Scilab Textbook Companion for Solid State Physics Principles And Applications by R. Asokamani[1]

Created by
Pankaj Biswas
Solid State Physics
Physics
Shri Mata Vaishno Devi University
College Teacher
Dr. Kamni
Cross-Checked by
Lavitha Pereira

July 31, 2019

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

Book Description

Title: Solid State Physics Principles And Applications

Author: R. Asokamani

Publisher: Anamaya Publishers & New Delhi

Edition: 1

Year: 2006

ISBN: 81-88342-07-6

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

List of Figures

Chapter 1

Structure of Solids

Scilab code Exa 1.1 Lattice parameter and atomic radius of fcc structure

```
1 // Scilab Code Ex1.1  Page-13 (2006)
2 clc; clear;
3 r = 1.278e-010;      // Atomic radius of fcc
     structure, m
4 a = 4*r/sqrt(2);     // Lattice parameter of fcc
     strucure, m
5 V = a^3;     // Volume of fcc unit cell, metre, cube
6 printf("\nThe lattice parameter of fcc strucure = %4
     .2e m", a);
7 printf("\nThe volume of fcc unit cell = %5.2e metre,
      cube", V);
8
9 // Result
10 // The lattice parameter of fcc strucure = 3.61e-010
      m
11 // The volume of fcc unit cell = 4.72e-029 metre
     cube
```

Scilab code Exa 1.2 Determining type of niobium cubic structure

```
1  // Scilab Code Ex1.2 Page-14 (2006)
2  clc; clear;
3  r = 0.143e-09;  // Radius of Nb unit cell, m
4  d = 8.57e+03; // Density of Nb unit cell, kg/metre-
       cube
5  M = 92.91e-03;          // Atomic weight of Nb, kg per
        mole
6  N = 6.023D+23;     // Avogadro's No.
7
8  // For fcc
9  a = 4*r/sqrt(2);     // Lattice parameter for fcc
       structure of Nb, m
10 n = a^3*d*N/M; // Number of lattice points per unit
       cell
11 if (modulo(n, int(n)) < 0.001) then
12 printf("\nThe number of atoms associated with the
       cell is %d, Nb should have fcc structure", int(n)
       );
13 end
14
15 // For bcc
16 a = 4*r/sqrt(3);     // Lattice parameter for bcc
       structure of Nb, m
17 n = a^3*d*N/M; // Number of lattice points per unit
       cell
18 if (modulo(n, int(n)) < 0.001) then
19 printf("\nThe number of atoms associated with the
       cell is %d, Nb should have bcc structure", int(n)
       );
20 end
21
22 // Result
23 // The number of atoms associated with the cell is
       2, Nb should have bcc structure
```

Scilab code Exa 1.3 Lattice constants of hcp structure of Ti

```
// Scilab Code Ex1.3 : Page-17 (2006)
clc; clear;
V = 10.58e-29;  // Volume of the unit cell, metre cube
a = poly(0, 'a');       // Declare a variable
a = roots(3*sqrt(3)/2*1.58*a^3-V);        // First lattice parameter, m
c = 1.58*a(3);      // Third lattice parameter, m
printf("\nThe lattice parameters of hcp structure of Ti are:");
printf("\na = %4.2f angstorm, c = %4.2f angstorm", a(3)/1e-010, c/1e-010);

// Result
// The lattice parameters of hcp structure of Ti are :
// a = 2.95 angstorm, c = 4.67 angstorm
```

Scilab code Exa 1.4 c by a c by a ratios of Mg and Cd

```
// Scilab Code Ex1.4 : Page-17 (2006)
clc; clear;
c_by_a_ratio = 1.633;    // Ideal c/a ratio
A = cell(2,4);  // Declare a cell
// Assign values to the elements of the cell from the table
A(1,1).entries = 'Mg';
A(2,1).entries = 'Cd';
A(1,2).entries = 5.21;
A(2,2).entries = 5.62;
A(1,3).entries = 3.21;
A(2,3).entries = 2.98;
A(1,4).entries = A(1,2).entries/A(1,3).entries;
```

```
A(2,4).entries = A(2,2).entries/A(2,3).entries;
if (A(1,4).entries - c_by_a_ratio) < 0.01 then
    printf("\n%s satisfies ideal c/a ratio and %s
       has large deviation from this value.", A(1,1)
       .entries, A(2,1).entries);
else if (A(1,4).entries - c_by_a_ratio) < 0.01 then
    printf("\n%s satisfies ideal c/a ratio and %s
       has large deviation from this value.", A(2,1)
       .entries, A(1,1).entries);
    end
end

// Result
// Mg satisfies ideal c/a ratio and Cd has large
   deviation from this value.
```

Scilab code Exa 1.5 Lattice constant of NaCl unit cell

```
// Scilab Code Ex 1.5 : Page-18 (2006)
clc; clear;5
M_Na = 23;          // Atomic weight of Na, gram per
   mole
M_Cl = 35.5;          // Atomic weight of Cl, gram per
    mole
d = 2.18e+06;     // Density of Nacl salt, g per
   metre cube
n = 4;     // No. of atoms per unit cell for an fcc
   lattice of NaCl crystal
N = 6.023D+23;     // Avogadro's No.
// Volume of the unit cell is given by
// a^3 = M*n/(N*d)
// Solving for a
a = (n*(M_Na + M_Cl)/(d*N))^(1/3);     // Lattice
   constant of unit cell of NaCl
printf("\nLattice constant for the NaCl crystal = %4
```

```
    .2f angstorm", a/1e-010);
13
14 // Result
15 // Lattice constant for the NaCl crystal = 5.63
    angsotrm
```

Scilab code Exa 1.6 Ionic packing factor of fcc KCl

```
1 // Scilab Code Ex 1.6 : Page-18 (2006)
2 clc; clear;
3 r = 1.33;          // Ionic radii of K+ ion, angstrom
4 R = 1.81;        // Ionic radii of Cl- ion, angstrom
5 n = 4;     // No. of atoms per unit cell for an fcc
    lattice of NaCl crystal
6 APF = (n*(4*%pi*r^3/3)+n*(4*%pi*R^3/3))/(2*r+2*R)^3;
        // Atomic packing factor of fcc KCl
7 printf("\nThe ionic packing factor of fcc KCl = %4.2
    f", APF);
8
9 // Result
10 // The ionic packing factor of fcc KCl = 0.56
```

Scilab code Exa 1.7 The number of atoms in unit cells of diamond and graphite

```
1 // Scilab Code Ex 1.7 : Page-20 (2006)
2 clc; clear;
3 N = 6.023e+23;  // Avogadro's number
4 M = 12.01e-03;  // Atomic weight of diamond/graphite
    , kg
5
6 // For diamond
7 a = 3.568e-010;      // Lattice parameter of diamond,
    m
```

```
8  rho = 3.518e+03;     // Density of diamond, kg per
       metre cube
9  n = a^3*rho*N/M;     // Number of atoms in the unit
       cell of diamond structure
10 printf("\nThe number of atoms in the unit cell of
       diamond structure = %1d", n);
11
12 // For graphite
13 a = 2.451e-010;      // First lattice parameter of
       graphite, m
14 c = 6.701e-010;      // Third lattice parameter of
       graphite, m
15 rho = 2.2589e+03;     // Density of graphite, kg per
       metre cube
16 V = 3*sqrt(3)*a^2*c/2;  // Volume of hexagonal unit
       cell of graphite, metre cube
17 n = V*rho*N/M;     // Number of atoms in the unit
       cell of graphite structure
18 printf("\nThe number of atoms in the unit cell of
       graphite structure = %2d", ceil(n));
19
20 // Result
21 // The number of atoms in the unit cell of diamond
       structure = 8
22 // The number of atoms in the unit cell of graphite
       structure = 12
```

Scilab code Exa 1.8 Densities of si and GaAs

```
1  // Scilab Code Ex 1.8 :Page-21 (2006)
2  clc; clear;
3  N = 6.023e+23;  // Avogadro's number
4
5  // For silicon crystallized into diamond structure
6  a = 5.43e-08;   // Lattice parameter of Si, cm
```

```
7 M = 28.1;    // Atomic mass of Si, g/mol
8 n = 8/a^3;        // Number of atoms per unit volume,
     atoms per cm cube
9 d = n*M/N;       // Density of Si crytal, g/cm
10 printf("\nThe density of crystallized Si = %4.2f
     gram per cm cube", d);
11
12 // For GaAs crystallized into Zinc Blende structure
13 a = 5.65e-08;    // Lattice parameter of GaAs, cm
14 M_Ga = 69.7;     // Atomic weight of Ga, g/mol
15 M_As = 74.9;     // Atomic weight of As, g/mol
16 M = M_Ga + M_As;    // Atomic weight of GaAs, g/mol
17 n = 4/a^3;        // Number of atoms per unit volume,
     atoms per cm cube
18 d = n*M/N;       // Density of Si crytal, g/cm
19 printf("\nThe density of crystallized GaAs = %5.3f
     gram per cm cube", d);
20
21 // Result
22 // The density of crystallized Si = 2.33 gram per cm
     cube
23 // The density of crystallized GaAs = 5.324 gram per
     cm cube 12
```

Scilab code Exa 1.9 Lattice parameters of GaP and GaAs

```
1 // Scilab Code Ex 1.9 :Page-21 (2006)
2 clc; clear;
3 N = 6.023e+23;  // Avogadro's number
4
5 r1 = 0.122e-09;      // Ionic radii of Ga, m
6 r2 = 0.125e-09;      // Ionic radii of As, m
7 r3 = 0.11e-09;      // Ionic radii of P, m
8
9 // For GaP
```

```
10 r = r1 + r3;          // Interatomic separation
   between Ga and P atoms, m
11 a = 4*r/3^(1/2);      // Lattice parameter of GaP
   structure, m
12 printf("\nThe lattice parameter of GaP structure =
   %5.3f angstrom", a/1e-10);
13
14 // For GaAs
15 r = r1 + r2;          // Interatomic separation
   between Ga and As atoms, m
16 a = 4*r/3^(1/2);      // Lattice parameter of GaP
   structure, m
17 printf("\nThe lattice parameter of GaAs structure =
   %4.2f angstrom", a/1e-10);
18
19 // Result
20 // The lattice parameter of GaP structure = 5.358
   angstrom
21 // The lattice parameter of GaAs structure = 5.70
   angstrom
```

Scilab code Exa 1.10 Crystal structures of some ionic compounds

```
1  // Scilab Code Ex 1.10 : Page-24 (2006)
2  clc; clear;
3  function str = structure(r_ratio)
4      if r_ratio > 0.732 then
5          str = 'Caesium Chloride';
6      else if  r_ratio < 0.732 & r_ratio > 0.414 then
7              str = 'Rock Salt';
8          else if  r_ratio < 0.414 then
9                  str = 'Rutile'
10             end
11         end
12     end
```

```
13 endfunction
14
15 crystal = cell(6,2);     // Declare cells of 6 rows
      and 2 columns
16 crystal(1,1).entries = 'I';
17 crystal(1,2).entries = 2.19;          // Ionic radius
      of I, angstrom
18 crystal(2,1).entries = 'Cl';
19 crystal(2,2).entries = 1.81;          // Ionic radius
      of Cl, angstrom
20 crystal(3,1).entries = 'Na';
21 crystal(3,2).entries = 0.95;          // Ionic radius
      of Na, angstrom
22 crystal(4,1).entries = 'Cs';
23 crystal(4,2).entries = 1.69;          // Ionic radius
      of Cs, angstrom
24 crystal(5,1).entries = 'Mg';
25 crystal(5,2).entries = 0.99;          // Ionic radius
      of Mg2+, angstrom
26 crystal(6,1).entries = 'O';
27 crystal(6,2).entries = 1.40;          // Ionic radius
      of O2-, angstrom
28
29 printf("\nThe crystal structure of %s%s with radius
      ratio = %6.4f is %s", crystal(3,1).entries,
      crystal(1,1).entries, crystal(3,2).entries/
      crystal(1,2).entries, structure(crystal(3,2).
      entries/crystal(1,2).entries));
30
31 printf("\nThe crystal structure of %s%s with radius
      ratio = %6.4f is %s", crystal(3,1).entries,
      crystal(2,1).entries, crystal(3,2).entries/
      crystal(2,2).entries, structure(crystal(3,2).
      entries/crystal(2,2).entries));
32
33 printf("\nThe crystal structure of %s%s with radius
      ratio = %6.4f is %s", crystal(4,1).entries,
      crystal(2,1).entries, crystal(4,2).entries/
```

```
    crystal(2,2).entries, structure(crystal(4,2).
    entries/crystal(2,2).entries));
34
35 printf("\nThe crystal structure of %s%s with radius
    ratio = %6.4f is %s", crystal(4,1).entries,
    crystal(1,1).entries, crystal(4,2).entries/
    crystal(1,2).entries, structure(crystal(4,2).
    entries/crystal(1,2).entries));
36
37 printf("\nThe crystal structure of %s%s with radius
    ratio = %6.4f is %s", crystal(5,1).entries,
    crystal(6,1).entries, crystal(5,2).entries/
    crystal(6,2).entries, structure(crystal(5,2).
    entries/crystal(2,2).entries));
38
39 // Result
40 //The crystal structure of NaI with radius ratio =
    0.4338 is Rock Salt
41 //The crystal structure of NaCl with radius ratio =
    0.5249 is Rock Salt
42 //The crystal structure of CsCl with radius ratio =
    0.9337 is Caesium Chloride
43 //The crystal structure of CsI with radius ratio =
    0.7717 is Caesium Chloride
44 //The crystal structure of MgO with radius ratio =
    0.7071 is Rock Salt
```

