minimum number of lines in the required diffraction grating = 500
```

Scilab code Exa 3.d.312 Minimum number of lines per cm in grating to just resolve the D1 and D2 lines of sodium

```
// Scilab Code Ex3d.12: Page-209 (2008)
clc; clear;
lambda = 5890e-008;     // Wavelength of light, cm
n = 2;     // Order of diffraction
```

```
d_lambda = 6e-008;    // Difference in wavelengths
   of D1 and D2 lines, cm
w = 2.5;    // Width of the grating, cm
N = lambda/(n*d_lambda);    // Number of lines on
   grating
printf("\nThe minimum number of lines per cm in the
   diffraction grating = %5.1f", N/w);

// Result
// The minimum number of lines per cm in the
   diffraction grating = 196.3
```

Scilab code Exa 3.d.313 Maximum resolving power of a plane transmission grating

```
// Scilab Code Ex3d.13: Page-210 (2008)
clc; clear;
lambda = 5000e-008;    // Wavelength of light, cm
theta = 90;    // Angle of diffraction for the
   maximum resolving power, degree
N = 40000;    // Number of lines on grating
a_plus_b = 12.5e-005;    // Grating element, cm
n = 2;    // Order of diffraction
n_max = N*a_plus_b*sind(theta)/lambda;    // Maximum
   resolving power
printf("\nThe maximum resolving power = %d", n_max);

// Result
// The maximum resolving power = 100000
```

Scilab code Exa 3.d.314 Maximum number of lines of a grating

```
// Scilab Code Ex3d.14: Page-209 (2008)
```

```
clc; clear;
lambda = 5890e-008;     // Wavelength of light , cm
n = 3;     // Order of diffraction
d_lambda = 6e-008;     // Difference in wavelengths of D1 and D2 lines , cm
N = lambda/(n*d_lambda);     // Maximum number of lines of a grating
printf("\nThe maximum number of lines of the grating = %d", N);

// Result
// The maximum number of lines of the grating = 327
```

Chapter 4

Special Theory of Relativity

Scilab code Exa 4.1 Fringe shift in the Michelson Morley experiment

```
// Scilab Code Ex4.1: Page-233 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
v = 3e+004;     // Speed of earth, m/s
d = 7;     // Effective length of each path, m
lambda = 7000e-010;     // Wavelength of light used, m
n = 2*d*v^2/(lambda*c^2);     // Fringe shift
printf("\nThe expected fringe shift = %3.1f", n);

// Result
// The expected fringe shift = 0.2
```

Scilab code Exa 4.2 Apparent length of rod relative to the observer

```
// Scilab Code Ex4.2: Page-233 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
```

```
v = 3e+007;     // Speed of metre rod, m/s
L0 = 1;     // Actual length of the rod, m
L = L0*sqrt(1-v^2/c^2);     // Apparent length of rod
    from Lorentz transformation, m
printf("\nThe apparent length of rod realtive to the
    observer = %5.3f m", L);

// Result
// The apparent length of rod realtive to the
   observer = 0.995 m
```

Scilab code Exa 4.3 Apparent length of a meter stick for different speeds

```
// Scilab Code Ex4.3: Page-234 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
v = [c c/sqrt(2) sqrt(3)/2*c c/2 0.8*c];     //
   Different speeds of metre rod, m/s
L0 = 100;     // Actual length of the rod, cm
for i = 1:1:5
    L = L0*sqrt(1-v(i)^2/c^2);     // Apparent length
        of rod from Lorentz transformation, m
    printf("\nFor v = %4.2e m/s, L = %4.1f cm", v(i)
        , L);
end

// Result
// For v = 3.00e+008 m/s, L =  0.0 cm
// For v = 2.12e+008 m/s, L = 70.7 cm
// For v = 2.60e+008 m/s, L = 50.0 cm
// For v = 1.50e+008 m/s, L = 86.6 cm
// For v = 2.40e+008 m/s, L = 60.0 cm
```

Scilab code Exa 4.4 Lorentz transformations applied to a rigid bar

```
// Scilab Code Ex4.4: Page-235-236 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
// Part (a)
v = 0.98*c ;     // Speed of the rigid bar, m/s
L2 = 1.5;     // Length of the rigid bar in S_prime frame, m
L1 = L2*sqrt(1-v^2/c^2);     // Apparent length of rod from Lorentz transformation, m
theta2 = 45;     // Angle which the bar makes w.r.t. x-aixs in S_prime frame, degree
theta1 = atand(tand(theta2)/sqrt(1-v^2/c^2));     // Orientation of bar relative to S frame, degree
printf("\nThe orientation of the %d m bar relative to S frame = %4.1f degree", L2, theta1);
// Part(b)
v = 0.6*c ;     // Speed of the rigid bar, m/s
L2 = 5;     // Length of the rigid bar in S_prime frame, m
L1 = L2*sqrt(1-v^2/c^2);     // Apparent length of rod from Lorentz transformation, m
theta2 = 30;     // Angle which the bar makes w.r.t. x-aixs in S_prime frame, degree
theta1 = atand(tand(theta2)/sqrt(1-v^2/c^2));     // Orientation of bar relative to S frame, degree
printf("\nThe orientation of the %d m bar relative to S frame = %4.1f degree", L2, theta1);

// Result
// The orientation of the 1 m bar relative to S frame = 78.7 degree
// The orientation of the 5 m bar relative to S frame = 35.8 degree
```

Scilab code Exa 4.5 Velocity of pi meson

```
// Scilab Code Ex4.5: Page-236 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
t0 = 2.5e-008;     // Proper life time of pi-meson, s
t = 2.5e-007;     // MEan life time of pi-meson, s
// As t = t0/(sqrt(1-v^2/c^2)), solving for v
v = sqrt(1-(t0/t)^2)*c;     // Velocity of pi meson, m/s
printf("\nThe velocity of pi meson = %5.3f c = %4.2e m/s", v/c, v);

// Result
// The velocity of pi meson = 0.995 c = 2.98e+008 m/s
```

Scilab code Exa 4.6 Relative speed of the ships as measured by an observer

```
// Scilab Code Ex4.6: Page-237 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
v = 0.8*c;     // Speed of the first spaceship, m/s
u_prime = 0.9*c;     // Speed of the second spaceship, m/s
u = (u_prime+v)/(1+u_prime*v/c^2);     // Relative speed of the ships as measured by the observer on either one from Velocity addition rule, m/s
printf("\nThe relative speed of the ships as measured by an observer in either one = %5.3f c = %4.2e m/s", u/c, u);
```

```
// Result
// The relative speed of the ships as measured by an observer in either one = 0.988 c = 2.97e+008 m/s
```

Scilab code Exa 4.7 Velocity of one particle relative to the other

```
// Scilab Code Ex4.7: Page-237 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
v = 0.9*c;     // Speed of the first particle, m/s
u_prime = 0.9*c;     // Speed of the oppositely moving second particle, m/s
u = (u_prime+v)/(1+u_prime*v/c^2);     // Velocity of one particle relative to the other from Velocity addition rule, m/s
printf("\nThe velocity of one particle relative to the other = %5.3f c = %4.2e m/s", u/c, u);

// Result
// The velocity of one particle relative to the other = 0.994 c = 2.98e+008 m/s
```

Scilab code Exa 4.8 Velocity of the rocket as observed from the earth

```
// Scilab Code Ex4.8: Page-237 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
// Case 1: when velocity of firing is away from the earth
v = 0.5*c;     // Speed of the rocket away from the earth, m/s
```

```
u_prime = 0.8*c;    // Speed of the outgoing
   spaceship relative to earth , m/s
u = (u_prime+v)/(1+u_prime*v/c^2);    // Velocity of
    rocket moving away relative to the earth , m/s
printf("\nThe velocity of rocket moving away
   relative to the earth = %4.2f c = %4.2e m/s", u/c
   , u);
// Case 2: when velocity of firing is towards the
   earth
v = 0.5*c;    // Speed of the rocket moving towards
   the earth , m/s
u_prime = -0.8*c;    // Speed of the outgoing
   spaceship relative to earth , m/s
u = (u_prime+v)/(1+u_prime*v/c^2);    // Velocity of
    approaching rocket relative to the earth , m/s
printf("\nThe velocity of approaching rocket
   relative to the earth = %3.1f c = %3.1e m/s", u/c
   , u);

// Result
// The velocity of rocket moving away relative to
   the earth = 0.93 c = 2.79e+008 m/s
// The velocity of approaching rocket relative to
   the earth = -0.5 c = -1.5e+008 m/s
```

Scilab code Exa 4.9 Velocity of the particle when its total energy is thrice its rest energy

```
// Scilab Code Ex4.9: Page-237 (2008)
clc; clear;
c = 3e+008;    // Speed of light in vacuum, m/s
E0 = 1;    // Assume the rest energy of the particle
    to be unity
E = 3*E0;    // Total energy of the particle
v = sqrt(1-(E0/E)^2)*c;    // Velocity of the
```

```
particle from relativistic variation of mass with
 speed , m/s
printf("\nThe velocity of the particle when its
total energy is thrice its rest energy = %5.3e cm
/s", v);

// Result
// The velocity of the particle when its total
energy is thrice its rest energy = 2.828e+008 cm/
s
```

Scilab code Exa 4.10 Relativisti variation of mass of electron with velocity

```
// Scilab Code Ex4.10: Page-238 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
m0 = 9.1e-031;     // Rest mass of the electron , kg
E0 = m0*c^2;     // Rest energy of the electron , J
printf("\nThe rest energy of the electron = %4.2f
 MeV", E0/1.6e-013);
E = 1.25*E0;     // Total energy of the particle
v = sqrt(1-(E0/E)^2)*c;     // Velocity of the
 particle from relativistic variation of mass with
  speed , m/s
printf("\nThe velocity of the electron when its
 total energy is 1.25 times its rest energy = %3.1
 f c = %3.1e cm/s", v/c, v);

// Result
// The rest energy of the electron = 0.51 MeV
// The velocity of the electron when its total
energy is 1.25 times its rest energy = 0.6 c =
1.8e+008 cm/s
```

Scilab code Exa 4.11 An electron subjected to relativistic motion