Scilab code Exa 1.11 Maximum radius of the sphere to fit into void between two bcc

```
1 // Scilab Code Ex 1.11 :Page-25 (2006)
2 clc; clear;
3 R = 1;  // For simplicity we assume radius of atom
    to be unity, m
4 // For bcc Structure,
5 a = 4*R/sqrt(3);    // Lattice parameter of bcc
```

```
    crystal , m
6 // We have R+r = a/2, solving for r
7 r = a/2-R    // Relation between radius of the void
    and radius of the atom, m
8 printf("\nThe maxiumum radius of the sphere that can
     fit into void between two bcc unit cells = %5.3
    fR", r);
9
10 // Result
11 // The maxiumum radius of the sphere that can fit
    into void between two bcc unit cells = 0.155R
```

Scilab code Exa 1.12 Maximum radius of the sphere to fit into void between two fcc

```
1 // Scilab Code Ex 1.12 :Page-25 (2006)
2 clc; clear;
3 R = 1;  // For simplicity we assume radius of atom
    to be unity, m
4 // For fcc Structure,
5 a = 4*R/sqrt(2);     // Lattice parameter of fcc
    crystal , m
6 // We have R+r = a/2, solving for r
7 r = a/2-R    // Relation between radius of the void
    and radius of the atom, m
8 printf("\nThe maxiumum radius of the sphere that can
     fit into void between two fcc unit cells = %5.3
    fR", r);
9
10 // Result
11 // The maxiumum radius of the sphere that can fit
    into void between two fcc unit cells = 0.414R
```

Scilab code Exa 1.13 Radius of largest void in the bcc lattice

```
// Scilab Code Ex 1.13 :Page-26 (2006)
clc; clear;
R = 1;  // For simplicity we assume radius of atom
   to be unity, m
// For bcc Structure,
a = 4*R/sqrt(3);    // Lattice parameter of bcc
   crystal, m
// We have (R+r)^2 = (a/2)^2+(a/4)^2, solving for r
r = sqrt(5)*a/4-R   // Relation between radius of
   the void and radius of the atom, m
printf("\nThe radius of largest void in the bcc
   lattice = %4.2fR", r);

// For fcc Structure,
a = 4*R/sqrt(2);    // Lattice parameter of fcc
   crystal, m
// We have (R+r)^2 = (a/2)^2+(a/4)^2, solving for r
r_fcc = a/2-R   // Relation between radius of the
   void and radius of the atom, m
printf("\nThe radius of largest void in the fcc
   lattice is %4.2f times larger than that in the
   bcc lattice", r_fcc/r);

// Result
// The radius of largest void in the bcc lattice =
   0.29R
// The radius of largest void in the fcc lattice is
   1.42 times larger than that in the bcc lattice
```

Scilab code Exa 1.14 Radius of void for carbon atoms in iron

```
// Scilab Code Ex 1.14 :Page-26 (2006)
clc; clear;
R = 1;  // For simplicity we assume radius of atom
   to be unity, m
```

```
4
5 // For bcc Structure,
6 a = 4*R/sqrt(3);     // Lattice parameter of bcc
    crystal, m
7 // We have (R+r)^2 = (a/2)^2+(a/4)^2, solving for r
8 r = a/2-R    // Relation between radius of the void
    and radius of the atom, m
9 printf("\nThe radius of void for carbon atoms in
    iron = %5.3fR", r);
10
11 // Result
12 //The radius of void for carbon atoms in iron =
    0.155R
```

Scilab code Exa 1.15 Radius of triangular void

```
1 // Scilab Code Ex 1.15 :Page-27 (2006)
2 clc; clear;
3 R = 1;  // For simplicity we assume radius of atom
    to be unity, m
4 // From the right triangle LMO, LM/LO = R/(R + r) =
    cosd(30), solving for r
5 r = poly(0, 'r');
6 r = roots(R/cosd(30)-R-r);
7 printf("\nThe radius of triangular void = %5.3fR", r
    );
8
9 // Result
10 // The radius of triangular void = 0.155R
```

Scilab code Exa 1.16 Radius ratio of tetrahedral void

```
1 // Scilab Code Ex 1.16 :Page-27 (2006)
```

```
clc; clear;
R = 1;  // For simplicity we assume radius of atom
    to be unity, m
// From the right triangle LMN similar to trinagle
    LPO, LM/LO = R/(R + r) = LP/LO = sqrt(2/3),
    solving for r
r = poly(0, 'r');
r = roots(R/sqrt(2/3)-R-r);
printf("\nThe radius ratio of tetragonal void = %5.3
    f", r/R);

// Result
// The radius ratio of tetragonal void = 0.225
```

Scilab code Exa 1.17 Radius ratio of octahedral void

```
// Scilab Code Ex 1.17 :Page-28 (2006)
clc; clear;
R = 1;  // For simplicity we assume radius of atom
    to be unity, m
// From the isosceles right triangle LMN, LM/LO = (R
    + r)/R = sqrt(2)/1, solving for r
r = poly(0, 'r');
r = roots(R*sqrt(2)-R-r);
printf("\nThe radius ratio of octahedral void = %5.3
    f", r/R);

// Result
// The radius ratio of octahedral void = 0.414
```

Scilab code Exa 1.18 Miller indices of the crystal plane

```
// Scilab Code Ex 1.18 Page-32 (2006)
```

```
2 clc; clear;
3 p = 3; q = -3; r = 3/2; // Coefficients of
    intercepts along three axes
4 h = 1/p;          // Reciprocate the first coefficient
5 k = 1/q;          // Reciprocate the second
    coefficient
6 l = 1/r;          // Reciprocate the third coefficient
7 mul_fact = double(lcm(int32([p,q,r]))); // Find l.c.
    m. of m,n and p
8 h = h*mul_fact;     // Clear the first fraction
9 k = k*mul_fact;     // Clear the second fraction
10 l = l*mul_fact;    // Clear the third fraction
11 printf("\nThe required miller indices are : (%d %d
    %d) ", h,k,l);
12
13 // Result
14 // The required miller indices are : (1 -1 2)
```

Scilab code Exa 1.19 Miller indices of planes cutting axes of an orthorhombic crys

```
1 // Scilab Code Ex 1.19 Page-32 (2006)
2 clc; clear;
3 p = 2; q = 3; r = 4; // Coefficients of intercepts
    along three axes
4 h = 1/p;          // Reciprocate the first coefficient
5 k = 1/q;          // Reciprocate the second
    coefficient
6 l = 1/r;          // Reciprocate the third coefficient
7 mul_fact = double(lcm(int32([p,q,r]))); // Find l.c.
    m. of m,n and p
8 h = h*mul_fact;     // Clear the first fraction
9 k = k*mul_fact;     // Clear the second fraction
10 l = l*mul_fact;    // Clear the third fraction
11 printf("\nThe required miller indices are : (%d %d
    %d) ", h,k,l);
```

```
12
13 // Result
14 // The required miller indices are : (6 4 3)
```

Scilab code Exa 1.20 Miller indices of set of parallel planes

```
1 // Scilab Code Ex 1.20 Page-32 (2006)
2 clc; clear;
3 p = 4; q = 4; r = %inf; // Coefficients of
     intercepts along three axes
4 h = 1/p;          // Reciprocate the first coefficient
5 k = 1/q;          // Reciprocate the second
     coefficient
6 l = 1/r;          // Reciprocate the third coefficient
7 mul_fact = double(lcm(int32([p,q]))); // Find l.c.m.
     of m,n and p
8 h = h*mul_fact;     // Clear the first fraction
9 k = k*mul_fact;     // Clear the second fraction
10 l = l*mul_fact;     // Clear the third fraction
11 printf("\nThe required miller indices are : (%d %d
     %d) ", h,k,l);
12
13 // Result
14 // The required miller indices are : (1 1 0)
```

Scilab code Exa 1.21 Miller indices of planes with given intercepts

```
1 // Scilab Code Ex 1.21 Page-32 (2006)
2 clc; clear;
3 a = 0.424; b = 2; c = 0.367;     // Intercepts on
     planes along three axes, m
4 // Here pa = 0.424; qb = 2; rc = 0.183, solving for
     p, q and r, we have
```

```
5 p = 0.424/a; q = 2/b; r = 0.183/c; // Coefficients
    of intercepts along three axes
6 h = 1/p;          // Reciprocate the first coefficient
7 k = 1/q;          // Reciprocate the second
    coefficient
8 l = 1/r;          // Reciprocate the third coefficient
9 printf("\nThe required miller indices are : (%d %d
    %d) ", h,k,l);
10
11 // Result
12 // The required miller indices are : (1 1 2)
```

Scilab code Exa 1.22 Interplanar spacing in cubic fcc crystal

```
1 // Scilab Code Ex 1.22 Page-33 (2006)
2 clc; clear;
3 r = 1.746e-010;      // Atomic radius of lead atom,
    angstrom
4 a = 4*r/sqrt(2);     // Interatomic spacing, m
5 h = 1; k = 0; l = 0; // Miller Indices for planes in
    a cubic crystal
6 d_100 = a/(h^2+k^2+l^2)^(1/2);  // The interplanar
    spacing for cubic crystals, m
7 printf("\nThe interplanar spacing between
    consecutive (100) planes = %4.2f angstrom", d_100
    /1e-010);
8
9 h = 1; k = 1; l = 0; // Miller Indices for planes in
    a cubic crystal
10 d_110 = a/(h^2+k^2+l^2)^(1/2);  // The interplanar
    spacing for cubic crystals, m
11 printf("\nThe interplanar spacing between
    consecutive (110) planes = %5.3f angstrom", d_110
    /1e-010);
12
```

```
13 h = 1; k = 1; l = 1; // Miller Indices for planes in
       a cubic crystal
14 d_111 = a/(h^2+k^2+l^2)^(1/2);  // The interplanar
      spacing for cubic crystals, m
15 printf("\nThe interplanar spacing between
      consecutive (111) planes = %4.2f angstrom", d_111
      /1e-010);
16
17 // Result
18 // The interplanar spacing between consecutive (100)
       planes = 4.94 angstrom
19 // The interplanar spacing between consecutive (110)
       planes = 3.492 angstrom
20 // The interplanar spacing between consecutive (111)
       planes = 2.85 angstrom
```

Scilab code Exa 1.23 Wavelength of K alpha radiation of copper using Bohr atom mod

```
1 // Scilab Code Ex 1.23 Page-34 (2006)
2 clc; clear;
3 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
4 h = 6.626e-034;      // Planck's constant, Js
5 c = 3.0e+08;     // Speed of light, m/s
6 E_K = 13.6*29^2;     // Energy of electron in the K-
      shell
7 E_L = 13.6*29^2/4;   // Energy of electron in the L-
      shell
8 // As E_K - E_L = h*c/lambda, solving for lambda
9 lambda = h*c/((E_K - E_L)*e);        // Wavelength of
       K_alpha radiation of tungsten, m
10 printf("\nThe wavelength of K_alpha radiation of Cu
      = %5.3f angstrom", lambda/1e-010);
11
12 // Result
13 // The wavelength of K_alpha radiation of Cu = 1.448
```

```
angstrom
```

Scilab code Exa 1.24 Wavelength of K alpha radiation of tungesten

```
// Scilab Code Ex 1.24 Page-35 (2006)
clc; clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
h = 6.626e-034;      // Planck's constant, Js
c = 3.0e+08;     // Speed of light, m/s
E_K = 13.6*74^2;     // Energy of electron in the K-
    shell
E_L = 13.6*74^2/4;  // Energy of electron in the L-
    shell
// As E_K - E_L = h*c/lambda, solving for lambda
lambda = h*c/((E_K - E_L)*e);        // Wavelength of
     K_alpha radiation of tungsten, m
printf("\nThe wavelength of K_alpha radiation of
    tungsten = %4.2e angstrom", lambda/1e-010);

// Result
// The wavelength of K_alpha radiation of tungsten =
     2.22e-01 angstrom
```

Scilab code Exa 1.25 Lattice constants of copper palladium alloy in different prop

```
// Scilab Code Ex 1.25 Page-35 (2006)
clc; clear;
a_Cu = 3.61;     // Lattice constant of Cu, angstrom
a_Pd = 3.89;     // Lattice constant of Pd, angstrom

// For x = 20% of Pd
x = 0.20;    // Percentage of Pd in Cu-Pd alloy
a_Cu_Pd = ((1-x)*a_Cu + x*a_Pd);
```

```
9  printf("\nFor %2d percent of Pd in Cu-Pd alloy, a =
       %4.2f angstrom", x*100, a_Cu_Pd);
10
11 // For x = 40% of Pd
12 x = 0.40;    // Percentage of Pd in Cu-Pd alloy
13 a_Cu_Pd = ((1-x)*a_Cu + x*a_Pd);
14 printf("\nFor %2d percent of Pd in Cu-Pd alloy, a =
       %5.3f angstrom", x*100, a_Cu_Pd);
15
16 // For x = 60% of Pd
17 x = 0.60;    // Percentage of Pd in Cu-Pd alloy
18 a_Cu_Pd = ((1-x)*a_Cu + x*a_Pd);
19 printf("\nFor %2d percent of Pd in Cu-Pd alloy, a =
       %5.3f angstrom", x*100, a_Cu_Pd);
20
21 // For x = 80% of Pd
22 x = 0.80;    // Percentage of Pd in Cu-Pd alloy
23 a_Cu_Pd = ((1-x)*a_Cu + x*a_Pd);
24 printf("\nFor %2d percent of Pd in Cu-Pd alloy, a =
       %5.3f angstrom", x*100, a_Cu_Pd);
25
26 // Result
27 // For 20 percent of Pd in Cu-Pd alloy, a = 3.67
       angstrom
28 // For 40 percent of Pd in Cu-Pd alloy, a = 3.722
       angstrom
29 // For 60 percent of Pd in Cu-Pd alloy, a = 3.778
       angstrom
30 // For 80 percent of Pd in Cu-Pd alloy, a = 3.834
       angstrom
```

Scilab code Exa 1.26 Amount of required Rh in Pt to change the unit cell volume

```
1 // Scilab Code Ex 1.26 Page-36 (2006)
2 clc; clear;
```

```
3 a_Rh = 3.80;     // Lattice constant of Rh, angstrom
4 a_Pt = 3.92;     // Lattice constant of Pt, angstrom
5 a_Pt_Rh = 3.78;     // Lattice constant of unit cell
    of Pt-Rh alloy, angstrom
6 V = (a_Pt*1e-08)^3; // Volume of unit cell of Pt,
    metre cube
7 V_90 = 0.9*V;    // 90 percent of the cell volume of
    Pt, metre cube
8
9 // For x = 20% of Rh in Pt-Rh alloy, we have
10 // a_Pt_Rh = ((1-x)*a_Pt + x*a_Rh), solving for x
11 x = poly(0, 'x');
12 x = roots (a_Pt_Rh - a_Pt + x*a_Pt - x*a_Rh);
    // Amount of required Rh in Pt to change the unit
     cell volume
13 printf("\nThe amount of Rh required in Pt to change
    the unit cell volume = %4.2f percent", x);
14
15 // Result
16 // The amount of Rh required in Pt to change the
    unit cell volume = 1.17 percent
```

Scilab code Exa 1.27 Percent volume change with the structural change

```
1 // Scilab Code Ex 1.27 :Page-36 (2006)
2 clc; clear;
3 r_bcc = 0.126;     // Atomic radius of the iron
    atoms in the bcc structure, nm
4 r_fcc = 0.129;      // Atomic radius of the iron
    atoms in the fcc structure, nm
5 a_bcc = 4*r_bcc/sqrt(3);
6 a_fcc = 4*r_fcc/sqrt(2);
7 V_bcc = 2*a_bcc^3;      // Volume of bcc unit cell,
    nm cube
8 V_fcc = a_fcc^3;      // Volume of fcc unit cell, nm
```

```
    cube
delta_V = V_fcc - V_bcc;     // Change in volume from
    bcc to fcc structure, nm cube
V = V_bcc;
V_frac = delta_V/V;      // Fractional change in
   volume from bcc to fcc structure

printf("\nThe percentage change in volume from bcc
   to fcc structure = %3.1f percent", V_frac*100);

// Result
// The percentage change in volume from bcc to fcc
   structure = -1.4 percent
```

Chapter 2

Bonding in Solids

Scilab code Exa 2.1 Binding energy of KCl

```
1 // Scilab Code Ex2.1 : Page-62 (2006)
2 clc; clear;
3 epsilon_0 = 8.854e-012; // Absolute electrical
     permittivity of free space, F/m
4 e = 1.6e-019;   // Energy equivalent of 1 eV, eV/J
5 r = 3.147e-010;     // Nearest neighbour distance
     for KCl, m
6 n = 9.1;    // Repulsive exponent of KCl
7 A = 1.748;  // Madelung constant for lattice binding
     energy
8 E = A*e^2/(4*%pi*epsilon_0*r)*(n-1)/n/e;     //
     Binding energy of KCl, eV
9 printf("\nThe binding energy of KCl = %5.3f eV", E);
10
11 // Result
12 // The binding energy of KCl = 7.110 eV
```

Scilab code Exa 2.2 Lattice energy of NaCl

```
// Scilab Code Ex2.2 : Page-62 (2006)
clc; clear;
epsilon_0 = 8.854e-012; // Absolute electrical
   permittivity of free space, F/m
N = 6.023e+023;      // Avogadro's number
e = 1.6e-019;    // Energy equivalent of 1 eV, eV/J
a0 = 5.63e-010;       // Lattice parameter of NaCl, m
r0 = a0/2;       // Nearest neighbour distance for
   NaCl, m
n = 8.4;     // Repulsive exponent of NaCl
A = 1.748;   // Madelung constant for lattice binding
    energy
E = A*e^2/(4*%pi*epsilon_0*r0)*(n-1)/n/e;      //
   Binding energy of NaCl, eV
printf("\nThe binding energy of NaCl = %5.3f kcal/
   mol", E*N*e/(4.186*1e+03));

// Result
// The binding energy of NaCl = 181.101 eV
```

Scilab code Exa 2.3 Nearest neighbour distance of KCl

```
// Scilab Code Ex2.3 : Page-62 (2006)
clc; clear;
epsilon_0 = 8.854e-012; // Absolute electrical
   permittivity of free space, F/m
N = 6.023e+023;      // Avogadro's number
e = 1.6e-019;    // Energy equivalent of 1 eV, eV/J
E = 162.9e+03;   // Binding energy of KCl, cal/mol
n = 8.6;     // Repulsive exponent of KCl
A = 1.747;   // Madelung constant for lattice binding
    energy
// As lattice binding energy, E = A*e^2/(4*%pi*
   epsilon_0*r0)*(n-1)/n, solving for r0
r0 = A*N*e^2/(4*%pi*epsilon_0*E*4.186)*(n-1)/n;
```

```
    // Nearest neighbour distance of KCl, m
11 printf("\nThe nearest neighbour distance of KCl = %4
    .2f angstorm", r0/1e-010);
12
13 // Result
14 // The nearest neighbour distance of KCl = 3.14
    angstorm
```

Scilab code Exa 2.4 Nearest distance of CsCl

```
1  // Scilab Code Ex2.4 : Page-63 (2006)
2  clc; clear;
3  epsilon_0 = 8.854e-012; // Absolute electrical
    permittivity of free space, F/m
4  N = 6.023e+023;      // Avogadro's number
5  e = 1.6e-019;    // Energy equivalent of 1 eV, eV/J
6  E = 152e+03;  // Binding energy of CsCl, cal/mol
7  n = 10.6;     // Repulsive exponent of CsCl
8  A = 1.763;  // Madelung constant for lattice binding
    energy
9  // As lattice binding energy, E = A*e^2/(4*%pi*
    epsilon_0*r0)*(n-1)/n, solving for r0
10 r0 = A*N*e^2/(4*%pi*epsilon_0*E*4.186)*(n-1)/n;
    // Nearest neighbour distance of CsCl, m
11 printf("\nThe nearest neighbour distance of CsCl =
    %4.2f angstrom", r0/1e-010);
12
13 // Result
14 // The nearest neighbour distance of CsCl = 3.48
    angstrom
```

Scilab code Exa 2.5 Repulsive exponent in NaI

```
// Scilab Code Ex2.5 : Page-63 (2006)
clc; clear;
epsilon_0 = 8.854e-012; // Absolute electrical
    permittivity of free space, F/m
N = 6.023e+023;      // Avogadro's number
e = 1.6e-019;    // Energy equivalent of 1 eV, eV/J
r0 = 6.46e-010;      // Nearest neighbour distance of
     NaI
E = 157.1e+03;  // Binding energy of NaI, cal/mol
A = 1.747;  // Madelung constant for lattice binding
     energy
// As lattice binding energy, E = -A*e^2/(4*%pi*
    epsilon_0*r0)*(n-1)/n, solving for n
n = 1/(1+(4.186*E*4*%pi*epsilon_0*r0)/(N*A*e^2));
    // Repulsive exponent of NaI
printf("\nThe repulsive exponent of NaI = %5.3f", n)
    ;

// Result
// The repulsive exponent of NaI = 0.363
```

Scilab code Exa 2.6 Compressibility of solid

```
// Scilab Code Ex2.6 : Page-63 (2006)
clc; clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, eV/J
a0 = 2.815e-010;      // Nearest neighbour distance
    of solid
A = 1.747;  // Madelung constant for lattice binding
     energy
n = 8.6;     // The repulsive exponent of solid
c = 2;  // Structural factor for rocksalt
// As n = 1 + (9*c*a0^4)/(K0*e^2*A), solving for K0
K0 = 9*c*a0^4/((n-1)*e^2*A);         //
    Compressibility of solid, metre square per newton
```

```
10 printf("\nThe compressibility of the solid = %5.3e
     metre square per newton", K0);
11
12 // Result
13 // The compressibility of the solid = 3.325e-001
     metre square per newton (Answer Given in the
     textbook is wrong)
```

Scilab code Exa 2.7 Percentage ionic character present in a solid

```
1 // Scilab Code Ex2.7 : Page-69 (2006)
2 clc; clear;
3 chi_diff = 1;   // Electronegativity difference
     between the constituent of elements of solid
4 percent_ion = 100*(1-exp(-(0.25*chi_diff^2)));  //
     Percentage ionic character present in solid given
      by Pauling
5 printf("\nThe percentage ionic character present in
     solid = %2d percent ", percent_ion);
6
7 // Result
8 // The percentage ionic character present in solid =
      22 percent
```

Scilab code Exa 2.8 Fractional ionicity of compounds

```
1 // Scilab Code Ex2.8 : Page-69 (2006)
2 clc; clear;
3 A = cell(2,3);   // Declare a cell of 3X2
4 A(1,1).entries = 'GaAs';     // First compound name
5 A(1,2).entries = 4.3;        // Homopolar gap of
     first compound, eV
```

```
A(1,3).entries = 2.90;        // Ionic gap of first
   compound, eV
A(2,1).entries = 'CdTe';      // Second compound name
A(2,2).entries = 3.08;         // Homopolar gap of
   second compound, eV
A(2,3).entries = 4.90;        // Ionic gap of second
   compound, eV
printf("\nThe fractional ionicity of the compounds
   are given in the last column of the following
   table:");
printf("\nCompound       Eh       C        fi");
for i = 1:1:2
printf("\n%s           %3.1f      %4.2f     %5.3f", A(i
   ,1).entries, A(i,2).entries, A(i,3).entries, A(i
   ,3).entries^2/(A(i,2).entries^2+A(i,3).entries^2)
   ); // Philips and Vanvechten model of fractional
    ionicity
end

// Result
// The fractional ionicity of the compounds are
   given in the last column of the following table:
// Compound       Eh       C        fi
// GaAs           4.3      2.90     0.313
// sCdTe           3.1      4.90     0.717
```

Chapter 3

Specific Heat of Solids and Lattice Vibrations

Scilab code Exa 3.1 Grunesien parameter for Pb

```
1 // Scilab Code Ex3.1: Page-79 (2006)
2 clc; clear;
3 V0 = 9.1e-05;    // Atomic volume of Pb, metre cube
     per kg
4 K = 2.3e-011;    // Compressibility of Pb, metre
     square per newton
5 alpha = 86e-06; // Coefficient of thermal expansion,
      per K
6 Cv = 1.4e+02;    // Specific heat at constant volume,
      J/kg
7 gama = alpha*V0/(K*Cv);     // Grunesien parameter
     for Pb
8 printf("\nThe Grunesien parameter for Pb = %3.1f",
     gama);
9
10 // Result
11 // The Grunesien parameter for Pb = 2.4
```

Scilab code Exa 3.2 Heat capacity of Cu

```
1 // Scilab Code Ex3.2: Page-79 (2006)
2 clc; clear;
3 V0 = 11e-05;    // Atomic volume of Cu, metre cube
     per kg
4 K = 0.75e-011;    // Compressibility of Cu, metre
     square per newton
5 alpha = 49e-06; // Coefficient of thermal expansion,
      per K
6 gama = 1.9;      // The Grunesien parameter for Cu =
     2.4
7 Cv = alpha*V0/(K*gama);    // Specific heat of Cu at
     constant volume, J/kg
8 printf("\nThe specific heat capacity of Cu = %3.1e J
     /kg", Cv);
9
10 // Result
11 // The specific heat capacity of Cu = 3.8e+02 J/kg
```

Scilab code Exa 3.3 Debye cut off frequency of Al

```
1 // Scilab Code Ex3.3: Page-88 (2006)
2 clc; clear;
3 N = 6.02e+26;  // Avogadro's number, per kmole
4 C_t = 6.32e+03;      // Velocity of transverse wave,
     m/s
5 C_l = 3.1e+03;      // Velocity of longitudinal wave,
      m/s
6 rho = 2.7e+03;      // Density of Al, kg per metre
     cube
7 M = 26.97;  // Atomic weight of Al, gram per mol
```

```
8 V = M/rho;  // Atomic volume of Al, metre cube
9 f_c = (9*N/(4*%pi*V*(1/C_t^3+2/C_l^3)))^(1/3);
10 printf("\nThe Debye cut-off frequency of Al = %4.2e
    per sec", f_c);
11
12 // Result
13 // The Debye cut-off frequency of Al = 8.47e+012 per
    sec
```