```
// Scilab Code Ex4.11: Page-238 (2008)
clc; clear;
c = 3e+008;     // Speed of light in vacuum, m/s
v = 0.99*c;     // Speed of the electron, m/s
m0 = 9.1e-031;     // Rest mass of the electron, kg
m = m0/sqrt(1-v^2/c^2);     // Moving mass of the
    electron, kg
E = m*c^2;     // Total energy of the electron, J
printf("\nThe total energy of the electron = %4.2e J
    ", E);
KE_ratio = m0/(2*(m-m0))*(v/c)^2;     // Ratio of
    Newtonian kinetic energy to the relativistic
    kinetic energy
printf("\nThe ratio of Newtonian kinetic energy to
    the relativistic kinetic energy = %4.2f",
    KE_ratio);

// Result
// The total energy of the electron = 5.81e-013 J
// The ratio of Newtonian kinetic energy to the
    relativistic kinetic energy = 0.08
```

Chapter 5

Quantum Mechanics

Scilab code Exa 5.2 Temperature of the surface of sun

```
// Scilab Code Ex5.2: Page-284 (2008)
clc; clear;
lambda_m = 4753e-010;     // Wavelength from the sun
    at which maximum energy is emitted, m
b = 2.88e-003;     // Wein's constant, m-K
T = b/lambda_m;     // Temperature of the surface of
    sun
printf("\nThe temperature of the surface of sun = %d
    K", ceil(T));

// Result
// The temperature of the surface of sun = 6060 K
```

Scilab code Exa 5.3 Wavelength of maximum intensity of radiation

```
// Scilab Code Ex5.3: Page-284 (2008)
clc; clear;
b = 2.898e-003;     // Wein's constant, m-K
```

```
T = 3000 + 273;     // Temperature of the source , K
lambda_m = b/T;     // Wavelength of maximum
    intensity of radiation emitted from the source , m
printf("\nThe wavelength of maximum intensity of
    radiation emitted from the source = %d angstrom",
     lambda_m/1e-010);

// Result
// The wavelength of maximum intensity of radiation
    emitted from the source = 8854 angstrom
```

Scilab code Exa 5.4 Kinetic energy of the ejected photoelectrons

```
// Scilab Code Ex5.4: Page-285 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant , Js
c = 3e+008;     // Speed of light , m/s
lambda = 2300e-010;     // Thereshold wavelength for
    tungsten , m
phi = h*c/lambda;     // Work function for tungsten ,
    J
lambda = 1800e-010;     // Wavelength of incident
    radiation , m
E = h*c/lambda;     // Energy of the incidnt
    radiation , J
KE = E - phi;     // Kinetic energy of the ejected
    photoelectrons , J
printf("\nThe kinetic energy of the ejected
    photoelectrons = %3.1f eV", KE/1.6e-019);

// Result
// The kinetic energy of the ejected photoelectrons
    = 1.5 eV
```

Scilab code Exa 5.5 Possibility of electron emission with the given incident wavelengths

```
// Scilab Code Ex5.5: Page-285 (2008)
clc; clear;
function [] = check_energy(E, L)
phi = 4.8;      // Work function for tungsten, eV
    if E > phi then
        printf("\nThe wavelength %d angstrom will be
            able to liberate an electron.", ceil(L/1
           e-010));
    else
        printf("\nThe wavelength %d angstrom will
           not be able to liberate an electron.",
           ceil(L/1e-010));
    end
endfunction
h = 6.62e-034;      // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
// Case 1
lambda = 2000e-010;     // Wavelength of incident
    radiation, m
E = h*c/(lambda*1.6e-019);      // Energy of the
    incidnt radiation, eV
check_energy(E, lambda);      // Check for the
    wavelength
// Case 2
lambda = 5000e-010;     // Wavelength of incident
    radiation, m
E = h*c/(lambda*1.6e-019);      // Energy of the
    incidnt radiation, eV
check_energy(E, lambda);      // Check for the
    wavelength
```

```
// Result
// The wavelength 2000 angstrom will be able to
   liberate an electron.
// The wavelength 5000 angstrom will not be able to
   liberate an electron.
```

Scilab code Exa 5.6 Velocity of emitted photoelectrons

```
// Scilab Code Ex5.6: Page-286 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;     // Energy quivalent of 1 eV, J
phi = 2.28*e;     // Work function for material, J
m = 9.1e-031;     // Mass of an electron, kg
lambda = 3000e-010;     // Wavelength of incident
   radiation, m
E = h*c/lambda;     // Energy of the incidnt
   radiation, J
KE = E - phi;     // Kinetic energy of the ejected
   photoelectrons, J
v = sqrt(2*KE/m);     // Velocity of emitted electron
   , m/s
printf("\nThe velocity of the emitted electron = %4
   .2e m/s", v);

// Result
// The velocity of the emitted electron = 8.08e+005
   m/s
```

Scilab code Exa 5.7 A photosensitive material emitting photoelectrons

```
// Scilab Code Ex5.7: Page-286 (2008)
```

```
clc; clear;
h = 6.62e-034;      // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;     // Energy quivalent of 1 eV, J
phi = 4.2*e;     // Work function for material, J
lambda = 2000e-010;     // Wavelength of incident radiation, m
E = h*c/lambda;     // Energy of the incidnt radiation, J
KE_fast = (E - phi)/e;     // Kinetic energy of the fastest photoelectron, eV
KE_slow = 0;     // Kinetic energy of the slowest photoelectron, eV
printf("\nThe kinetic energy of the fastest photoelectron = %d eV", KE_fast);
printf("\nThe kinetic energy of the slowest photoelectron = %d eV", KE_slow);
V = (E - phi)/e;     // Stopping potential, V
printf("\nThe stopping potential = %d volt", V);

// Result
// The kinetic energy of the fastest photoelectron = 2 eV
// The kinetic energy of the slowest photoelectron = 0 eV
// The stopping potential = 2 volt
```

Scilab code Exa 5.8 Maximum wavelength of radiation which would start the emission of photoelectrons

```
// Scilab Code Ex5.8: Page-287 (2008)
clc; clear;
h = 6.62e-027;      // Planck's constant, erg-s
c = 3e+010;     // Speed of light, cm/s
phi = 3.31e-012;     // Work function for material,
```

```
erg
lambda0 = h*c/phi;     // Wavelength of incident radiation, cm
printf("\nThe maximum wavelength of radiation which would start the emission of photoelectrons = %d angstrom", lambda0/1e-008);

// Result
// The maximum wavelength of radiation which would start the emission of photoelectrons = 6000 angstrom
```

Scilab code Exa 5.9 Potassium surface exposed to UV radiation

```
// Scilab Code Ex5.9: Page-287 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;     // Energy quivalent of 1 eV, J
phi = 2.1*e;     // Work function for material, J
lambda = 3500e-010;     // Wavelength of incident UV radiation, m
E = 1e-004;     // Energy incident per sec on 1 Sq. cm of potassium surface, J
eta = 0.5/100;     // Efficiency of potassium surface
KE = (h*c/lambda-phi)/e;     // Maximum kinetic energy of the ejected photoelectrons, eV
N = eta*E/(KE*e);     // Number of photoelectrons emitted per second per Sq. cm of potassium surface
printf("\nThe maximum kinetic energy of the incidnt radiation = %4.2f eV", KE);
printf("\nThe number of photoelectrons emitted per second per Sq. cm of potassium surface = %4.2e", N);
```

```

// Result
// The maximum kinetic energy of the incidnt radiation = 1.45 eV
// The number of photoelectrons emitted per second per Sq. cm of potassium surface = 2.16e+012
```

Scilab code Exa 5.10 Planck constant and threshold wavelength of metal

```
// Scilab Code Ex5.10: Page-288 (2008)
clc; clear;
c = 3e+008;     // Speed of light, m/s
KE1 = 3.62e-019;     // Maximum kinetic energy of photoelectrons with first wavelength, eV
lambda1 = 3000;     // First wavelength of incident radiation, angstrom
KE2 = 0.972e-019;     // Maximum kinetic energy of photoelectrons with second wavelength, eV
lambda2 = 5000;     // Second wavelength of incident radiation, angstrom
A = [c/lambda1, -1; c/lambda2, -1];     // Declare a square matrix as per Einstein's Photoelectric relation, KE = h*c/lambda - phi
B = [KE1; KE2];     // Put KEs in a column matrix
X = inv(A)*B;     // Apply inverse multiplication of a matrix to fing h and phi
lambda0 = X(1)*1e-010*c/X(2);     // Threshold wavelength of metal, m
printf("\nh = %4.2e Js\nphi = %1.0e J", X(1)*1e-010, X(2));
printf("\nThe threshold wavelength of metal = %d angstrom", ceil(lambda0/1e-010));

// Result
// h = 6.62e-034 Js
```

```
// phi = 3e-019 J
// The threshold wavelength of metal = 6620 angstrom
```

Scilab code Exa 5.11 Energy and wavelength of incident photon

```
// Scilab Code Ex5.11: Page-288 (2008)
clc; clear;
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;     // Energy equivalent of 1 eV, J
h = 6.62e-034;     // Planck's constant, Js
m0 = 9.1e-031;     // Rest mass of an electron, kg
alpha = 90;     // Scattering angle for X-ray photon,
    degree
d_lambda = h/(m0*c)*(1-cosd(alpha));     //
   Wavelength shift after collision, m
lambda = d_lambda;     // Wavelength of the incident
   photon according to the condition, m
E = h*c/(lambda*e*1e+006);     // Energy of the
   incident photon, MeV
printf("\nThe wavelength of the incident photon = %6
   .4e m", lambda);
printf("\nThe energy of the incident photon = %4.2f
   MeV", E);

// Result
// The wavelength of the incident photon = 2.4249e
   -012 m
// The energy of the incident photon = 0.51 MeV
```

Scilab code Exa 5.12 Energy lost by an X ray photon in collsion with an electron

```
// Scilab Code Ex5.12: Page-289 (2008)
```

```
clc; clear;
c = 3e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
h = 6.6e-034;     // Planck's constant, Js
lambda = 0.1;     // Wavelength of X ray photon, angstrom
m0 = 9.1e-031;     // Rest mass of an electron, kg
alpha = 90;     // Scattering angle for X-ray photon, degree
d_lambda = h/(m0*c*1e-010)*(1-cosd(alpha));     // Wavelength shift after collision, angstrom
lambda_prime = lambda + d_lambda;     // Wavelength of the scattered photon, angstrom
dE = h*c*1e+010/e*(1/lambda - 1/lambda_prime);     // Energy lost by the X ray photon by collision, eV
printf("\nThe energy lost by the X ray photon by collision = %4.1f KeV", dE/1e+003);