Scilab code Exa 3.4 Specific heat capacity of diamond

```
1 // Scilab Code Ex3.4: Page-89 (2006)
2 clc; clear;
3 N = 6.02e+23;  // Avogadro's number, per mole
4 k = 1.38e-023;  // Boltzmann constant, J/K
5 R = N*k;    // Molar gas constant, J/mol/K
6 theta_D = 2230;        // Debye temperature for
    diamond, K
7 T = 300;        // Room temperature, K
8 C_v = 12/5*(%pi^4*R)*(T/theta_D)^3; // Specific heat
    capacity per unit volume of diamond, J/mol-K
9 printf("\nThe heat capacity per unit volume of
    diamond = %4.2f J/mol-K", C_v);
10
11 // Result
12 // The heat capacity per unit volume of diamond =
    4.73 J/mol-K
```

Scilab code Exa 3.5 Debye cut off frequency of Be

```
1 // Scilab Code Ex3.5: Page-89 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/K
```

```
4 theta_D = 1440;          // Debye temperature for Be,
    K
5 h = 6.626e-034;       // Planck's constant, Js
6 f_D = k*theta_D/h;        // Debye cut off frequency
    of Be, Hz
7 printf("\nThe Debye cut off frequency of Be = %g per
    sec", f_D);
8
9 // Result
10 // The Debye cut off frequency of Be = 2.99909e+013
    per sec
```

Scilab code Exa 3.6 Electronic and lattice heat capacities of Cu

```
1 // Scilab Code Ex3.6: Page-89 (2006)
2 clc; clear;
3 N = 6.023e+023; // Avogadro's number, per kmol
4 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;   // Boltzmann constant, J/K
6 R = N*k;      // Molar gas constant, J/kmol/K
7 E_F = 7;      // Fermi energy of Cu, eV
8 theta_D = 348;   // Debye temperature of Cu, K
9 T = 300;      // Room temperature, K
10 T_F = E_F/k;     // Fermi temperature of Cu, K
11 C_e = %pi^2/2*R*1e+03*(T/(T_F*e));     // Electronic
    heat capacity of Cu, J/kmol/K
12 C_l = 12/5*(%pi^4*R)*(T/theta_D)^3;      // Lattice
    heat capacity of Cu, J/kmol/K
13 printf("\nThe electronic heat capacity of Cu = %3d J
    /kmol/K", round(C_e));
14 printf("\nThe lattice heat capacity of Cu = %4.2e J/
    mol/K", C_l);
15
16 // Result
17 // The electronic heat capacity of Cu = 152 J/kmol/K
```

```
18 // The lattice heat capacity of Cu = 1.24e+003 J/mol
   /K
```

Scilab code Exa 3.7 Heat capacities of Cu at a given temperature

```
1  // Scilab Code Ex3.7: Page-90 (2006)
2  clc; clear;
3  N = 6.023e+023; // Avogadro's number, per kmol
4  e = 1.602e-019;   // Energy equivalent of 1 eV, J/eV
5  k = 1.38e-023;  // Boltzmann constant, J/K
6  R = N*k;     // Molar gas constant, J/kmol/K
7  E_F = 7;     // Fermi energy of Cu, eV
8  theta_D = 348;  // Debye temperature of Cu, K
9  T = 0.01;     // Room temperature, K
10 T_F = E_F/k;     // Fermi temperature of Cu, K
11 C_e = %pi^2/2*R*(T/(T_F*e));     // Electronic heat
   capacity of Cu, J/mol/K
12 C_l = 12/5*(%pi^4*R)*(T/theta_D)^3;     // Lattice
   heat capacity of Cu, J/kmol/K
13 printf("\nThe electronic heat capacity of Cu = %4.2e
    J/mol/K", C_e);
14 printf("\nThe lattice heat capacity of Cu = %3.1e J/
   mol/K", C_l);
15
16 // Result
17 // The electronic heat capacity of Cu = 5.05e-006 J/
   mol/K
18 // The lattice heat capacity of Cu = 4.6e-011 J/mol/
   K
```

Scilab code Exa 3.8 Electronic specific heat of Na at 20 K

```
1  // Scilab Code Ex3.8: Page-90 (2006)
```

```
2 clc; clear;
3 N = 6.023e+023; // Avogadro's number, per kmol
4 e = 1.602e-019;   // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;  // Boltzmann constant, J/K
6 R = N*k;    // Molar gas constant, J/kmol/K
7 E_F = 3.2;    // Fermi energy of Cu, eV
8 theta_D = 150;  // Debye temperature of Cu, K
9 T = 20;    // Given temperature, K
10 T_F = E_F/k;    // Fermi temperature of Cu, K
11 C_e = %pi^2/2*R*(T/(T_F*e));    // Electronic heat
   capacity of Cu, J/mol/K
12 C_l = 12/5*(%pi^4*R)*(T/theta_D)^3;     // Lattice
   heat capacity of Cu, J/kmol/K
13 printf("\nThe electronic heat capacity of Na = %5.3e
    J/mol/K", C_e);
14 printf("\nThe lattice heat capacity of Na = %6.4e J/
   mol/K", C_l);
15
16 // Result
17 // The electronic heat capacity of Na = 2.208e-002 J
   /mol/K
18 // The lattice heat capacity of Na = 4.6059e+000 J/
   mol/K
```

Scilab code Exa 3.9 Lattice specific heat of Hf

```
1 // Scilab Code Ex3.9: Page-91 (2006)
2 clc; clear;
3 N = 6.023e+023; // Avogadro's number, per kmol
4 e = 1.602e-019;   // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;  // Boltzmann constant, J/K
6 R = N*k;    // Molar gas constant, J/kmol/K
7 E_F = 3.2;    // Fermi energy of Hf, eV
8 theta_D = 242;  // Debye temperature of Hf, K
9 T_F = E_F/k;    // Fermi temperature of Hf, K
```

```
10 T = [300, 200, 100, 10, 5]; // Declare a vector of 5
       temperature values, K
11 printf("\n-------------------------");
12 printf("\nT(K)       C_l (J/kmol/K)");
13 printf("\n-------------------------")
14 for i = 1:1:5
15     C_l = 12/5*(%pi^4*R)*(T(i)/theta_D)^3;      //
          Lattice heat capacity of Hf, J/kmol/K
16     printf("\n%3d        %8.3f", T(i), C_l);
17 end
18 printf("\n-------------------------")
19
20 // Result
21 // -------------------------
22 // T(K)       C_l (J/kmol/K)
23 // -------------------------
24 // 300        3701.863
25 // 200        1096.848
26 // 100         137.106
27 //  10           0.137
28 //   5           0.017
29 // -------------------------
```

Scilab code Exa 3.10 Temperature at which lattice specific heat equals electronic

```
1 // Scilab Code Ex3.10: Page-91 (2006)
2 clc; clear;
3 N = 6.023e+023; // Avogadro's number, per kmol
4 e = 1.602e-019;   // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;  // Boltzmann constant, J/K
6 R = N*k;     // Molar gas constant, J/kmol/K
7 E_F = 7;     // Fermi energy of Hf, eV
8 theta_D = 343;  // Debye temperature of Hf, K
9 T_F = E_F/k;     // Fermi temperature of Hf, K
10 // As C_l = 12/5*(%pi^4*R)*(T/theta_D)^3 and C_e =
```

```
    %pi^2/2*R*(T/(T_F*e)) so that
// For C_l = C_e, we have
T = sqrt((%pi^2/2*R*1/(T_F*e))/(12/5*%pi^4*R)*
    theta_D^3); // Required temperature when C_l =
    C_e, K
printf("\nThe temperature at which lattice specific
    heat equals electronic specific heat for Cu = %4
    .2f K", T);

// Result
// The temperature at which lattice specific heat
    equals electronic specific heat for Cu = 3.24 K
```

Scilab code Exa 3.11 Debye temperature of Al

```
// Scilab Code Ex3.11: Page-92 (2006)
clc; clear;
C11 = 1.08e+12, C12 = 0.62e+12, C44 = 0.28e+12;
    // Elastic constants of Al, dynes/cm square
a = 4.05e-08;    // Lattice constant for Al cubic
    structure, cm
rho = 2.70;      // g/cm cube
k = 1.38e-023;   // Boltzmann constant, J/K
h = 6.626e-034; // Planck's constant, Js
s = 4;  // Number of atoms in Al unit cell
Va = a^3;    // Volume of unit cell, cm cube
theta_D = (3.15/(8*%pi)*(h/k)^3*s/(rho^(3/2)*Va)*(
    C11-C12)^(1/2)*(C11+C12+2*C44)^(1/2)*C44^(1/2))
    ^(1/3);
printf("\nThe Debye temperature of Al = %3d K",
    theta_D);

// Result
// The Debye temperature of Al = 466 K
```

Scilab code Exa 3.12 Debye temperatures of Cu and Na

```
1  // Scilab Code Ex3.12: Page-93 (2006)
2  clc; clear;
3  k = 1.38e-023;  // Boltzmann constant, J/K
4  h = 6.626e-034; // Planck's constant, Js
5  A = cell(2,8);  // Declare a matrix of 2X8
6  A(1,1).entries = 'Cu';
7  A(1,2).entries = 1.684e+012;
8  A(1,3).entries = 1.214e+012;
9  A(1,4).entries = 0.754e+012;
10 A(1,5).entries = 4;
11 A(1,6).entries = 3.61e-08;
12 A(1,7).entries = 8.96;
13 A(2,1).entries = 'Na';
14 A(2,2).entries = 0.055e+012;
15 A(2,3).entries = 0.047e+012;
16 A(2,4).entries = 0.049e+012;
17 A(2,5).entries = 2;
18 A(2,6).entries = 4.225e-08;
19 A(2,7).entries = 0.971;
20
21 // For Cu
22 Va = A(1,6).entries^3;    // Volume of unit cell, cm
      cube
23 A(1,8).entries = (3.15/(8*%pi)*(h/k)^3*A(1,5).
      entries/(A(1,7).entries^(3/2)*Va)*(A(1,2).entries
      -A(1,3).entries)^(1/2)*(A(1,2).entries+A(1,3).
      entries+2*A(1,4).entries)^(1/2)*A(1,4).entries
      ^(1/2))^(1/3);
24
25 // For Na
26 Va = A(2,6).entries^3;    // Volume of unit cell, cm
      cube
```

```
27 A(2,8).entries = (3.15/(8*%pi)*(h/k)^3*A(2,5).
      entries/(A(2,7).entries^(3/2)*Va)*(A(2,2).entries
      -A(2,3).entries)^(1/2)*(A(2,2).entries+A(2,3).
      entries+2*A(2,4).entries)^(1/2)*A(2,4).entries
      ^(1/2))^(1/3);
28
29 printf("\n-----------------------------------------")
      ;
30 printf("\nMetal      C11      C12      C44      thetaD")
31 printf("\n-----------------------------------------")
      ;
32 for i = 1:1:2
33     printf("\n%s          %5.3f    %5.3f    %5.3f    %3d"
          , A(i,1).entries, A(i,2).entries/1e+12, A(i
          ,3).entries/1e+12, A(i,4).entries/1e+12, A(i
          ,8).entries);
34 end
35 printf("\n-----------------------------------------")
      ;
36
37 // Result
38 // -----------------------------------------
39 // Metal      C11      C12      C44      thetaD
40 // -----------------------------------------
41 // Cu         1.684    1.214    0.754    380
42 // Na         0.055    0.047    0.049    150
43 // -----------------------------------------
```

Scilab code Exa 3.13 Debye temperature as a function of temperature

```
1 // Scilab Code Ex3.13: Page-93 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/K
4 h = 6.626e-034; // Planck's constant, Js
5 A = cell(4,5);  // Declare a matrix of 4X5
```

```
A(1,1).entries = 300;
A(1,2).entries = 0.878e+010;
A(1,3).entries = 0.483e+010;
A(1,4).entries = 0.448e+010;
A(2,1).entries = 200;
A(2,2).entries = 0.968e+010;
A(2,3).entries = 0.508e+010;
A(2,4).entries = 0.512e+010;
A(3,1).entries = 100;
A(3,2).entries = 1.050e+010;
A(3,3).entries = 0.540e+010;
A(3,4).entries = 0.579e+010;
A(4,1).entries = 20;
A(4,2).entries = 1.101e+010;
A(4,3).entries = 0.551e+010;
A(4,4).entries = 0.624e+010;
s = 2;       // Number of atoms in a unit cell
a = 4.225e-10;  // Lattice parameter of Na, m
rho = 0.971e+03;     // Density of Na, kg/metre-cube
Va = a^3;    // Volume of unit cell, metre cube
printf("\n-----------------------------------------")
   ;
printf("\nT        C11      C12      C44      thetaD")
printf("\n-----------------------------------------")
   ;
for i=1:1:4
   A(i,5).entries = (3.15/(8*%pi)*(h/k)^3*s/(rho
      ^(3/2)*Va)*(A(i,2).entries-A(i,3).entries)
      ^(1/2)*(A(i,2).entries+A(i,3).entries+2*A(i,4)
      .entries)^(1/2)*A(i,4).entries^(1/2))^(1/3);
printf("\n%3d      %5.3f    %5.3f    %5.3f    %3d", A(i
   ,1).entries, A(i,2).entries/1e+10, A(i,3).entries
   /1e+10, A(i,4).entries/1e+10, A(i,5).entries);
end
printf("\n-----------------------------------------")
   ;

// Result
```

```
36 // ----------------------------------------
37 // T       C11     C12     C44     thetaD
38 // ----------------------------------------
39 // 300     0.878   0.483   0.448   197
40 // 200     0.968   0.508   0.512   210
41 // 100     1.050   0.540   0.579   222
42 //  20     1.101   0.551   0.624   229
43 // ----------------------------------------
44 // The theta values given in the textbook are wrong
```

Scilab code Exa 3.14 Variation of Gruneisen frequency and Debye temperature for Lu

```
1  // Scilab Code Ex3.12: Page-93 (2006)
2  clc; clear;
3  Lu = cell(6,5);  // Declare a matrix of 6X5
4  Lu(1,1).entries = 0;
5  Lu(1,2).entries = 5.58;
6  Lu(1,3).entries = 3.517;
7  Lu(1,5).entries = 0.750;
8  Lu(2,1).entries = 36;
9  Lu(2,2).entries = 5.409;
10 Lu(2,3).entries = 3.440;
11 Lu(2,5).entries = 0.560;
12 Lu(3,1).entries = 103;
13 Lu(3,2).entries = 5.213;
14 Lu(3,3).entries = 3.341;
15 Lu(3,5).entries = 0.492;
16 Lu(4,1).entries = 157;
17 Lu(4,2).entries = 5.067;
18 Lu(4,3).entries = 3.259;
19 Lu(4,5).entries = 0.388;
20 Lu(5,1).entries = 191;
21 Lu(5,2).entries = 4.987;
22 Lu(5,3).entries = 3.217;
23 Lu(5,5).entries = 0.357;
```

```
24 Lu(6,1).entries = 236;
25 Lu(6,2).entries = 4.921;
26 Lu(6,3).entries = 3.179;
27 Lu(6,5).entries = 0.331;
28 V0 = 3*sqrt(3)/2*Lu(1,3).entries^2*Lu(1,2).entries;
29 V = zeros(6);    // Declare volume array
30 printf("\
      n----------------------------------------------------------------
      ");
31 printf("\nP(kbar)     c(angstrom)      a(angstrom)
      gamma_G       nu_G    ");
32 printf("\
      n----------------------------------------------------------------
      ");
33 for i=1:1:6
34     V(i) = 3*sqrt(3)/2*Lu(i,3).entries^2*Lu(i,2).
          entries;
35     Lu(i,4).entries = Lu(i,5).entries*V(i)/V0+2/3*(1-
          V(i)/V0)^(1/2);
36 printf("\n%3d          %5.3f             %5.3f
      %5.3f          %5.3f", Lu(i,1).entries, Lu(i,2).
      entries, Lu(i,3).entries, Lu(i,4).entries, Lu(i
      ,5).entries);
37 end
38 printf("\
      n----------------------------------------------------------------
      ");
39
40 cnt = 0;
41 printf("\n------------------------");
42 printf("\nP(kbar)        Theta_D(K)");
43 printf("\n------------------------");
44 for i=1:1:6
45     theta_D = exp(integrate('-1*Lu(i,5).entries*exp(
          x)/V0-2/3*(1-exp(x)/V0)^(1/2)', 'x', -0.8+cnt
          , log(V(i)/1000000)));
46     cnt = cnt + 0.01;
47     printf("\n%3d              %3.0f", Lu(i,1).
```

```
        entries, theta_D);
48 end
49 printf("\n-------------------------");
50
51 // Result
52 //
    ------------------------------------------------------------------

53 // P(kbar)     c(angstrom)      a(angstrom)     gamma_G
         nu_G
54 //
    ------------------------------------------------------------------

55 //   0          5.580            3.517           0.750
          0.750
56 //  36          5.409            3.440           0.699
          0.560
57 // 103          5.213            3.341           0.679
          0.492
58 // 157          5.067            3.259           0.615
          0.388
59 // 191          4.987            3.217           0.602
          0.357
60 // 236          4.921            3.179           0.591
          0.331
61 //
    ------------------------------------------------------------------

62 // -------------------------
63 // P(kdbar)        Theta_D(K)
64 // -------------------------
65 //   0               185
66 //  36               195
67 // 103               210
68 // 157               222
69 // 191               230
70 // 236               237
71 // -------------------------
```

Scilab code Exa 3.15 Lindemann rule to calculate the Debye temperature

```
1 // Scilab Code Ex3.15: Page-94 (2006)
2 clc; clear;
3 T_M = 1356;      // Melting temperature of Cu, K
4 V = 7.114;       // Atomic volume of Cu, cm cube per
     g-atom
5 M = 63.5;        // atomic weight of Cu, g/mole
6 K = 138.5;       // Lindemann constant
7 theta_M = K*(T_M/M)^(1/2)*(1/V)^(1/3); // Debye
     temperature by Lindemann method, K
8
9 printf("\nThe Debye temperature by Lindemann method
     = %3d K", ceil(theta_M));
10 printf("\nThe values obtained from other methods are
     :");
11 printf("\ntheta_s = 342 K;       theta_R = 336 K;
           theta_E = 345 K");
12
13 // Result
14 // The Debye temperature by Lindemann method = 333 K
15 // The values obtained from other methods are:
16 // theta_s = 342 K;       theta_R = 336 K;
     theta_E = 345 K
```

Scilab code Exa 3.16 Frequency of vibration of ions in InSb crystal

```
1 // Scilab Code Ex3.16: Page-100 (2006)
2 clc; clear;
3 N_A = 6.023e+023;        // Avogadro's number
4 c = 3.0e+08;     // Speed of light, m/s
```

```
epsilon_0 = 15; // Dielectric constant of the medium
m = 2.0e-022;        // Mass of ion, g
e = 4.8e-010;    // Charge on the ion, C
rho = 7;     // Average density of solid, g/cc
A = 120;     // Average atomic weight of solid, g
N = rho/A*N_A;  // Number of ions per cc, per cm cube
f_P = 1/(2*%pi)*sqrt(4*%pi*N*e^2/(m*epsilon_0));
        // Plasma frequency of vibrating ions in the crystal, Hz
lambda_P = c/f_P;    // Plasma wavelength of vibrating ions in the crystal, cm
printf("\nThe plasma frequency of vibrating ions in InSb crystal = %3.1e Hz", f_P);
printf("\nThe plasma wavelength of vibrating ions in InSb crystal = %3d micron", lambda_P/1e-06);
printf("\nThe calculated frequency lies in the infrared region.")

// Result
// The plasma frequency of vibrating ions in InSb crystal = 9.3e+011 Hz
// The plasma wavelength of vibrating ions in InSb crystal = 323 micron
// The calculated frequency lies in the infrared region.
```

Scilab code Exa 3.17 Debye temperature for diamond

```
// Scilab Code Ex3.17: Page-103 (2006)
clc; clear;
h = 6.624e-034;      // Planck's constant, Js
k = 1.38e-023;  // Boltzmann constant, J/mol/K
q = 1.486e+011;      // Young's modulus of diamond, N/metre-square
```

```
6  rho = 3500;      // Density of diamond, kg/metre-cube
7  c = sqrt(q/rho);     // Speed of transverse wave
      through diamond, m/s
8  m = 12*1.66e-027;       // Atomic weight of carbon,
      kg
9  theta_D = (h/k)*c*(3*rho/(4*%pi*m))^(1/3);     //
      Debye temperature for diamond, K
10 printf("\nThe Debye temperature for diamond = %4d K"
      , theta_D);
11
12 // Result
13 // The Debye temperature for diamond = 1086 K
```

Chapter 4

Free Electron Theory of Metals

Scilab code Exa 4.1 Collision time for an electron in monovalent Cu

```
1 // Scilab Code Ex4.1: Page-112 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of an electron, kg
4 e = 1.6e-019;    // Charge on an electron, C
5 n = 8.5e+028;    // Concentration of electron in Cu,
     per metre cube
6 rho = 1.7e-08;   // Resistivity of Cu, ohm-m
7 t = m/(n*e^2*rho);  // Collision time for an
     electron in monovalent Cu, s
8 printf("\nThe collision time for an electron in
     monovalent Cu = %3.1e s", t);
9
10 // Result
11 // The collision time for an electron in monovalent
     Cu = 2.5e-014 s
```

Scilab code Exa 4.2 Relaxation time and mean free path at 0K

```
// Scilab Code Ex4.2: Page-112 (2006)
clc; clear;
m = 9.1e-031;    // Mass of an electron, kg
e = 1.6e-019;    // Charge on an electron, C
n = 1e+029;    // Concentration of electron in
   material, per metre cube
rho = 27e-08;  // Resistivity of the material, ohm-m
tau = m/(n*e^2*rho);  // Collision time for an
   electron in the material, s
v_F = 1e+08;     // Velocity of free electron, cm/s
lambda = v_F*tau;    // Mean free path of electron in
    the material, cm
printf("\nThe collision time for an electron in
   monovalent Cu = %3.1e s", tau);
printf("\nThe mean free path of electron at 0K = %3
   .1e cm", lambda);

// Result
// The collision time for an electron in monovalent
   Cu = 1.3e-015 s
// The mean free path of electron at 0K = 1.3e-007
   cm
```

Scilab code Exa 4.3 Free electron density and electrical conductivity of monovalen

```
// Scilab Code Ex4.3: Page-112 (2006)
clc; clear;
m = 9.1e-031;    // Mass of an electron, kg
e = 1.6e-019;    // Charge on an electron, C
r = 1.28e-010;  // Atomic radius of cupper, m
a = 4*r/sqrt(2);     // Lattice parameter of fcc
   structure of Cu, m
V = a^3;     // Volume of unit cell of Cu, metre cube
n = 4/V;     // Number of atoms per unit volume of Cu
   , per metre cube
```

```
9  tau = 2.7e-04;  // Relaxation time for an electron
     in monovalent Cu, s
10 sigma = n*e^2*tau/m;    // Electrical conductivity
     of Cu, mho per cm
11 printf("\nThe free electron density in monovalent Cu
     = %5.3e per metre cube", n);
12 printf("\nThe electrical conductivity of monovalent
     Cu = %5.3e mho per cm", sigma);
13
14 // Result
15 // The free electron density in monovalent Cu =
     8.429e+028 per metre cube
16 // The electrical conductivity of monovalent Cu =
     6.403e+017 mho per cm
```

Scilab code Exa 4.4 Energy difference between two levels for the free electrons

```
1  // Scilab Code Ex4.4: Page-118 (2006)
2  clc; clear;
3  m = 9.1e-031;   // Mass of an electron, kg
4  e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
5  h = 6.625e-034;    // Planck's constant, Js
6  L = 10e-03;     // Length of side of the cube, m
7  // For nth level
8  nx = 1, ny = 1, nz = 1;     // Positive integers
     along three axis
9  En = h^2/(8*m*L^2)*(nx^2+ny^2+nz^2)/e;      //
     Energy of nth level for electrons, eV
10 // For (n+1)th level
11 nx = 2, ny = 1, nz = 1;     // Positive integers
     along three axis
12 En_plus_1 = h^2/(8*m*L^2)*(nx^2+ny^2+nz^2)/e;
     // Energy of (n+1)th level for electrons, eV
13 delta_E = En_plus_1 - En;       // Energy difference
     between two levels for the free electrons
```

```
14 printf("\nThe energy difference between two levels
     for the free electrons = %4.2e eV", delta_E);
15
16 // Result
17 // The energy difference between two levels for the
     free electrons = 1.13e-014 eV
```

Scilab code Exa 4.5 Probability of the electron in tungsten at room temperature

```
1 // Scilab Code Ex4.5: Page-119 (2006)
2 clc; clear;
3 T = 300;     // Room temperature of tungsten, K
4 k = 1.38e-023;  // Boltzmann constant, J/mol/K
5 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
6 E_F = 4.5*e;    // Fermi energy of tungsten, J
7 E = E_F-0.1*E_F; // 10% energy below Fermi energy, J
8 f_T = 1/(1+exp((E-E_F)/(k*T)));      // Probability
     of the electron in tungsten at room temperature
     at an nergy 10% below the Fermi energy
9 printf("\nThe probability of the electron at an
     energy 10 percent below the Fermi energy in
     tungsten at 300 K = %4.2f", f_T);
10 E = 2*k*T+E_F; // For energy equal to 2kT + E_F
11 f_T = 1/(1+exp((E-E_F)/(k*T)));      // Probability
     of the electron in tungsten at an energy 2kT
     above the Fermi energy
12 printf("\nThe probability of the electron at an
     energy 2kT above the Fermi energy = %6.4f", f_T);
13
14 // Result
15 // The probability of the electron at an energy 10
     percent below the Fermi energy in tungsten at 300
      K = 1.00
16 // The probability of the electron at an energy 2kT
     above the Fermi energy = 0.1192
```

Scilab code Exa 4.6 Fermi energy of a monovalent bcc solid

```
1 // Scilab Code Ex4.6: Page-121 (2006)
2 clc; clear;
3 h = 6.625e-034; // Planck's constant, Js
4 h_cross = h/(2*%pi);     // Reduced Planck's constant
    , Js
5 m = 9.1e-031;    // Mass of an electron, kg
6 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
7 a = 5.34e-010;       // Lattice constant of
    monovalent bcc lattice, m
8 V = a^3;     // Volume of bcc unit cell, metre cube
9 n = 2/V;     // Number of atoms per metre cube
10 E_F = h_cross^2/(2*m*e)*(3*%pi^2*n)^(2/3);     //
    Fermi energy of monovalent bcc solid, eV
11
12 printf("\nThe Fermi energy of a monovalent bcc solid
    = %5.3f eV", E_F);
13
14 // Result
15 // The Fermi energy of a monovalent bcc solid =
    2.034
```