// Result
// The energy lost by the X ray photon by collision = 24.1 KeV
```

Scilab code Exa 5.13 The Compton effect stidied at different scattering angles

```
// Scilab Code Ex5.13: Page-289 (2008)
clc; clear;
c = 3e+008;     // Speed of light, m/s
e = 1.602e-019;     // Energy equivalent of 1 eV, J
h = 6.6e-034;     // Planck's constant, Js
m0 = 9.1e-031;     // Rest mass of an electron, kg
alpha = [90 60 45 180];     // Different scattering angle for X-ray photon, degrees
d_lambda = zeros(4);
for i = 1:1:4
```

```
    d_lambda(i) = h/(m0*c*1e-010)*(1-cosd(alpha(i))); // Wavelength shift after collision, angstrom
    printf("\nFor alpha = %d degree, d_lambda = %6.4f angstrom", alpha(i), d_lambda(i));
end
lambda = 0.2; // Given wavelength of incident X-ray photon, angstrom
lambda_prime = lambda + d_lambda(3); // Wavelength of the scattered photon at 45 degree, angstrom
printf("\nThe wavelength of the photon scattered at 45 degree = %5.3f angstrom", lambda_prime);
lambda_prime = lambda + d_lambda(4); // Maximum wavelength of the photon scattered at 180 degree, angstrom
KE_max = h*c*1e+010*(1/lambda - 1/lambda_prime); // Maximum kinetic energy of the recoil electron, J
printf("\nThe maximum kinetic energy of the recoil electron = %4.2e J", KE_max);

// Result
// For alpha = 90 degree, d_lambda = 0.0242 angstrom
// For alpha = 60 degree, d_lambda = 0.0121 angstrom
// For alpha = 45 degree, d_lambda = 0.0071 angstrom
// For alpha = 180 degree, d_lambda = 0.0484 angstrom
// The wavelength of the photon scattered at 45 degree = 0.207 angstrom
// The maximum kinetic energy of the recoil electron = 1.93e-015 J
```

Scilab code Exa 5.15 de Broglie wavelength associated with moving masses

```
// Scilab Code Ex5.15: Page-292 (2008)
clc; clear;
h = 6.6e-034;    // Planck's constant, Js
// For golf ball
m = 0.046;    // Mass of the golf ball, kg
v = 36;    // Velocity of the golf ball, m/s
lambda = h/(m*v);    // de-Broglie wavelength
    associated with the moving golf ball, m
printf("\nThe de-Broglie wavelength associated with
    the moving golf ball = %1.0e m", lambda);
if lambda/1e-010 > 0.1 then
    printf("\nThe moving golf ball may exhibit wave
        character.");
end
// For an electron
m = 9.11e-031;    // Mass of the electron, kg
v = 1e+007;    // Velocity of the electron, m/s
lambda = h/(m*v);    // de-Broglie wavelength
    associated with the moving electron, m
printf("\nThe de-Broglie wavelength associated with
    the moving electron = %3.1e m", lambda);
if lambda/1e-010 > 0.1 then
    printf("\nThe moving electron may exhibit wave
        character.");
end

// Result
// The de-Broglie wavelength associated with the
    moving golf ball = 4e-034 m
// The de-Broglie wavelength associated with the
    moving electron = 7.2e-011 m
// The moving electron may exhibit wave character.
```

Scilab code Exa 5.16 Voltage applied to the electron microscope to produce the required wavelength

```
// Scilab Code Ex5.16: Page-292 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
lambda = 0.40e-010; // de-Broglie wavelength associated with the moving electron, m
m = 9.11e-031;     // Rest mass of an electron, kg
V = (h/lambda)^2/(2*m*e);     // Voltage applied to the electron microscope to produce the required wavelength, volt
printf("\nThe voltage applied to the electron microscope to produce the required de-Broglie wavelength = %5.1f volt", V);

// Result
// The voltage applied to the electron microscope to produce the required de-Broglie wavelength = 938.4 volt
```

Scilab code Exa 5.18 de Broglie wavelength of a neutron of given energy

```
// Scilab Code Ex5.18: Page-293 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
E_k = 12.8e+006;     // Energy of the moving neutron, eV
m0 = 1.675e-027;     // Rest mass of a neutron, kg
lambda = h/sqrt(2*m0*E_k*e) // de-Broglie wavelength associated with the moving neutron, m
printf("\nThe de-Broglie wavelength of the moving neutron = %3.1e angstrom", lambda/1e-010);

// Result
// The de-Broglie wavelength of the moving neutron =
```

```
8.0e-005 angstrom
```

Scilab code Exa 5.19 Minimum uncertainty in momentum and kinetic energy of a proton confined within nucleus

```
// Scilab Code Ex5.19: Page-294 (2008)
clc; clear;
h = 6.62e-034;      // Planck's constant, Js
e = 1.602e-019;      // Energy equivalent of 1 eV, J
m = 1.67e-027;      // Rest mass of a proton, kg
r = 5e-015;     // Radius of the nucleus, m
delta_x = 2*r;      // Minimum uncertainty in position of the proton, m
delta_p = h/(2*%pi*delta_x);      // Minimum uncertainty in proton's momentum, kg-m/s
KE = delta_p^2/(2*m);      // Minimum kinetic emergy of the proton, J
printf("\nThe minimum uncertainty in momentum of the proton = %4.2e kg-m/s", delta_p);
printf("\nThe minimum kinetic emergy of the proton = %5.3f MeV", KE/(e*1e+006));

// Result
// The minimum uncertainty in momentum of the proton = 1.05e-020 kg-m/s
// The minimum kinetic emergy of the proton = 0.207 MeV
```

Scilab code Exa 5.20 Minimum uncertainty in the measurement of velocity of the electron

```
// Scilab Code Ex5.20: Page-294 (2008)
clc; clear;
```

```
h = 6.62e-034;     // Planck's constant, Js
m = 9.11e-031;     // Rest mass of a electron, kg
delta_x = 1e-009;     // Minimum uncertainty in
   position of the electron, m
delta_p_min = h/delta_x;     // Minimum uncertainty
   in electron's momentum, kg-m/s
delta_v = delta_p_min/m;     // Minimum uncertainty
   in the measurement of velocity of the electron, m
   /s
printf("\nThe minimum uncertainty in the measurement
    of velocity of the electron = %4.2e m/s",
   delta_v);

// Result
// The minimum uncertainty in the measurement of
   velocity of the electron = 7.27e+005 m/s
```

Scilab code Exa 5.22 Minimum uncertainty in the position of the particle

```
// Scilab Code Ex5.22: Page-295 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
m = 1e-009;     // Mass of the particle, kg
v = 1;     // Velocity of the particle, m/s
delta_v = v*0.01/100;     // Minimum uncertainty in
   the velocity of the particle, m/s
delta_x = h/(m*delta_v);     // Minimum uncertainty
   in the position of the particle, m
printf("\nThe minimum uncertainty in the position of
    the particle = %4.2e m", delta_x);

// Result
// The minimum uncertainty in the position of the
   particle = 6.62e-021 m
```

Scilab code Exa 5.23 Uncertainty with which position of the electron can be located

```
// Scilab Code Ex5.23: Page-295 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
m = 9.1e-031;     // Mass of the electron, kg
v = 1e+003;     // Velocity of the electron, m/s
delta_v = v*0.05/100;     // Minimum uncertainty in the velocity of the electron, m/s
delta_x = h/(m*delta_v);     // Minimum uncertainty in the position of the electron, m
printf("\nThe minimum uncertainty in the position of the electron = %4.2e m", delta_x);

// Result
// The minimum uncertainty in the position of the electron = 1.45e-003 m
```

Scilab code Exa 5.24 Minimum uncertainty in energy of the excited state of an atom

```
// Scilab Code Ex5.24: Page-295 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
delta_t = 1e-008;     // Life time of excited state of an atom, s
delta_E = h/(2*%pi*delta_t);     // Minimum uncertainty in the energy of the excited state of the atom, J
```

```
printf("\nThe minimum uncertainty in the energy of the excited state of the atom = %3.1e eV", delta_E/e);

// Result
// The minimum uncertainty in the energy of the excited state of the atom = 6.6e-008 eV
```

Scilab code Exa 5.25 Probable uncertainty in energy and frequency of gamma ray photon

```
// Scilab Code Ex5.25: Page-296 (2008)
clc; clear;
h = 6.62e-034;     // Planck's constant, Js
delta_t = 1e-012;     // Life time of a nucleus in the excited state, s
delta_E = h/(2*%pi*delta_t);     // Minimum uncertainty in the energy of the excited state of the nucleus, J
// As E = h*nu, solving for delta_nu
delta_nu = delta_E/h;     // Minimum uncertainty in the frequency of the excited state of the nucleus, Hz
printf("\nThe minimum uncertainty in the energy of the excited state of the nucleus = %5.3e J", delta_E);
printf("\nThe minimum uncertainty in the frequency of the excited state of the nucleus = %4.2e MHz", delta_nu/1e+006);

// Result
// The minimum uncertainty in the energy of the excited state of the nucleus = 1.054e-022 J
// The minimum uncertainty in the frequency of the excited state of the nucleus = 1.59e+005 MHz
```

Scilab code Exa 5.29 Lowest energy of an electron in one dimensional force free region

```
// Scilab Code Ex5.29: Page-300 (2008)
clc; clear;
h = 6.62e-034;      // Planck's constant, Js
e = 1.602e-019;     // Energy equivalent of 1 eV, J
m = 9.11e-031;      // Rest mass of the electron, kg
l = 4e-010;     // Length of the force free region, m
n = 1;      // Principal quantum number for lowest
    energy state
E1 = n^2*h^2/(8*m*l^2);     // Lowest energy of an
    electron in one dimensional force free region, J
printf("\nThe lowest energy of an electron in one
    dimensional force free region = %4.2f eV", E1/e);

// Result
// The lowest energy of an electron in one
    dimensional force free region = 2.35 eV
```

Scilab code Exa 5.30 The excited state energies of the particle entrapped in a one dimensional box

```
// Scilab Code Ex5.30: Page-300 (2008)
clc; clear;
e = 1.602e-019;     // Energy equivalent of 1 eV, J
E1 = 3.2e-018/e;     // Minimum energy possible for a
     particle entrapped in a one dimensional box, eV
n = [1 2 3 4];     // Principal quantum number for K,
     L, M and N states
printf("\nThe next three energies which the particle
     can have are:");
```

```
for i = 2:1:4
    printf("\nE%d = %d eV", i, ceil(i^2*E1));
end

// Result
// The next three energies which the particle can
    have are:
// E2 = 80 eV
// E3 = 180 eV
// E4 = 320 eV
```

Scilab code Exa 5.31 Probability of finding the particle within a given interval

```
// Scilab Code Ex5.31: Page-301 (2008)
clc; clear;
delta_x = 4;    // Interval at the centre of the box
     at which the probability is to be found out,
    angstrom
l = 10;    // Width of one dimensional infinite
    height box, angstrom
P = 2*delta_x/l;    // Probability of finding the
    particle within 4 angstrom interval
printf("\nThe probability of finding the particle
    within the %d angstrom interval at the centre of
    the box = %3.1f", delta_x, P);

// Result
// The probability of finding the particle within
    the 4 angstrom interval at the centre of the box
    = 0.8
```

Scilab code Exa 5.32 Probability of finding a particle within given range of 1D box for different energy states

```
// Scilab Code Ex5.32: Page-301 (2008)
clc; clear;
L = 1;    // Assume the length of the box to be unity, m
L1 = 0.4*L;    // Lower limit, m
L2 = 0.6*L;    // Upper limit, m
x = (L1+L2)/2;    // Mean position of particle, m
delta_x = L2 - L1;    // Uncertainty in position of the particle, m
for n = 1:1:3
    P = 2/L*sin(n*%pi*x/L)^2;    // Probability density, per m
    printf("\nFor n = %d, the probability, P = %3.1f", n, P*delta_x);
end

// Result
// For n = 1, the probability, P = 0.4
// For n = 2, the probability, P = 0.0
// For n = 3, the probability, P = 0.4
```

Chapter 6

Classical Statistics and Quantum Statistics

Scilab code Exa 6.1 Probability of distribution of distinguishable particles

```
// Scilab Code Ex6.1: Page-345 (2008)
clc; clear;
n = 14;    // Total number of particles
C = 2;    // Total number of compartments
N_micro = C^n;    // Total number of microstates
n1 = [10 7 14];    // Set of number of particles in
    first compartment
n2 = [4 7 0];    // Set of number of particles in
    second compartment
for i = 1:1:3
    W = factorial(n1(i) + n2(i))/(factorial(n1(i))*
       factorial(n2(i)));
    P = W/N_micro;
    printf("\nThe probability of microstate (%d, %d)
        = %8.6f", n1(i), n2(i), P);
end

// Result
```

```
// The probability of microstate (10, 4) = 0.061096
// The probability of microstate (7, 7) = 0.209473
// The probability of microstate (14, 0) = 0.000061
```

Scilab code Exa 6.6 Most probable distribution for total energy

```
// Scilab Code Ex6.6: Page-348 (2008)
clc; clear;
MAX = 10;
// Look for all the possible set of values for n1,
   n2 and n3
printf("\nThe most probable distribution is for ");
for i = 0:1:5
    for j = 0:1:5
        for k = 0:1:5
      // Check for the condition and avoid
         repetition of set of values
            if ((i + j + k) == 5) & ((j+2*k) == 3)
               then
                W = factorial(i + j + k)/(factorial(
                   i)*factorial(j)*factorial(k));
                if W > MAX then
                   printf("\nn1 = %d, n2 = %d and n3
                       = %d", i, j, k);
                end
            end
        end
    end
end

// Result
// The most probable distribution is for
// n1 = 3, n2 = 1 and n3 = 1
```

Scilab code Exa 6.8 Probability for a Maxwell Boltzmann system to be in given states

```
// Scilab Code Ex6.8: Page-349 (2008)
clc; clear;
k = 1.38e-016;    // Boltzmann constant, erg/K
T = 100;    // Given temperature, K
E1 = 0;    // Energy of the first state, erg
E2 = 1.38e-014;    // Energy of the second state,
    erg
E3 = 2.76e-014;    // Energy of the third state, erg
g1 = 2, g2 = 5, g3 = 4;    // Different ways of
    occuring for E1, E2 and E3 states
P1 = g1*exp(-E1/(k*T));    // Probability of
    occurence of state E1
P2 = g2*exp(-E2/(k*T));    // Probability of
    occurence of state E2
P3 = g3*exp(-E3/(k*T));    // Probability of
    occurence of state E3
PE_3 = P3/(P1+P2+P3);    // Probability for the
    system to be in any one microstates of E3
P0 = P1/(P1+P2+P3);    // Probability for the system
     to be in ground state
printf("\nThe probability for the system to be in
    any one microstates of E3 = %6.4f", PE_3);
printf("\nThe probability for the system to be in
    ground state = %5.3f", P0);

// Result
// The probability for the system to be in any one
    microstates of E3 = 0.1236
// The probability for the system to be in ground
    state = 0.457
```

Scilab code Exa 6.9 Number of microstates in the given macostate of a Fermi Dirac system

```
// Scilab Code Ex6.9: Page-350 (2008)
clc; clear;
g1 = 6, g2 = 8;     // Total number of cells in the
    first and the second compartments respectively
n1 = 2, n2 = 3;     // Given number of cells in the
    first and the second compartments respectively
    for given macrostate
W_23 = factorial(g1)/(factorial(n1)*factorial(g1 -
    n1))*factorial(g2)/(factorial(n2)*factorial(g2 -
    n2));    // Total number of microstates in the
    macrostate (2, 3)
printf("\nThe total number of microstates in the
    macrostate (%d, %d) = %d", n1, n2, W_23);

// Result
// The total number of microstates in the macrostate
    (2, 3) = 840
```

Scilab code Exa 6.10 Number of microstates formed by particles obeying Fermi Dirac statistics

```
// Scilab Code Ex6.10: Page-350 (2008)
clc; clear;
g1 = 8, g2 = 10;     // Total number of cells in the
    first and the second compartments respectively
n1 = 3, n2 = 4;     // Given number of cells in the
    first and the second compartments respectively
    for given macrostate
```

```
W_34 = factorial(g1)/(factorial(n1)*factorial(g1 -
   n1))*factorial(g2)/(factorial(n2)*factorial(g2 -
   n2));    // Total number of microstates in the
   macrostate (3, 4)
printf("\nThe total number of microstates in the
   macrostate (%d, %d) = %d", n1, n2, W_34);

// Result
// The total number of microstates in the macrostate
    (3, 4) = 11760
```

Scilab code Exa 6.11 Fermi energy and internal energy for metallic silver at 0 K

```
// Scilab Code Ex6.11: Page-351 (2008)
clc; clear;
h = 6.6e-034;     // Planck's constant, Js
m = 9.1e-031;     // Mass of an electron, kg
e = 1.6e-019;     // Energy equivalent of 1 eV, J
rho = 10.5;     // Density of silver, g/cc
A = 108;     // Atomic weight of Ag, g/mole
N_A = 6.023e+023;     // Avogadro's number
E_F0 = h^2/(8*m)*(3*N_A*rho*1e+006/(%pi*A))^(2/3);
        // Fermi energy of silver at 0 K, J
U = 3/5*(N_A*rho*1e+006/A)*E_F0;     // Internal
   energy of the electron gas per unit volume at 0 K
   , J/metre-cube
printf("\nThe Fermi energy of silver at 0 K = %3.1f
   eV", E_F0/e);
printf("\nThe internal energy of the electron gas
   per unit volume at 0 K = %4.2e J/cubic-metre", U)
   ;

// Result
// The Fermi energy of silver at 0 K = 5.5 eV
```

```
// The internal energy of the electron gas per unit
   volume at 0 K = 3.07e+010 J/cubic-metre
```

Scilab code Exa 6.12 Number of conduction electrons per cc in silver at 0 K

```
// Scilab Code Ex6.12: Page-351 (2008)
clc; clear;
h = 6.6e-034;     // Planck's constant, Js
m = 9.1e-031;     // Mass of an electron, kg
e = 1.6e-019;     // Energy equivalent of 1 eV, J
E_F0 = 5.48;      // Fermi energy of silver at 0 K,
   eV
N_bar = (8*m/h^2)^(3/2)*%pi/3*(E_F0*e)^(3/2);    //
   Number density of conduction electrons in silver
   at 0 K, per cc
printf("\nThe number density of conduction electrons
    in silver at 0 K = %3.1e per cc", N_bar*1e-006);

// Result
// The number density of conduction electrons in
   silver at 0 K = 5.9e+022 per cc
```

Scilab code Exa 6.13 Fermi energy of conduction electrons in cesium

```
// Scilab Code Ex6.13: Page-351 (2008)
clc; clear;
h = 6.6e-034;     // Planck's constant, Js
m = 9.1e-031;     // Mass of an electron, kg
e = 1.6e-019;     // Energy equivalent of 1 eV, J
E_F0_Be = 14.44       // Fermi energy of Be at 0 K,
   eV
```

```
N_bar_Be = 24.2e+022;    // Number density of
    conduction electrons in Be at 0 K, per cc
N_bar_Cs = 0.91e+022;    // Number density of
    conduction electrons in Cs at 0 K, per cc
E_F0_Cs = E_F0_Be*(N_bar_Cs/N_bar_Be)^(2/3);    //
    Fermi energy of conduction electrons in cesium,
    eV
printf("\nThe Fermi energy of conduction electrons
    in cesium = %5.3f eV", E_F0_Cs);

// Result
// The Fermi energy of conduction electrons in
    cesium = 1.621 eV
// The answer is given wrongly in the textbook
```

Chapter 7

Classical Statistics and Quantum Statistics

Scilab code Exa 7.1 Atomic packing fractions of SC FCC and BCC unit cells