Scilab code Exa 4.7 Number of states at Fermi energy

```
1 // Scilab Code Ex4.7: Page-121 (2006)
2 clc; clear;
3 h = 6.625e-034; // Planck's constant, Js
4 h_cross = h/(2*%pi);     // Reduced Planck's constant
    , Js
5 m = 9.11e-031;    // Mass of an electron, kg
```

```
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
V = 1e-05;     // Volume of cubical box, metre cube
E_F = 5*e;  // Fermi energy, J
D_EF = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*E_F^(1/2)*e;      // Density of states at Fermi energy, states/eV
printf("\nThe density of states at Fermi energy = %4.2e states/eV", D_EF);

// Result
// The density of states at Fermi energy = 1.52e+023 states/eV
```

Scilab code Exa 4.8 Energy separation between adjacent energy levels of Mg and Ca

```
// Scilab Code Ex4.8: Page-121 (2006)
clc; clear;
h = 6.626e-034; // Planck's constant, Js
h_cross = h/(2*%pi);     // Reduced Planck's constant, Js
m = 9.1e-031;    // Mass of an electron, kg
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
V = 1e-06;     // Volume of cubical box, metre cube
E_F = 7.13*e;  // Fermi energy for Mg, J
D_EF = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*E_F^(1/2);        // Density of states at Fermi energy for Cs, states/eV
E_Mg = 1/D_EF;  // The energy separation between adjacent energy levels of Mg, J
printf("\nThe energy separation between adjacent energy levels of Mg = %5.3e eV", E_Mg/e);
E_F = 1.58*e;  // Fermi energy for Cs, J
D_EF = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*E_F^(1/2);        // Density of states at Fermi energy for Mg, states/eV
```

```
14 E_Mg = 1/D_EF;  // The energy separation between
      adjacent energy levels of Cs, J
15 printf("\nThe energy separation between adjacent
      energy levels of Cs = %5.3e eV", E_Mg/e);
16
17 // Result
18 // The energy separation between adjacent energy
      levels of Mg = 5.517e-023 eV
19 // The energy separation between adjacent energy
      levels of Cs = 1.172e-022 eV
```

Scilab code Exa 4.9 Fermi momentum of sodium

```
1 // Scilab Code Ex4.9: Page-122 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of an electron, kg
4 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
5 E_F = 3.2*e;     // Fermi energy of sodium, J
6 P_F = sqrt(E_F*2*m);     // Fermi momentum of sodium,
     kg-m/s
7 printf("\nThe Fermi momentum of sodium = %5.3e kg-m/
   sec", P_F);
8
9 // Result
10 // The Fermi momentum of sodium = 9.653e-025 kg-m/
   sec
```

Scilab code Exa 4.10 Change in Fermi energy with temperature

```
1 // Scilab Code Ex4.10: Page-122 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/mol/K
4 T = 500;     // Rise in temperature of Al, K
```

```
5 EF_0 = 11.63;     // Fermi energy of Al, eV
6 EF_T = EF_0*(1-%pi^2/12*(k*T/EF_0)^2);  // Change in
    Fermi energy of Al with temperature, eV
7 printf("\nThe change in Fermi energy of Al with
    tempertaure rise of 500 degree celsius = %5.2f eV
    ", EF_T);
8
9 // Result
10 // The change in Fermi energy of Al with tempertaure
    rise of 500 degree celsius = 11.63 eV
```

Scilab code Exa 4.11 Electrical conductivity of Pb and Ag

```
1 // Scilab Code Ex4.11: Page-122 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of an electron, kg
4 e = 1.6e-019;    // Charge on an electron, C
5 lambda = 1.0e-09;    // Mean free path of electron in
    metal, m
6 v = 1.11e+05;     // Average velocity of the electron
    in metal, m/s
7
8 // For Lead
9 n = 13.2e+028;  // Electronic concentration of Pb,
    per metre cube
10 sigma = n*e^2*lambda/(m*v);      // Electrical
    conductivity of lead, mho per metre
11 printf("\nThe electrical conductivity of lead = %4.2
    e mho per metre", sigma);
12
13 // For Silver
14 n = 5.85e+28;  // Electronic concentration of Ag,
    per metre cube
15 sigma = n*e^2*lambda/(m*v);      // Electrical
    conductivity of Ag, mho per metre
```

```
16 printf("\nThe electrical conductivity of silver = %4
   .2e mho per metre", sigma);
17
18 // Result
19 // The electrical conductivity of lead = 3.35e+007
   mho per metre
20 // The electrical conductivity of silver = 1.48e+007
    mho per metre
```

Scilab code Exa 4.12 Lorentz number

```
1 // Scilab Code Ex4.12: Page-125 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/mol/K
4 e = 1.6e-019;   // Charge on an electron, C
5 L = %pi^2/3*(k/e)^2;    // Lorentz number, watt-ohm/
   degree-square
6 printf("\nThe Lorentz number = %4.2e watt-ohm/degree
   -square", L);
7
8 // Result
9 // The Lorentz number = 2.45e-008 watt-ohm/degree-
   square
```

Scilab code Exa 4.13 Lorentz numbers for metals at 273 K and comparison with the g

```
1 // Scilab Code Ex4.13: Page-125 (2006)
2 clc; clear;
3 A = cell(4,4);      // Declare a 4X4 cell
4 A(1,1).entries = 'Mg';
5 A(1,2).entries = 2.54e-05;
6 A(1,3).entries = 1.5;
7 A(1,4).entries = 2.32e+02;
```

```
A(2,1).entries = 'Cu';
A(2,2).entries = 6.45e-05;
A(2,3).entries = 3.85;
A(2,4).entries = 2.30e+02;
A(3,1).entries = 'Al';
A(3,2).entries = 4.0e-05;
A(3,3).entries = 2.38;
A(3,4).entries = 2.57e+02;
A(4,1).entries = 'Pt';
A(4,2).entries = 1.02e-05;
A(4,3).entries = 0.69;
A(4,4).entries = 2.56e+02;
T1 = 273;    // First temperature, K
T2 = 373;    // Second temperature, K
printf("\
   n----------------------------------------------------------------------
   ");
printf("\nMetal      sigma x 1e-05    K(W/cm-K)
   Lorentz number                ");
printf("\n           (mho per cm)                  (
   watt-ohm/deg-square)x1e-02")
printf("\
   n----------------------------------------------------------------------
   ");
for i = 1:1:4
    L1 = A(i,3).entries/(A(i,2).entries*T1); L2 = A(
       i,4).entries;
    printf("\n%s         %4.2f              %4.2f
                %4.2f              %4.2f", A(i,1).
       entries, A(i,2).entries/1e-05, A(i,3).entries
       , L1/1e+02, L2/1e+02);
end
printf("\
   n----------------------------------------------------------------------
   ");

// Result
//
```

```
34 // Metal      sigma x 1e-05   K(W/cm-K)    Lorentz
   number
35 //            (mho per cm)                (watt-ohm/
   deg-square)x1e-02
36 //
```

```
37 // Mg         2.54            1.50         2.16
                2.32
38 // Cu         6.45            3.85         2.19
                2.30
39 // Al         4.00            2.38         2.18
                2.57
40 // Pt         1.02            0.69         2.48
                2.56
41 //
```

Scilab code Exa 4.14 Thermal conductivity of gold at 100 K and 273 K

```
1 // Scilab Code Ex4.14: Page-125 (2006)
2 clc; clear;
3 A = cell(2,2);       // Declare a 2X3 cell
4 A(1,1).entries = 1.6e+08;  // Electrcal conductivity
     of Au at 100 K, mho per metre
5 A(1,2).entries = 2.0e-08;   // Lorentz number of Au
   at 100 K, volt/K-square
6 A(2,1).entries = 5.0e+08;  // Electrcal conductivity
     of Au at 273  K, mho per metre
7 A(2,2).entries = 2.4e-08;   // Lorentz number of Au
   at 273 K, volt/K-square
8 T1 = 100;         // First temperature, K
```

```
T2 = 273;        // Second temperature, K

printf("\n----------------------------------------------------------------------");
printf("\n           T = 100 K                                    T = 273 K
                    ");
printf("\n-----------------------------------");
printf("\nElectrical conductivity)  L   Electrical conductivity)  L       ");
printf("\n    mho per metre        V/K-square              mho per metre        V/K-square");
printf("\n----------------------------------------------------------------------");
K1 = A(1,1).entries*T1*A(1,2).entries; K2 = A(2,1).entries*T2*A(2,2).entries;
 printf("\n%3.1e                    %3.1e        %3.1e                    %3.1e", A(1,1).entries, A(1,2).entries, A(2,1).entries, A(2,2).entries);
 printf("\nK = %3d W/cm-K                                K = %3d W/cm-K", K1, K2);
printf("\n----------------------------------------------------------------------");

// Result
// ----------------------------------------------------------------------

//           T = 100 K                                    T = 273 K
// -----------------------------------
-----------------------------------
```

```
26 // Electrical conductivity)  L
   Electrical conductivity)  L
27 //      mho per metre        V/K-square              mho
   per metre        V/K-square
28 //
   ------------------------------------------------------------------

29 // 1.6e+008                    2.0e-008        5.0e+008
                        2.4e-008
30 // K = 320 W/cm-K                              K = 3276
    W/cm-K
31 //
   ------------------------------------------------------------------
```

Scilab code Exa 4.15 Hall coefficient of sodium

```
1 // Scilab Code Ex4.15: Page-131 (2006)
2 clc; clear;
3 e = 1.6e-019;   // Electronic charge, C
4 a = 0.428e-09;  // Lattice constant of Na, m
5 V = a^3;    // Volume of unit cell, metre cube
6 N = 2;  // No. of atoms per unit cell of Na
7 n = N/V;    // No. of electrons per metre cube, per
   metre cube
8 R_H = -1/(n*e);  // Hall coeffcient of Na, metre
   cube per coulomb
9 printf("\nThe Hall coefficient of sodium = %4.2e
   metre cube per coulomb", R_H);
10
11 // Result
12 // The Hall coefficient of sodium = -2.45e-010 metre
    cube per coulomb
```

Scilab code Exa 4.16 Hall coefficient of beryllium

```
1 // Scilab Code Ex4.16: Page-131 (2006)
2 clc; clear;
3 e = 1.6e-019;    // Electronic charge, C
4 n = 24.2e+028;     // No. of electrons per metre cube
    , per metre cube
5 R_H = -1/(n*e);  // Hall coeffcient of Be, metre
    cube per coulomb
6 printf("\nThe Hall coefficient of beryllium = %4.2e
    metre cube per coulomb", R_H);
7
8 // Result
9 // The Hall coefficient of beryllium = -2.58e-011
    metre cube per coulomb
```

Scilab code Exa 4.17 Electronic concentration of silver from Hall coefficient

```
1 // Scilab Code Ex4.17: Page-131 (2006)
2 clc; clear;
3 e = 1.6e-019;    // Electronic charge, C
4 R_H = -8.4e-011;  // Hall coeffcient of Ag, metre
    cube per coulomb
5 n = -3*%pi/(8*R_H*e);     // Electronic concentration
    of Ag, per metre cube
6 printf("\nThe electronic concentration of Ag = %3.1e
    per metre cube", n);
7
8 // Result
9 // The electronic concentration of Ag = 8.8e+028 per
    metre cube
```

Scilab code Exa 4.18 Resistivity of a metal using Matthiessen rule

```
1 // Scilab Code Ex4.18: Page-134 (2006)
2 clc; clear;
3 // We have from Mattheissen rule, rho = rho_0 +
    alpha*T1
4 T1 = 300;    // Initial temperature, K
5 T2 = 1000;   // Final temperature, K
6 rho = 1e-06;     // Resistivity of the metal, ohm-m
7 delta_rho = 0.07*rho;       // Increase in
    resistivity of metal, ohm-m
8 alpha = delta_rho/(T2-T1);   // A constant, ohm-m/K
9 rho_0 = rho - alpha*T1;      // Resistivity at room
    temperature, ohm-m
10 printf("\nThe resistivity at room temperature = %4.2
    e ohm-m", rho);
11
12 // Result
13 // The resistivity at room temperature = 1.00e-006
    ohm-m
```

Scilab code Exa 4.19 Resitivity of Ge at 20 degree celsius

```
1 // Scilab Code Ex4.19: Page-134 (2006)
2 clc; clear;
3 // We have from Mattheissen rule, rho = rho_0 +
    alpha*T1
4 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;   // Boltzmann constant, J/mol/K
6 rho_40 = 0.2;    // Resistivity of Ge at 40 degree
    celsius, ohm-m
7 E_g = 0.7;  // Bandgap for Ge, eV
```

```
8 T1 = 20+273;     // Second temperature, K
9 T2 = 40 + 273;   // First temperature, K
10 rho_20 = rho_40*exp(E_g*e/(2*k)*(1/T1-1/T2));     //
    Resistivity of Ge at 20 degree celsius, ohm-m
11 printf("\nThe resistivity of Ge at 20 degree celsius
    = %3.1f ohm-m", rho_20);
12
13 // Result
14 // The resistivity of Ge at 20 degree celsius = 0.5
   ohm-m
```