```
// Scilab Code Ex7.1: Page-376 (2008)
clc; clear;
a = poly(0, 'a');    // Lattice parameter for a cubic unit cell, m
// For simple cubic cell
n = 1;    // Number of atoms per simple cubic unit cell
r = a/2;    // Atomic radius for a simple cubic cell, m
f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));    // Atomic packing fraction for a simple cubic cell
printf("\nFor simple cubic cell, f = %s percent", f);
// For face centered cubic cell
n = 2;    // Number of atoms per face centered cubic unit cell
r = sqrt(3)/4*a;    // Atomic radius for a face
```

```
centered cubic cell, m
f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));    // Atomic packing fraction for a face centered cubic cell
printf("\nFor face centered cubic cell, f = %s percent", f);
// For body centered cubic cell
n = 4;    // Number of atoms per body centered cubic unit cell
r = a/(2*sqrt(2));    // Atomic radius for a body centered cubic cell, m
f = pol2str(int(numer(n*4/3*%pi*r^3/a^3)*100));    // Atomic packing fraction for a body centered cubic cell
printf("\nFor body centered cubic cell, f = %s percent", f);

// Result
// For simple cubic cell, f = 52 percent
// For face centered cubic cell, f = 68 percent
// For body centered cubic cell, f = 74 percent
```

Scilab code Exa 7.3 Distance between two adjacent atoms in the NaCl

```
// Scilab Code Ex7.3: Page-377 (2008)
clc; clear;
M = 58.46;    // Gram atomic mass of NaCl, g/mole
N = 6.023e+023;    // Avogadro's number
rho = 2.17;    // Density of NaCl, g/cc
m = M/N;    // Mass of each NaCl molecule, g
n = rho/m;    // Number of NaCl molecules per unit volume, molecules/cc
N = 2*n;    // Number of atoms per unit volume, atoms/cc
a = (1/N)^(1/3);    // Distance between two adjacent
```

```
atoms in the NaCl, cm
printf("\nThe distance between two adjacent atoms in
    the NaCl = %4.2f angstrom", a/1e-008);

// Result
// The distance between two adjacent atoms in the
   NaCl = 2.82 angstrom
```

Scilab code Exa 7.4 Type of unit cell of Cs

```
// Scilab Code Ex7.4: Page-377 (2008)
clc; clear;
function p = find_cell_type(x)
    if x == 1 then
        p = 'simple cubic';
    end
    if x == 2 then
        p = 'body centered';
    end
    if x == 4 then
        p = 'face centered';
    end
endfunction
M = 130;    // Gram atomic weight of Cs, g/mole
N = 6.023e+023;    // Avogadro's number
rho = 2;    // Density of Cs, g/cc
a = 6e-008;    // Distance between two adjacent
   atoms in the Cs, cm
m = M/N;    // Mass of each Cs atom, g
x = rho*a^3*N/M;    // Number of Cs atoms in cubic
   unit cell
c_type = find_cell_type(int(x));    // Call function
    to determine the type of cell
printf("\nThe cubic unit cell of Cs is %s.", c_type)
   ;
```

```

// Result
// The cubic unit cell of Cs is body centered.
```

Scilab code Exa 7.5 Miller indices of given planes

```
// Scilab Code Ex7.5: Page-378 (2008)
clc; clear;
m = 2; n = 3; p = 6; // Coefficients of intercepts along three axes
m_inv = 1/m;          // Reciprocate the first coefficient
n_inv = 1/n;          // Reciprocate the second coefficient
p_inv = 1/p;          // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe required miller indices are : (%d %d %d) ", m1,m2,m3);

// Result
// The required miller indices are : (3 2 1)
```

Scilab code Exa 7.6 Meaning of hkl notation of planes

```
// Scilab Code Ex7.6: Page-378 (2008)
clc; clear;
// For first set (3, 2, 2)
```

```
m = 3; n = 2; p = 2; // Coefficients of intercepts along three axes
m_inv = 1/m;         // Reciprocate the first coefficient
n_inv = 1/n;         // Reciprocate the second coefficient
p_inv = 1/p;         // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe plane (%d %d %d) has intercepts %da, %db and %dc on the three axes.", m, n, p, m1, m2, m3);
// For second set (1 1 1)
m = 1; n = 1; p = 1; // Coefficients of intercepts along three axes
m_inv = 1/m;         // Reciprocate the first coefficient
n_inv = 1/n;         // Reciprocate the second coefficient
p_inv = 1/p;         // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe plane (%d %d %d) has intercepts a, b and c on the three axes.", m, n, p);

// Result
// The plane (3 2 2) has intercepts 2a, 3b and 3c on the three axes.
// The plane (1 1 1) has intercepts a, b and c on the three axes.
```

Scilab code Exa 7.9 Lengths of intercepts along y and z axis

```
// Scilab Code Ex7.9: Page-379 (2008)
clc; clear;
h = 2; k = 3; l = 1; // Miller indices of the set of
    planes
p = 1/h;         // Reciprocate h
q = 1/k;         // Reciprocate k
r = 1/l;         // Reciprocate l
lx = 1.2;     // Intercept cut by plane along x-axis ,
    angstrom
a = 1.2, b = 1.8, c = 2;     // Primitives of the
    crystal , angstrom
mul_fact = double(lcm(int32([h, k, l]))); // Find l.
    c.m. of h, k and l
pa = mul_fact*p*a;
qb = mul_fact*q*b;
rc = mul_fact*r*c;
ly = lx*qb/pa;     // Length of intercept along y-
    axis
lz = lx*rc/pa;     // Length of intercept along z-
    axis
printf("\nThe length of intercept along y-axis = %3
    .1f angstrom", ly);
printf("\nThe length of intercept along z-axis = %3
    .1f angstrom", lz);

// Result
// The length of intercept along y-axis = 1.2
    angstrom
// The length of intercept along z-axis = 4.0
    angstrom
```

Scilab code Exa 7.10 Interplanar spacing for a set of planes in a cubic lattice

```
// Scilab Code Ex7.10: Page-380 (2008)
clc; clear;
h = 3; k = 2; l = 1; // Miller Indices for planes in
    a cubic crystal
a = 4.21D-10;    // Interatomic spacing, m
d = a/(h^2+k^2+l^2)^(1/2);  // The interplanar
   spacing for cubic crystals, m
printf("\nThe interplanar spacing between
   consecutive (321) planes = %3.1e m", d);

// Result
// The interplanar spacing between consecutive (321)
    planes = 1.1e-010 m
```

Scilab code Exa 7.11 Determining Planck constant from given set of X ray data

```
// Scilab Code Ex7.11: Page-380 (2008)
clc; clear;
e = 1.6e-019;    // The energy equivalent of 1 eV, J
c = 3e+008;    // Speed of light in vacuum, m/s
V = [30 44 50 200];    // Operating voltages of X
   ray, kV
lambda_min = [0.414 0.284 0.248 0.062];    //
   Minimum wavelengths of emitted continuous X rays,
    angstrom
for i = 1:1:4
    h = e*V(i)*1e+003*lambda_min(i)*1e-010/c;    //
       Planck's constant, Js
```

```
    printf("\nFor V = %d kV and lambda_min = %5.3f
       angstrom, h = %4.2e Js", V(i), lambda_min(i),
        h);
end

// Result
// For V = 30 kV and lambda_min = 0.414 angstrom, h
   = 6.62e-034 Js
// For V = 44 kV and lambda_min = 0.284 angstrom, h
   = 6.66e-034 Js
// For V = 50 kV and lambda_min = 0.248 angstrom, h
   = 6.61e-034 Js
// For V = 200 kV and lambda_min = 0.062 angstrom, h
    = 6.61e-034 Js
```

Scilab code Exa 7.12 Maximum speed of striking electron and the shortest wavelength of X ray produced

```
// Scilab Code Ex7.12: Page-381 (2008)
clc; clear;
e = 1.6e-019;    // The energy equivalent of 1 eV, J
m = 9.11e-031;    // Rest mass of an electron, kg
h = 6.62e-034;    // Planck's constant, Js
c = 3e+008;    // Speed of light in vacuum, m/s
V = [20 100];    // Operating voltages of X ray, kV
for i = 1:1:2
    v = sqrt(2*e*V(i)*1e+003/m);    // Maximum
       striking speed of the electron, m/s
    lambda_min = c*h/(e*V(i)*1e+003*1e-010);    //
       Minimum wavelength of emitted continuous X
       rays, angstrom
    printf("\nFor V = %d kV:", V(i));
    printf("\nThe maximum striking speed of the
       electron = %5.2e m/s", v);
    printf("\nThe minimum wavelength of emitted
```

```
continuous X rays = %5.3f angstrom\n",
lambda_min);
end

// Result
// For V = 20 kV:
// The maximum striking speed of the electron = 8.38
e+007 m/s
// The minimum wavelength of emitted continuous X
rays = 0.621 angstrom
//
// For V = 100 kV:
// The maximum striking speed of the electron = 1.87
e+008 m/s
// The minimum wavelength of emitted continuous X
rays = 0.124 angstrom
// There are small variation in the answers as
approximations are used in the text
```

Scilab code Exa 7.13 Interatomic spacing using Bragg relation

```
// Scilab Code Ex7.13: Page-381 (2008)
clc; clear;
n = 1;    // Order of diffraction
lambda = 1.75e-010;    // Wavelength of X rays, m
h = 1, k = 1, l = 1;    // Miller indices for the
set of planes
theta = 30;    // Bragg's angle, degree
// As from Bragg's law, 2*d*sind(theta) = n*lambda
and d = a/sqrt(h^2+k^2+l^2). solving for a we
have
a = sqrt(h^2+k^2+l^2)*n*lambda/(2*sind(theta)*1e
-010);    // Interatomic spacing of the crystal,
angstrom
printf("\nThe interatomic spacing of the crystal =
```

```
%5.3f angstrom", a);

// Result
// The interatomic spacing of the crystal = 3.031 angstrom
```

Scilab code Exa 7.14 Value of Planck constant from Bragg relation

```
// Scilab Code Ex7.14: Page-382 (2008)
clc; clear;
e = 1.6e-019;    // The energy equivalent of 1 eV, J
c = 3e+008;    // Speed of light in vacuum, m/s
n = 1;    // Order of diffraction
d = 2.82e-010;    // Interplanar spacing, m
V = 9.1e+003;    // Operating voltage of X rays
theta = 14;    // Bragg's angle, degree
lambda = 2*d*sind(theta)/n;    // Wavelength of X rays, m
nu = c/lambda;    // Frequency of X rays, Hz
h = e*V/nu;    // Planck's constant, Js
printf("\nThe value of Planck constant, h = %4.2e Js ", h);

// Result
// The value of Planck constant, h = 6.62e-034 Js
```

Scilab code Exa 7.15 Diffraction of X rays from a crystal

```
// Scilab Code Ex7.15: Page-382 (2008)
clc; clear;
e = 1.6e-019;    // The energy equivalent of 1 eV, J
c = 3e+008;    // Speed of light in vacuum, m/s
lambda = 0.5e-010;    // Wavelength of X rays, m
```

```
theta = 5;     // Bragg's angle, degree
n = 1;     // Order of diffraction
d = n*lambda/(2*sind(theta)*1e-010);     // Interplanar spacing, angstrom
n = 2;     // Ordr of diffraction
theta1 = asind(n*lambda/(2*d*1e-010));     // Angle at which the second maximum occur, degree
printf("\nThe spacing between adjacent planes of the crystal = %4.2f angstrom", d);
printf("\nThe angle at which the second maximum occur = %5.2f degree", theta1);

// Result
// The spacing between adjacent planes of the crystal = 2.87 angstrom
// The angle at which the second maximum occur = 10.04 degree
```

Scilab code Exa 7.16 Wavelength of X rays from grating space of the rock salt

```
// Scilab Code Ex7.16: Page-383 (2008)
clc; clear;
M = 58.5     // Gram atomic mass of NaCl, kg/mole
N = 6.023e+026;     // Avogadro's number per kmol
rho = 2.17e+003;     // Density of NaCl, kg/metre-cube
m = M/N;     // Mass of each NaCl molecule, g
V = m/rho;     // Volume of each NaCl molecule, metre-cube
d = (V/2)^(1/3)/1e-010;     // Atomic apacing in the NaCl crystal, angstrom
theta = 26;     // Bragg's angle, degree
n = 2;     // Order of diffraction
lambda = 2*d*sind(theta)/n;     // Wavelength of X
```

```
rays, m
printf("\nThe grating spacing of rock salt = %4.2f angstrom", d);
printf("\nThe wavelength of X rays = %4.2f angstrom", lambda);

// Result
// The grating spacing of rock salt = 2.82 angstrom
// The wavelength of X rays = 1.24 angstrom
```

Scilab code Exa 7.17 Diffraction of X rays by the calcite crystal

```
// Scilab Code Ex7.17: Page-383 (2008)
clc; clear;
d = 3.02945e-010;    // Atomic apacing in the calcite crystal, m
lambda_alpha = 0.563e-010;    // Wavelength of the K-alpha line of Ag, m
n = 1;    // Order of diffraction
theta = asind(n*lambda_alpha/(2*d));    // Angle of reflection for the first order, degree
theta_max = 90;    // Angle of reflection for the highest order, degree
n = 2*d*sind(theta_max)/lambda_alpha;    // The highest order for which the line may be observed
printf("\nThe angle of reflection for the first order = %4.2f degree", theta);
printf("\nThe highest order for which the line may be observed = %d", n);

// Result
// The angle of reflection for the first order = 5.33 degree
// The highest order for which the line may be observed = 10
```

Scilab code Exa 7.18 Interatomic spacing for given crystal planes

```
// Scilab Code Ex7.18: Page-384 (2008)
clc; clear;
lambda = 1.8e-010;     // Wavelength of the X rays, m
n = 1;     // Order of diffraction
theta = 60;     // Angle of diffraction for the first
    order, degree
d = n*lambda/(2*sind(theta));     // Interplanar
    spacing, m
// Since for a simple cubic lattice, d_111 = d = a/
    sqrt(3), solving for a
a = sqrt(3)*d;     // The interatomic spacing for the
    given crystal planes, m
printf("\nThe interatomic spacing for the given
    crystal planes, a = %3.1f angstrom", a/1e-010);

// Result
// The interatomic spacing for the given crystal
    planes, a = 1.8 angstrom
```

Scilab code Exa 7.19 Smallest angle between the crystal plane and the X ray beam

```
// Scilab Code Ex7.19: Page-384 (2008)
clc; clear;
function [d, m] = deg2degmin(theta)
    d = int(theta);
    m = (theta-d)*60;
endfunction
h = 6.626e-034;     // Planck's constant, Js
```

```
e = 1.6e-019;     // The energy equivalent of 1 eV, J
c = 3e+008;     // Speed of light in vacuum, m/s
V = 50e+003;     // Operating voltage of X ray, V
lambda_min = h*c/(e*V);     // Minimum wavelength of
    emitted continuous X rays, angstrom
n = 1;     // Order of diffraction
d = 3.02945e-010;     // Interplanar spacing, m
theta = asind(n*lambda_min/(2*d));     // The
    smallest angle between the crystal plane and the
    X ray beam, degree
[deg , m] = deg2degmin(theta);
printf("\nThe smallest angle between the crystal
    plane and the X ray beam = %d degree %d min", deg
    , m);

// Result
// The smallest angle between the crystal plane and
    the X ray beam = 2 degree 21 min
```

Chapter 8

Laser and Fibre Optics

Scilab code Exa 8.1 Image produced by laser beam

```
// Scilab Code Ex8.1: Page-397 (2008)
clc; clear;
lambda = 6000e-008;      // Wavelength of the lase beam, cm
P = 10e-003;     // Power of the laser beam, W
theta = 1.5e-004;    // Angular spread of laser beam, rad
f = 10;     // Focal length of the lens, cm
r = f*theta;     // Radius of the image, cm
rho = P/(%pi*r^2*1e+003);  // Power density of the image, kW/Sq.cm
L_w = lambda/(theta/10);      // Coherence width, mm
printf("\nThe radius of the image = %3.1e cm", r);
printf("\nThe power density of the image = %3.1f kW/Sq.cm", rho);
printf("\nThe coherence width = %d mm", L_w);

// Result
// The radius of the image = 1.5e-03 cm
// The power density of the image = 1.4 kW/Sq.cm
// The coherence width = 4 mm
```

Scilab code Exa 8.2 Pumping energy required for He Ne laser transition

```
// Scilab Code Ex8.2: Page-398 (2008)
clc; clear;
lambda = 632.8e-009;     // Wavelength of the lase beam, cm
E_2P = 15.2e-019;     // Energy of 2P level, J
h = 6.626e-034;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
E_Pump = E_2P + h*c/lambda; // The required pumping energy, J
printf("\nThe pumping energy required for He Ne laser transition = %5.2f eV", E_Pump/e);

// Result
// The pumping energy required for He Ne laser transition = 11.46 eV
```

Scilab code Exa 8.3 Wavelength of radiation emitted at room temperature

```
// Scilab Code Ex8.3: Page-398 (2008)
clc; clear;
h = 6.626e-034;     // Planck's constant, Js
c = 3e+008;     // Speed of light, m/s
T = 27+273;     // Room temperature, K
k = 1.38e-023;   // Boltzmann constant, J/mol/K
lambda = h*c/(k*T); // Wavelength of radiation mitted at room temperature, m
printf("\nThe wavelength of radiation mitted at room temperature = %3.1e m", lambda);
```

```

// Result
// The wavelength of radiation mitted at room
   temperature = 4.8e-05 m
```

Scilab code Exa 8.4 Refractive index of the cladding in an optical fibre

```
// Scilab Code Ex8.4: Page-398 (2008)
clc; clear;
NA = 0.5;    // Numerical aperture of the optical
   fibre
n1 = 1.54;  // Refractive index of the core material
n2 = sqrt(n1^2-NA^2);    // Refractive index of the
   cladding in an optical fibre
printf("\nThe refractive index of the cladding in
   the optical fibre = %4.2f", n2);

// Result
// The refractive index of the cladding in the
   optical fibre = 1.46
```

Scilab code Exa 8.5 Numerical aperture and acceptance angle of the optical fibre

```
// Scilab Code Ex8.5: Page-398 (2008)
clc; clear;
n1 = 1.51;  // Refractive index of the core material
n2 = 1.47;    // Refractive index of the cladding
NA = sqrt(n1^2-n2^2);    // Numerical aperture of the
    optical fibre
n0 = 1;      // Refractive index of air
theta_a = asin(NA/n0);  // Acceptance angle of the
   optical fibre, rad
```

```
printf("\nThe numerical aperture of the optical
   fibre = %6.4f", NA);
printf("\nThe acceptance angle of the optical fibre
   = %4.2f degrees", theta_a*180/3.14);

// Result
// The numerical aperture of the optical fibre =
   0.3453
// The acceptance angle of the optical fibre = 20.21
    degrees
```

Chapter 9

Nuclear Physics

Scilab code Exa 9.1.1 Binding energy per nucleon for Ni

```
// Scilab Code Ex9.1.1:Page-411 (2008)
clc; clear;
u = 931.508;    // Energy equivalent of 1 amu, MeV
Z = 28;    // Atomic number of ni-64
A = 64;    // Mass number of Ni-64
m_p = 1.007825;    // Mass of a proton, u
m_n = 1.008665;    // Mass of a neutron, u
M_Ni = 63.9280;    // Atomic mass of Ni-64 nucleus,
    u
delta_m = Z*m_p + (A-Z)*m_n - M_Ni;    // Mass
    difference, u
BE = delta_m*u;    // Binding energy of Ni-64
    nucleus, MeV
BE_bar = BE/A;    // Binding energy per nucleon of
    Ni-64 nucleus, MeV
printf("\nThe binding energy per nucleon for Ni-64
    nucleus = %4.2f MeV/nucleon", BE_bar);

// Result
// The binding energy per nucleon for Ni-64 nucleus
    = 8.78 MeV/nucleon
```

Scilab code Exa 9.1.2 Binding energy per nucleon for deutron