Scilab code Exa 4.20 Solid radius and Fermi level quantities for Li

```
1 // Scilab Code Ex4.20: Page-135 (2006)
2 clc; clear;
3 rs_a0_ratio = 3.25;      // Ratio of solid radius to
   the lattice parameter
4 E_F = 50.1*(rs_a0_ratio)^(-2);  // Fermi level
   energy of Li, eV
5 T_F = 58.2e+04*(rs_a0_ratio)^(-2);  // Fermi level
   temperature of Li, K
6 V_F = 4.20e+08*(rs_a0_ratio)^(-1);  // Fermi level
   velocity of electron in Li, cm/sec
7 K_F = 3.63e+08*(rs_a0_ratio)^(-1);
8 printf("\nE_F = %4.2f eV", E_F);
9 printf("\nT_F = %4.2e K", T_F);
10 printf("\nV_F = %4.2e cm/sec", V_F);
11 printf("\nK_F = %4.2e per cm", K_F);
12
13 // Result
14 // E_F = 4.74 eV
15 // T_F = 5.51e+004 K
16 // V_F = 1.29e+008 cm/sec
17 // K_F = 1.12e+008 per cm
```

Scilab code Exa 4.21 Fermi energy for yittrium

```
1 // Scilab Code Ex4.21: Page-135 (2006)
2 clc; clear;
3 n = 6.04e+022;  // Concentration of electrons in
    yittrium, per metre cube
4 r_s = (3/(4*%pi*n))^(1/3)/1e-08;    // Radius of the
    solid, angstrom
5 a0 = 0.529; // Lattice parameter of yittrium,
    angstrom
6 rs_a0_ratio = r_s/a0;   // Solid radius to lattice
    parameter ratio
7 E_F = 50.1*(rs_a0_ratio)^(-2);  // Fermi level
    energy of Y, eV
8 printf("\nThe Fermi energy of yittrium = %5.3f eV",
    E_F);
9 Ryd = 13.6;     // Rydberg energy constant, eV
10 E_bs = 0.396*Ryd;   // Band structure energy value
    of Y, eV
11 printf("\nThe band structure value of E_F = %5.3f eV
    is in close agreement with the calculated value
    of %5.3f eV", E_bs, E_F);
12
13 // Result
14 // The Fermi energy of yittrium = 5.608 eV
15 // The band structure value of E_F = 5.386 eV is in
    close agreement with the calculated value of
    5.608 eV
```

Scilab code Exa 4.22 Plasmon energy of Al using free electron gas parameter

```
1 // Scilab Code Ex4.22: Page-137 (2006)
```

```
clc; clear;
rs_a0_ratio = 2.07;    // Solid radius to lattice
   parameter ratio for Al
E_F = 50.1*(rs_a0_ratio)^(-2);  // Fermi level
   energy of Y, eV
// According to Jellium model, h_cross*omega_P = E =
    47.1 eV *(rs_a0_ratio)^(-3/2)
E = 47.1*(rs_a0_ratio)^(-3/2);       // Plasmon
   energy of Al, eV
printf("\nThe plasmon energy of Al = %4.2f eV", E);
printf("\nThe experimental value is 15 eV");

// Result
// The plasmon energy of Al = 15.81 eV
// The experimental value is 15 eV
```

Scilab code Exa 4.23.1 Occupation probability of an electron at a given temperatur

```
// Scilab Code Ex4.1a: Page-137 (2006)
clc; clear;
E_F = 1;     // For simplicity assume Fermi energy to
    be unity, eV
k = 1.38e-023;       // Boltzmann constant, J/mol/K
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
dE = 0.1;         // Exces energy above Fermi level,
   eV
T = 300;         // Room temperature, K
E = E_F + dE;     // Energy of the level above Fermi
   level, eV
f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
   probability of the electron at 0.1 eV above E_F
printf("\nAt 300 K:");
printf("\n==========");
printf("\nThe occupation probability of electron at
   %3.1f eV above Fermi energy = %7.5f", dE, f_E);
```

```
13 E = E_F - dE;     // Energy of the level below Fermi
      level , eV
14 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
      probability of the electron at 0.1 eV below E_F
15 printf("\nThe occupation probability of electron at
      %3.1f eV below Fermi energy = %7.5f", dE, f_E);
16
17 T = 1000;          // New temperature , K
18 printf("\n\nAt 1000 K:");
19 printf("\n==========");
20 E = E_F + dE;     // Energy of the level above Fermi
      level , eV
21 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
      probability of the electron at 0.1 eV above E_F
22 printf("\nThe occupation probability of electron at
      %3.1f eV above Fermi energy = %4.2f", dE, f_E);
23 E = E_F - dE;     // Energy of the level below Fermi
      level , eV
24 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
      probability of the electron at 0.1 eV below E_F
25 printf("\nThe occupation probability of electron at
      %3.1f eV below Fermi energy = %4.2f", dE, f_E);
26
27 // Result
28 // At 300 K:
29 // ==========
30 // The occupation probability of electron at 0.1 eV
      above Fermi energy = 0.02054
31 // The occupation probability of electron at 0.1 eV
      below Fermi energy = 0.97946
32
33 // At 1000 K:
34 // ==========
35 // The occupation probability of electron at 0.1 eV
      above Fermi energy = 0.24
36 // The occupation probability of electron at 0.1 eV
      below Fermi energy = 0.76
```

Scilab code Exa 4.23.2 Variation of occupation probability with temperature

```
1 // Scilab Code Ex4.2a: Page-138 (2006)
2 clc; clear;
3 f_E = 0.01;      // Occupation probability of
     electron
4 E_F = 1;     // For simplicity assume Fermi energy to
      be unity, eV
5 k = 1.38e-023;       // Boltzmann constant, J/mol/K
6 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
7 dE = 0.5;         // Exces energy above Fermi level,
     eV
8 E = E_F + dE;     // Energy of the level above Fermi
     level, eV
9 // We have, f_E = 1/(exp((E-E_F)*e/(k*T))+1),
     solving for T
10 T = (E-E_F)*e/k*1/log(1/f_E-1);    // Temperature at
     which the electron will have energy 0.1 eV above
     the Fermi energy, K
11 printf("\nThe temperature at which the electron will
      have energy %3.1f eV above the Fermi energy =
     %4d K", dE, T);
12
13 // Result
14 // The temperature at which the electron will have
     energy 0.5 eV above the Fermi energy = 1261 K
```

Scilab code Exa 4.23.3 Average energy and speed of free electron in metal

```
1 // Scilab Code Ex4.3a: Page-139 (2006)
2 clc; clear;
3 E_F = 10;    // Fermi energy of electron in metal, eV
```

```
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
m = 9.1e-031;    // Mass of an electron, kg
E_av = 3/5*E_F;      // Average energy of free
   electron in metal at 0 K, eV
V_F = sqrt(2*E_av*e/m);    // Speed of free electron
   in metal at 0 K, eV
printf("\nThe average energy of free electron in
   metal at 0 K = %1d eV", E_av);
printf("\nThe speed of free electron in metal at 0 K
    = %4.2e m/s", V_F);

// Result
// The average energy of free electron in metal at 0
    K = 6 eV
// The speed of free electron in metal at 0 K = 1.45
   e+006 m/s
```

Scilab code Exa 4.23.4 Temperature dependence of occupation probability

```
// Scilab Code Ex4.4a: Page-139 (2006)
clc; clear;
f_E = 0.1;      // Occupation probability of electron
E_F = 5.5;     // Fermi energy of Cu, eV
k = 1.38e-023;       // Boltzmann constant, J/mol/K
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
dE = 0.05*E_F;         // Exces energy above Fermi
   level, eV
E = E_F + dE;     // Energy of the level above Fermi
   level, eV
// We have, f_E = 1/(exp((E-E_F)*e/(k*T))+1),
   solving for T
T = (E-E_F)*e/k*1/log(1/f_E-1);    // Temperature at
   which the electron will have energy 0.1 eV above
   the Fermi energy, K
printf("\nThe temperature at which the electron will
```

```
    have energy %1d percent above the Fermi energy
    %4d K", dE/E_F*100, T);
12
13
14 // Result
15 // The temperature at which the electron will have
    energy 5 percent above the Fermi energy 1451 K (
    The answer given in the textbook is wrong)
```

Scilab code Exa 4.23.5 Fermi velocity of Potassium

```
1 // Scilab Code Ex4.5a: Page-139 (2006)
2 clc; clear;
3 T_F = 24600;    // Fermi temperature of potassium, K
4 k = 1.38e-023;  // Boltzmann constant, J/mol/K
5 m = 9.1e-031;   // Mass of an electron, kg
6 E_F = k*T_F;    // Fermi energy of potassium, eV
7 v_F = sqrt(2*k*T_F/m);      // Fermi velocity of
    potassium, m/s
8 printf("\nThe Fermi velocity of potassium = %5.3e m/
    s", v_F);
9
10 // Result
11 // The Fermi velocity of potassium = 8.638e+005 m/s
```

Scilab code Exa 4.23.6 Energy level of Cu for given occupation probability

```
1 // Scilab Code Ex4.6a: Page-139 (2006)
2 clc; clear;
3 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
4 E_F = 7.0;  // Fermi energy of Cu, eV
5 f_E = 0.9;  // Occupation probability of Cu
6 k = 1.38e-023;  // Boltzmann constant, J/mol/K
```

```
7 T = 1000;    // Given temperature, K
8 // We have, f_E = 1/(exp((E-E_F)*e/(k*T))+1),
     solving for E
9 E = k*T*log(1/f_E-1) + E_F*e;      // Energy level of
     Cu for 10% occupation probability at 1000 K, J
10 printf("\nThe energy level of Cu for 10 percent
     occupation probability at 1000 K = %4.2f eV", E/e
     );
11
12 // Result
13 // The energy level of Cu for 10 percent occupation
     probability at 1000 K = 6.81 eV
```

Scilab code Exa 4.23.7 Electronic concentration in cesium

```
1 // Scilab Code Ex4.7a: Page-140 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of an electron, kg
4 e = 1.6e-019;    // Electronic charge, C
5 h = 6.626e-034; // Planck's constant, Js
6 E_F = 1.55;  // Fermi energy of Cu, eV
7 n = %pi/3*(8*m/h^2)^(3/2)*(E_F*e)^(3/2);     //
     Electronic concentration in cesium, electrons/cc
8 printf("\nThe electronic concentration in cesium =
     %5.3e electrons/cc", n);
9
10 // Result
11 // The electronic concentration in cesium = 8.733e
     +027 electrons/cc
```

Scilab code Exa 4.23.8 Fermi temperature corresponding to Fermi energy

```
1 // Scilab Code Ex4.8a: Page-141 (2006)
```

```
clc; clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
E_F = 7;  // Fermi energy, eV
k = 1.38e-023;  // Boltzmann constant, J/mol/K
T_F = E_F*e/k;     // Fermi temperature, K
printf("\nThe Fermi temperature corresponding to
   Fermi energy = %5.3e K", T_F);

// Result
// The Fermi temperature corresponding to Fermi
   energy = 8.116e+004 K
```

Scilab code Exa 4.23.9 Density of states for the electron in a cubical box

```
// Scilab Code Ex4.9a: Page-141 (2006)
clc; clear;
m = 9.1e-031;    // Mass of the electron, kg
h = 6.626e-034; // Planck's constant, Js
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
h_cross = h/(2*%pi);     // Reduced Planck's constant
   , Js
s = 0.01;    // Side of the box, m
E = 2;  // Energy range of the electron in the box,
   eV
V = s^3;     // Volume of the box, metre cube
I = integrate("E^(1/2)", 'E', 0, 2);     // Definite
   integral over E
D_E = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*I*e^(3/2);
   // Density of states for the electron in a
   cubical box, states
printf("\nThe density of states for the electron in
   a cubical box = %5.3e states", D_E);

// Result
// The density of states for the electron in a
```

```
cubical box = 1.280e+022 states
```

Scilab code Exa 4.23.10 Occupation probability of an electron above and below Ferm

```
1 // Scilab Code Ex4.10a: Page-141 (2006)
2 clc; clear;
3 E_F = 1;     // For simplicity assume Fermi energy to
      be unity, eV
4 k = 1.38e-023;        // Boltzmann constant, J/mol/K
5 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
6 dE = 0.5;          // Exces energy above Fermi level,
     eV
7 T = 300;          // Room temperature, K
8 E = E_F + dE;      // Energy of the level above Fermi
     level, eV
9 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
     probability of the electron at 0.1 eV above E_F
10 printf("\nAt 300 K:");
11 printf("\n==========");
12 printf("\nThe occupation probability of electron at
     %3.1f eV above Fermi energy = %11.9f", dE, f_E);
13 E = E_F - dE;     // Energy of the level below Fermi
     level, eV
14 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
     probability of the electron at 0.1 eV below E_F
15 printf("\nThe occupation probability of electron at
     %3.1f eV below Fermi energy = %11.9f", dE, f_E);
16
17 // Result
18 // At 300 K:
19 // ==========
20 // The occupation probability of electron at 0.5 eV
     above Fermi energy = 0.000000004
21 // The occupation probability of electron at 0.5 eV
     below Fermi energy = 0.999999996
```