```
// Scilab Code Ex9.1.2:Page-411 (2008)
clc; clear;
e = 1.6e-013;    // Energy equivalent of 1 MeV, J
m_p = 1.672e-027;     // Mass of a proton, kg
m_n = 1.675e-027;    // Mass of a neutron, kg
M_D = 3.343e-027;    // Mass of a deutron, kg
c = 3.00e+008;    // Speed of light in vacuum, m/s
delta_m = m_p + m_n - M_D;    // Mass defect, kg
E_B = delta_m*c^2/e;    // Binding energy for the
    deutron, MeV
BE_bar = E_B/2;    // Binding energy per nucleon for
     the deutron, MeV
printf("\nThe binding energy per nucleon for the
    deutron = %5.3f MeV/nucleon", BE_bar);

// Result
// The binding energy per nucleon for the deutron =
    1.125 MeV/nucleon
```

Scilab code Exa 9.1.3 Packing fraction and binding energy per nucleon for oxygen

```
// Scilab Code Ex9.1.3:Page-411 (2008)
clc; clear;
u = 931.508;    // Energy equivalent of 1 amu, MeV
Z = 8;    // Atomic number of O-16
A = 16;    // Mass number of O-16
m_p = 1.008142;     // Mass of a proton, u
m_n = 1.008982;    // Mass of a neutron, u
```

```
M_O = 15.994915;     // Atomic mass of O-16 nucleus, u
delta_m = Z*m_p + (A-Z)*m_n - M_O;     // Mass difference, u
BE = delta_m*u;     // Binding energy of O-16 nucleus, MeV
BE_bar = BE/A;     // Binding energy per nucleon of O-16 nucleus, MeV
delta_m = abs(M_O - A);     // Mass difference, u
PF = delta_m/A;     // Packing fraction for O-16 nucleus, u
printf("\nThe binding energy per nucleon for O-16 nucleus = %4.2f MeV/nucleon", BE_bar);
printf("\nThe packing fraction for O-16 nucleus = %5.3e u", PF);

// Result
// The binding energy per nucleon for O-16 nucleus = 8.27 MeV/nucleon
// The packing fraction for O-16 nucleus = 3.178e-004 u
```

Scilab code Exa 9.1.4 Atomic mass of neon

```
// Scilab Code Ex9.1.4: Page-411 (2008)
clc; clear;
u = 931.508;     // Energy equivalent of 1 amu, MeV
Z = 10;     // Atomic number of Ne-20
A = 20;     // Mass number of Ne-0
m_p = 1.007825;     // Mass of a proton, u
m_n = 1.008665;     // Mass of a neutron, u
BE = 160.64;     // Binding energy of Ne-20 nucleus, MeV
M = Z*m_p + (A-Z)*m_n + Z*0.51/u - BE/u;     // Atomic mass of Ne-20 nucleus, u
```

```
printf("\nThe atomic mass of Ne = %7.4f a.m.u", M);

// Result
// The atomic mass of Ne = 19.9979 a.m.u
```

Scilab code Exa 9.2.1 Average number of photons pe cubic metre in a monochromatic beam

```
// Scilab Code Ex9.2.1: Page-414 (2008)
clc; clear;
h = 6.63e-034;     // Planck's constant, Js
c = 3.00e+008;     // Speed of light in vacuum, m/s
I = 1e+004;     // Intensity of monochromatic beam, W/Sq.m
nu = 1e+004;     // Frequency of monochromatic beam, Hz
n = I/(h*nu*c);     // Average number of photons per cubic metre, photons/metre-cube
printf("\nThe average number of photons in the monochromatic beam of radiation = %4.2e photons/metre-cube", n);

// Result
// The average number of photons in the monochromatic beam of radiation = 5.03e+024 photons/metre-cube
```

Scilab code Exa 9.2.2 Average number of photons pe cubic metre in a monochromatic beam

```
// Scilab Code Ex9.2.2: : Page-414 (2008)
clc; clear;
h = 6.63e-034;     // Planck's constant, Js
```

```
c = 3.00e+008;     // Speed of light in vacuum, m/s
I = 1e+004;     // Intensity of monochromatic beam, W
    /Sq.m
nu = 1e+004;     // Frequency of monochromatic beam,
    Hz
n = I/(h*nu*c);     // Average number of photons per
    cubic metre, photons/metre-cube
printf("\nThe average number of photons in the
    monochromatic beam of radiation = %4.2e photons/
    metre-cube", n);

// Result
// The average number of photons in the
    monochromatic beam of radiation = 5.03e+024
    photons/metre-cube
```

Scilab code Exa 9.2.3 Photoelectric effect with silver

```
// Scilab Code Ex9.2.3: Page-414 (2008)
clc; clear;
h = 6.63e-034;     // Planck's constant, Js
c = 3.00e+008;     // Speed of light in vacuum, m/s
e = 1.6e-019;     // Energy equivalent of 1 eV, J
m_e = 9.1e-031;     // Rest mass of an electron, kg
lambda0 = 2762e-010;     // Thereshold wavelength of
    silver, m
lambda = 2000e-010;     // Wavelength of ultraviolet
    rays, m
E_max = h*c*(1/lambda - 1/lambda0);     // Maximum
    kinetic energy of the ejected electrons from
    Einstein's photoelectric equation, J
// As E_max = 1/2*m_e*v^2, solving for v
v_max = sqrt(2*E_max/m_e);     // Maximum velocity of
     the photoelectrons, m/s
V0 = E_max/e;     // Stopping potential for the
```

```
electrons, V
printf("\nThe maximum kinetic energy of the ejected electrons = %5.3e J", E_max);
printf("\nThe maximum velocity of the photoelectrons = %4.2e m/s", v_max);
printf("\nThe stopping potential for the electrons = %5.3f V", V0);

// Result
// The maximum kinetic energy of the ejected electrons = 2.744e-019 J
// The maximum velocity of the photoelectrons = 7.77e+005 m/s
// The stopping potential for the electrons = 1.715 V
```

Scilab code Exa 9.2.4 Work function of the metallic surface

```
// Scilab Code Ex9.2.4: Page-415 (2008)
clc; clear;
lambda1 = 3333e-010;    // First wavelength of the incident light, m
lambda2 = 2400e-010;    // Second wavelength of the incident light, m
c = 3e+008;    // Speed of light in free space, m/s
e = 1.6e-019;    // Energy equivalent of 1 eV, J
E1 = 0.6;    // Kinetic energy of the emitted photoelectrons for the first wavelength, eV
E2 = 2.04;    // Kinetic energy of the emitted photoelectrons for the second wavelength, eV
h = (E2 - E1)*lambda1*lambda2*e/(c*(lambda1 - lambda2));    // Planck's constant, Js
W0 = (E2*lambda2 - E1*lambda1)/(lambda1 - lambda2);    // Work function of the metal, eV
printf("\nThe value of Planck constant = %3.1e Js",
```

```
h);
printf("\nThe work function of the metal = %3.1f eV", W0);

// Result
// The value of Planck constant = 6.6e-034 Js
// The work function of the metal = 3.1 eV
```

Scilab code Exa 9.2.5 Wavelength of the scattered photon

```
// Scilab Code Ex9.2.5: Page-415 (2008)
clc; clear;
c = 3e+008;     // Speed of light in free space, m/s
h = 6.63e-034;     // Planck's constant, Js
m_e = 9.11e-031;     // Rest mass of an electron, kg
lambda = 0.3;     // Wavelength of incident X-ray photon, angstrom
phi = 45;     // The angle of scattering, degrees
lambda_prime = lambda + h/(m_e*c*1e-010)*(1-cosd(phi));     // The wavelength of the scattered photon, angstrom
printf("\nThe wavelength of the scattered photon = %6.4f angstrom", lambda_prime);

// Result
// The wavelength of the scattered photon = 0.3071 angstrom
```

Scilab code Exa 9.2.6 de Broglie wavelength of the valence electron in metallic sodium

```
// Scilab Code Ex9.2.6: Page-416 (2008)
clc; clear;
```

```
h = 6.63e-034;     // Planck's constant, Js
m_e = 9.11e-031;     // Rest mass of an electron, kg
e = 1.6e-019;     // Energy equivalent of 1 eV, J
K = 3*e;     // Kinetic energy of the electron in
   metllic sodium, J
lambda = h/sqrt(2*m_e*K)/1e-010;     // de Broglie
   wavelength of the valence electron, angstrom
printf("\nThe de-Broglie wavelength of the valence
   electron = %3.1f angstrom", lambda);

// Result
// The de-Broglie wavelength of the valence electron
   = 7.1 angstrom
```

Scilab code Exa 9.2.7 de Broglie wavelength of a moving electron

```
// Scilab Code Ex9.2.7: Page-416 (2008)
clc; clear;
h = 6.63e-034;     // Planck's constant, Js
m = 9.11e-031;     // Rest mass of an electron, kg
c = 3e+008;     // Speed of light in vacuum, m/s
bita = 3/5;     // Boost parameter
v = 3/5*c;     // Spped of the electron, m/s
lambda = h/(m*v)*sqrt(1-bita^2);     // de Broglie
   wavelength of the electron, m
printf("\nThe de-Broglie wavelength of the moving
   electron = %6.4f angstrom", lambda/1e-010);

// Result
// The de-Broglie wavelength of the moving electron
   = 0.0323 angstrom
```

Scilab code Exa 9.2.8 Uncertainty in energy and frequency of emitted light

```
// Scilab Code Ex9.2.8: Page-416 (2008)
clc; clear;
h = 6.63e-034;    // Planck's constant, Js
h_bar = h/(2*%pi);    // Reduced Planck's constant, Js
delta_t = 1e-008;    // Time during which the radiation is emitted, s
delta_E = h_bar/delta_t;    // Minimum uncertainty in energy of emitted light, J
// As delta_E = h*delta_nu from Planck's quantum theory, solving for delta_nu
delta_nu = delta_E/h;    // Minimum uncertainty in frequency of emitted light, Hz
printf("\nThe minimum uncertainty in energy of emitted light = %5.3e J", delta_E);
printf("\nThe minimum uncertainty in frequency of emitted light = %4.2e Hz", delta_nu);

// Result
// The minimum uncertainty in energy of emitted ligh = 1.055e-026 J
// The minimum uncertainty in frequency of emitted ligh = 1.59e+007 Hz
```

Scilab code Exa 9.2.9 Shortest wavelength present in the radiation from an X ray machine

```
// Scilab Code Ex9.2.9: Page-417 (2008)
clc; clear;
h = 6.63e-034;    // Planck's constant, Js
c = 3e+008;    // Speed of light in free space, m/s
e = 1.6e-019;    // Energy equivalent of 1 eV, J
```

```
V = 50000;      // Accelerating potential, V
lambda_min = h*c/(e*V);     // The shortest
    wavelength present in the radiation from an X-ray
    machine, m
printf("\nThe shortest wavelength present in the
    radiation from an X-ray machine = %6.4f nm",
    lambda_min/1e-009);

// Result
// The shortest wavelength present in the radiation
    from an X-ray machine = 0.0249 nm
```

Scilab code Exa 9.2.11 Q value of nuclear reaction

```
// Scilab Code Ex9.2.11: Page-418(2008)
clc; clear;
u = 931.5;      // Energy equivalent of 1 amu, MeV
m_x = 4.002603;     // Mass of projectile (alpha-
    particle), u
m_y = 1.007825;     // Mass of emitted particle (
    proton), u
M_X = 14.0031;      // Mass of target nucleus (N-14),
    u
M_Y = 16.9994;     // Mass of daughter nucleus (O-16)
    , u
Q = ((m_x + M_X) - (m_y + M_Y))*u;     // Q-value of
    the reaction, MeV
printf("\nThe Q-value of the nuclear reaction = %5.3
    f MeV", Q);

// Result
// The Q-value of the nuclear reaction = -1.418 MeV
```

Scilab code Exa 9.2.12 Threshold energy for the reactions

```
// Scilab Code Ex9.2.12: Page-418(2008)
clc; clear;
u = 931.5;    // Energy equivalent of 1 amu, MeV
// First reaction
m_x = 1.007825;    // Mass of projectile (proton), u
m_y = 2.014102;    // Mass of emitted particle (deutron), u
M_X = 208.980394;    // Mass of target nucleus (Bi-209), u
M_Y = 207.979731;    // Mass of daughter nucleus (Bi-208), u
Q = ((m_x + M_X) - (m_y + M_Y))*u;    // Q-value of the reaction, MeV
Ex_threshold = -Q*(m_x + M_X)/M_X;    // The smallest value of the projectile energy, MeV
printf("\nThe threshhold energy of the reaction Bi(209,83) + p --> Bi(208,83) + d = %4.2f MeV", Ex_threshold);
// Second reaction
m_x = 4.002603;    // Mass of projectile (alpha-particle), u
m_y = 1.007825;    // Mass of emitted particle (proton), u
M_X = 27.98210;    // Mass of target nucleus (Al-27), u
M_Y = 30.973765;    // Mass of daughter nucleus (P-31), u
Q = ((m_x + M_X) - (m_y + M_Y))*u;    // Q-value of the reaction, MeV
Ex_threshold = -Q*(m_x + M_X)/M_X;    // The smallest value of the projectile energy, MeV
printf("\nThe threshhold energy of the reaction Al(27,13) + He --> P(31,15) + p = %4.2f MeV", Ex_threshold);

// Result
```

```
// The threshhold energy of the reaction Bi(209,83)
   + p --> Bi(208,83) + d = 5.25 MeV
// The threshhold energy of the reaction Al(27,13) +
    He --> P(31,15) + p = -3.31 MeV
```

Scilab code Exa 9.2.13 Finding unknown particles in the nuclear reactions

```
// Scilab Code Ex9.2.13: Page-418(2008)
clc; clear;
function p = Find(Z, A)
    if Z == 2 & A == 4 then
        p = 'alpha';
    end
    if Z == -1 & A == 0 then
        p = 'beta-';
    end
    if  Z == 1 & A == 0 then
        p = 'beta+';
    end
endfunction
R1 = cell(4,3);
R2 = cell(4,3);
// Enter data for first cell (Reaction)
R1(1,1).entries = 'Li'; // Element
R1(1,2).entries = 3;   // Atomic number
R1(1,3).entries = 6;    // Mass number
R1(2,1).entries = 'd';
R1(2,2).entries = 1;
R1(2,3).entries = 2;
R1(3,1).entries = 'X';
R1(3,2).entries = 0;
R1(3,3).entries = 0;
R1(4,1).entries = 'He';
R1(4,2).entries = 2;
```

```
R1(4,3).entries = 4;
// Enter data for second cell (Reaction)
R2(1,1).entries = "Te";
R2(1,2).entries = 52;
R2(1,3).entries = 122;
R2(2,1).entries = 'X';
R2(2,2).entries = 0;
R2(2,3).entries = 0;
R2(3,1).entries = 'I';
R2(3,2).entries = 53;
R2(3,3).entries = 124;
R2(4,1).entries = 'd';
R2(4,2).entries = 1;
R2(4,3).entries = 2;
R1(3,2).entries = R1(1,2).entries+R1(2,2).entries-R1(4,2).entries
R1(3,3).entries = R1(1,3).entries+R1(2,3).entries-R1(4,3).entries
particle = Find(R1(3,2).entries, R1(3,3).entries);      // Find the unknown particle
printf("\nFor the reaction\n")
            printf("\t%s(%d) + %s(%d) --> %s + %s(%d)\n X must be an %s particle", R1(1,1).entries, R1(1,3).entries, R1(2,1).entries, R1(2,3).entries, R1(3,1).entries, R1(4,1).entries, R1(4,3).entries, particle);
R2(2,2).entries = R2(3,2).entries+R2(4,2).entries-R2(1,2).entries
R2(2,3).entries = R2(3,3).entries+R2(4,3).entries-R2(1,3).entries
particle = Find(R2(2,2).entries, R2(2,3).entries);      // Find the unknown particle
printf("\n\nFor the reaction\n")
            printf("\t%s(%d) + %s --> %s(%d)+%s(%d)\n X must be an %s particle", R2(1,1).entries, R2(1,3).entries, R2(2,1).entries, R2(3,1).entries, R2(3,3).
```

```
entries, R2(4,1).entries, R2(4,3).entries, particle);

// Result
// For the reaction
//  Li(6) + d(2) --> X + He(4)
//  X must be an alpha particle

// For the reaction
//  Te(122) + X --> I(124)+d(2)
//  X must be an alpha particle
```

Scilab code Exa 9.2.14 Comptom scattering

```
// Scilab Code Ex9.2.14: Page-419(2008)
clc; clear;
h = 6.63e-034;    // Planck's constant, Js
c = 3e+008;    // Speed of light, m/s
lambda = 10e-012;    // Wavelength of incident X-rays, m
lambda_c = 2.426e-012;    // Compton wavelength for the electron, m
phi = 45;    // Angle of scattering of X-rays, degree
lambda_prime = lambda + lambda_c*(1 - cosd(phi));    // Wavelength of scattered X-rays, m
// For maximum wavelength
phi = 180;    // Angle for maximum scattering, degree
lambda_prime_max = lambda + lambda_c*(1 - cosd(phi));    // Maximum wavelength present in the scattered X-rays, m
KE_max = h*c*(1/lambda-1/lambda_prime_max);    // Maximum kinetic energy of the recoil electrons, J
printf("\nThe wavelength of scattered X-rays = %5.2e
```

```
    m", lambda_prime);
printf("\nThe maximum wavelength present in the
    scattered X-rays = %6.3f pm", lambda_prime_max/1e
    -012);
printf("\nThe maximum kinetic energy of the recoil
    electrons = %5.3e J", KE_max);

// Result
// The wavelength of scattered X-rays = 1.07e-011 m
// The maximum wavelength present in the scattered X
    -rays = 14.852 pm
// The maximum kinetic energy of the recoil
    electrons = 6.498e-015 J
```

Scilab code Exa 9.2.16 Miller indices for the lattice planes

```
// Scilab Code Ex9.2.16: Page-420(2008)
clc; clear;
m = 3; n = 3; p = 2; // Coefficients of intercepts
    along three axes
m_inv = 1/m;        // Reciprocate the first
    coefficient
n_inv = 1/n;        // Reciprocate the second
    coefficient
p_inv = 1/p;        // Reciprocate the third
    coefficient
mul_fact = double(lcm(int32([m,n,p]))); // Find l.c.
   m. of m,n and p
m1 = m_inv*mul_fact;    // Clear the first fraction
m2 = n_inv*mul_fact;    // Clear the second fraction
m3 = p_inv*mul_fact;    // Clear the third fraction
printf("\nThe miller indices for planes with set of
    intercepts (%da, %db, %dc) are (%d %d %d) ", m, n
    , p, m1, m2, m3);
m = 1; n = 2; p = %inf; // Coefficients of
```

```
intercepts along three axes
m_inv = 1/m;          // Reciprocate the first coefficient
n_inv = 1/n;          // Reciprocate the second coefficient
p_inv = 1/p;          // Reciprocate the third coefficient
mul_fact = double(lcm(int32([m,n]))); // Find l.c.m. of m,n and p
m1 = m_inv*mul_fact;     // Clear the first fraction
m2 = n_inv*mul_fact;     // Clear the second fraction
m3 = p_inv*mul_fact;     // Clear the third fraction
printf("\nThe miller indices for planes with set of intercepts (%da, %db, %dc) are (%d %d %d) ", m, n, p, m1, m2, m3);

// Result
// The miller indices for planes with set of intercepts (3a, 3b, 2c) are (2 2 3)
// The miller indices for planes with set of intercepts (1a, 2b, Infc) are (2 1 0)
```

Scilab code Exa 9.2.19 Glancing angles for the second and third order reflections

```
// Scilab Code Ex9.2.19: Page-421(2008)
clc; clear;
d = 1;     // For simplicity assume interplanar spacing to be unity, m
theta = 15;     // Glancing angle for first order, degree
n = 1;     // Order of reflection
// From Bragg's law, 2*d*sind(theta) = n*lambda, solving for lambda
lambda = 2*d*sind(theta)/n;     // Wavelength of
```

```
incident X-ray, angstrom
// For second order reflection
n = 2
theta = asind(n*lambda/(2*d));     // Glancing angle for second order reflection, degree
printf("\nThe glancing angle for the second order reflection = %4.1f degree", theta);
// For third order reflection
n = 3;
theta = asind(n*lambda/(2*d));     // Glancing angle for third order reflection, degree
printf("\nThe glancing angle for the third order reflection = %4.1f degree", theta);

// Result
// The glancing angle for the second order reflection = 31.2 degree
// The glancing angle for the third order reflection = 50.9 degree
```