Scilab code Exa 4.23.11 Occupation probability at two different temperatures

```
1 // Scilab Code Ex4.9a: Page-141 (2006)
2 clc; clear;
3 E_F = 1;     // For simplicity assume Fermi energy to
      be unity, eV
4 k = 1.38e-023;       // Boltzmann constant, J/mol/K
5 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
6 dE = 0.2;          // Exces energy above Fermi level,
     eV
7 T = 0+273;          // Room temperature, K
8 E = E_F + dE;     // Energy of the level above Fermi
     level, eV
9 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
     probability of the electron at 0.1 eV above E_F
10 printf("\nAt 273 K:");
11 printf("\n==========");
12 printf("\nThe occupation probability of electron at
     %3.1f eV above Fermi energy = %4.2e", dE, f_E);
13 T = 100+273;     // Given temperature of 100 degree
     celsius, K
14 f_E = 1/(exp((E-E_F)*e/(k*T))+1);     // Occupation
     probability of the electron at 0.1 eV below E_F
15 printf("\n\nAt 373 K:");
16 printf("\n==========");
17 printf("\nThe occupation probability of electron at
     %3.1f eV above Fermi energy = %4.2e", dE, f_E);
18
19 // Result
20 // At 273 K:
21 // ==========
22 // The occupation probability of electron at 0.2 eV
     above Fermi energy = 2.05e-004
23
```

```
24 // At 373 K:
25 // ==========
26 // The occupation probability of electron at 0.2 eV
   above Fermi energy = 1.99e-003
```

Scilab code Exa 4.23.12 Concentration of free electrons and electrical conductivit

```
1  // Scilab Code Ex4.12a: Page-142 (2006)
2  clc; clear;
3  m = 9.1e-031;    // Mass of the electron, kg
4  e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
5  r = 1.28e-010;    // Atomic radius of Cu, m
6  a = 4*r/sqrt(2);     // Lattice constant of Cu, m
7  tau = 2.7e-14;       // Relaxation time for the
   electron in Cu, s
8  V = a^3;     // Volume of the cell, metre cube
9  n = 4/V;     // Concentration of free electrons in
   monovalent copper,
10 sigma = n*e^2*tau/m;     // Electrical conductivity
   of monovalent copper, mho per m
11 printf("\nThe electrical conductivity of monovalent
   copper = %5.3e mho per cm", sigma/100);
12
13 // Result
14 // The electrical conductivity of monovalent copper
   = 6.403e+005 mho per cm
```

Scilab code Exa 4.23.13 Interelectronic energy separation between bands of Al

```
1  // Scilab Code Ex4.13a: Page-142 (2006)
2  clc; clear;
3  n = 18.1e+022;  // Number of electrons per unit
   volume, per cm cube
```

```
4 N = n/2;     // Pauli's principle for number of
     energy levels, per cm cube
5 E_F = 11.58;      // Fermi energy of Al, eV
6 E = E_F/N;      // Interelectronic energy separation
     between bands of Al, eV
7 printf("\nThe interelectronic energy separation
     between bands of Al = %4.2e eV", E);
8
9 // Result
10 // The interelectronic energy separation between
     bands of Al = 1.28e-022 eV
```

Scilab code Exa 4.23.14 Density of states in Cu contained in cubic metal

```
1 // Scilab Code Ex4.14a: Page-142 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of the electron, kg
4 h = 6.626e-034; // Planck's constant, Js
5 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
6 h_cross = h/(2*%pi);     // Reduced Planck's constant
     , Js
7 E_F = 7;     // Fermi energy of Cu, eV
8 V = 1e-06;  // Volume of the cubic metal, metre cube
9 D_EF = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*(E_F)^(1/2)
     *e^(3/2);    // Density of states in Cu contained
     in cubic metal, states/eV
10 printf("\nThe density of states in Cu contained in
     cubic metal = %3.1e states/eV", D_EF);
11
12 // Result
13 // The density of states in Cu contained in cubic
     metal = 1.8e+022 states/eV
```

Scilab code Exa 4.23.15 Electronic energy level spacing between successive levels

```
// Scilab Code Ex4.15a: Page-143 (2006)
clc; clear;
m = 9.1e-031;    // Mass of the electron, kg
h = 6.626e-034; // Planck's constant, Js
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
h_cross = h/(2*%pi);     // Reduced Planck's constant, Js
E_F = 7;     // Fermi energy of Cu, eV
V = 1e-06;  // Volume of the cubic metal, metre cube
D_EF = V/(2*%pi^2)*(2*m/h_cross^2)^(3/2)*(E_F)^(1/2)*e^(3/2);   // Density of states in Cu contained in cubic metal, states/eV
d = 1/(D_EF);      // Electronic energy level spacing between successive levels of Cu, eV
printf("\nThe electronic energy level spacing between successive levels of Cu = %4.2e eV", d);

// Result
// The electronic energy level spacing between successive levels of Cu = 5.57e-023 eV
```

Scilab code Exa 4.23.16 Energy band gaps of the solids

```
// Scilab Code Ex4.16a: Page-143 (2006)
clc; clear;
A = cell(4,2);  // Declare a 3X2 cell
A(1,1).entries = 'Li';       //
A(1,2).entries = -0.4039;    // Energy of outermost atomic orbital of Li, Rydberg unit
A(2,1).entries = 'Na';       //
A(2,2).entries = -0.3777;    // Energy of outermost atomic orbital of Na, Rydberg unit
A(3,1).entries = 'F';        //
```

```
A(3,2).entries = -1.2502;    // Energy of outermost
    atomic orbital of F, Rydberg unit
A(4,1).entries = 'Cl';       //
A(4,2).entries = -0.9067;    // Energy of outermost
    atomic orbital of Cl, Rydberg unit
cf = 13.6;  // Conversion factor for Rydberg to eV
printf("\n-----------------------------------------")
    ;
printf("\nAtom                                  Energy gap")
    ;
printf("\n%s%s                                  %5.2f eV", A
    (2,1).entries, A(4,1).entries, (A(2,2).entries-A
    (4,2).entries)*cf);
printf("\n%s%s                                   %5.2f eV",
     A(2,1).entries, A(3,1).entries, (A(2,2).entries-
    A(3,2).entries)*cf);
printf("\n%s%s                                   %5.2f eV",
     A(1,1).entries, A(3,1).entries, (A(1,2).entries-
    A(3,2).entries)*cf);
printf("\n-----------------------------------------")
    ;

// Result
// ----------------------------------------
// Atom                                Energy gap
// NaCl                                 7.19 eV
// NaF                                  11.87 eV
// LiF                                  11.51 eV
// ----------------------------------------
```

Scilab code Exa 4.23.18 Solid radius and Fermi level quantities for Cu and Nb

```
// Scilab Code Ex4.18a: Page-144 (2006)
clc; clear;
// For Cu
```

```
4  rs_a0_ratio = 2.67;      // Ratio of solid radius to
     the lattice parameter
5  E_F = 50.1*(rs_a0_ratio)^(-2);  // Fermi level
     energy of Cu, eV
6  T_F = 58.2e+04*(rs_a0_ratio)^(-2);  // Fermi level
     temperature of Cu, K
7  V_F = 4.20e+08*(rs_a0_ratio)^(-1);  // Fermi level
     velocity of electron in Cu, cm/sec
8  K_F = 3.63e+08*(rs_a0_ratio)^(-1);
9  printf("\nFor Cu :");
10 printf("\n=========");
11 printf("\nE_F = %6.4f eV", E_F);
12 printf("\nT_F = %5.3e K", T_F);
13 printf("\nV_F = %7.5e cm/sec", V_F);
14 printf("\nK_F = %6.4e per cm", K_F);
15 rs_a0_ratio = 3.07;      // Ratio of solid radius to
     the lattice parameter
16 E_F = 50.1*(rs_a0_ratio)^(-2);  // Fermi level
     energy of Nb, eV
17 T_F = 58.2e+04*(rs_a0_ratio)^(-2);  // Fermi level
     temperature of Nb, K
18 V_F = 4.20e+08*(rs_a0_ratio)^(-1);  // Fermi level
     velocity of electron in Nb, cm/sec
19 K_F = 3.63e+08*(rs_a0_ratio)^(-1);
20 printf("\n\nFor Nb:");
21 printf("\n=========");
22 printf("\nE_F = %6.4f eV", E_F);
23 printf("\nT_F = %5.3e K", T_F);
24 printf("\nV_F = %6.4e cm/sec", V_F);
25 printf("\nK_F = %6.4e per cm", K_F);
26
27 // Result
28 // For Cu :
29 // =========
30 // E_F = 7.0277 eV
31 // T_F = 8.164e+004 K
32 // V_F = 1.57303e+008 cm/sec
33 // K_F = 1.3596e+008 per cm
```

```
34 //
35 // For Nb:
36 // ========
37 // E_F = 5.3157 eV
38 // T_F = 6.175e+004 K
39 // V_F = 1.3681e+008 cm/sec
40 // K_F = 1.1824e+008 per cm
```

Chapter 5

Band Theory of Solids

Scilab code Exa 5.1 Fermi energy of Na and K

```
1 // Scilab Code Ex5.1: Page-176 (2006)
2 clc; clear;
3 h = 6.626e-34;  // Planck's constant, Js
4 h_bar = h/(2*%pi);  // Reduced Planck's constant, Js
5 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
6 m = 9.1e-031;   // Mass of an electron, kg
7
8
9 // For Na
10 n_Na = 2.65e+28;    // electronic concentration of
     Na, per metre cube
11 k_F = (3*%pi^2*n_Na)^(1/3);    // Fermi wave vector,
     per cm
12 E_F = h_bar^2*k_F^2/(2*m*e);  // Fermi energy of Na,
     eV
13 printf("\nThe fermi energy of Na = %4.2f eV", E_F);
14 printf("\nThe band structure value of Na = %4.2f eV"
     , 0.263*13.6);
15 // For K
16 n_K = 1.4e+28;  // electronic concentration of K,
     per metre cube
```

```
17 k_F = (3*%pi^2*n_K)^(1/3);     // Fermi wave vector,
     per cm
18 E_F = h_bar^2*k_F^2/(2*m*e);  // Fermi energy of K,
     eV
19 printf("\nThe fermi energy of K = %4.2f eV", E_F);
20 printf("\nThe band structure value of K = %4.2f eV",
     0.164*13.6);
21 printf("\nThe agreement between the free electron
     and band theoretical values are fairly good both
     for Na and K");
22
23
24 // Result
25 // The fermi energy of Na = 3.25 eV
26 // The band structure value of Na = 3.58 eV
27 // The fermi energy of K = 2.12 eV
28 // The band structure value of K = 2.23 eV
29 // The agreement between the free electron and band
     theoretical values are fairly good both for Na
     and K
```

Scilab code Exa 5.3 Fermi momentum of Na

```
1 // Scilab Code Ex5.3: Page-177 (2006)
2 clc; clear;
3 n_Na = 2.65e+22;     // electronic concentration of
     Na, per cm cube
4 k_F = (3*%pi^2*n_Na)^(1/3);     // Fermi wave vector,
     per cm
5 printf("\nThe fermi momentum of Na = %4.2e per cm",
     k_F);
6
7 // Result
8 // The fermi momentum of Na = 9.22e+07 per cm
```

Scilab code Exa 5.5 Energy separation between adjacent energy levels

```
1 // Scilab Code Ex5.5: Page-177 (2006)
2 clc; clear;
3 h = 6.626e-34;  // Planck's constant, Js
4 h_bar = h/(2*%pi);  // Reduced Planck's constant, Js
5 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
6 m = 9.1e-031;   // Mass of an electron, kg
7 V = 1.0e-06; // Volume of unit cube of material,
    metre cube
8
9 // For Mg
10 E_F = 7.13*e;  // Fermi energy of Mg, J
11 s = 2*%pi^2/(e*V)*(h_bar^2/(2*m))^(3/2)*(E_F)^(-1/2)
    ;  // Energy separation between levels for Mg, eV
12 printf("\nThe energy separation between adjacent
    levels for Mg = %5.3e eV", s);
13
14 // For Cs
15 E_F = 1.58*e;  // Fermi energy of Cs, J
16 s = 2*%pi^2/(e*V)*(h_bar^2/(2*m))^(3/2)*(E_F)^(-1/2)
    ;  // Energy separation between levels for Cs, eV
17 printf("\nThe energy separation between adjacent
    levels for Cs = %5.3e eV", s);
18
19
20 // Result
21 // The energy separation between adjacent levels for
    Mg = 5.517e-23 eV
22 // The energy separation between adjacent levels for
    Cs = 1.172e-22 eV
```

Scilab code Exa 5.9 Coupling constant of superconducting lead

```
1 // Scilab Code Ex5.9: Page-180 (2006)
2 clc; clear;
3
4 gamma_expt = 7.0e-04;    // Experimental value of
     electronic specific heat, cal/mol/K-square
5 gamma_theory = 3.6e-04;    // Theoretical value of
     electronic specific heat, cal/mol/K-square
6 L = poly(0, 'L');
7 L = roots(gamma_expt - gamma_theory*(1 + L));
8 printf("\nThe electron-phonon coupling constant of
     superconductor = %3.1f", L);
9
10 // Result
11 // The electron-phonon coupling constant of
     superconductor = 0.9
```

Scilab code Exa 5.10 Electronic specific heat coefficient of superconductor

```
1 // Scilab Code Ex5.10: Page-181 (2006)
2 clc; clear;
3 N_Ef = 1.235;    // Density of states at fermi energy
     , electrons/atom-eV
4 N = 6.023e+23;  // Avogadro's number
5 k = 1.38e-23;    // Boltzmann constant, J/mol/K
6 e = 1.6e-019;    // Charge on an electron, C
7 gama = %pi^2*k^2/3*(N_Ef*N/e);    // Electronic
     specific heat coefficient, J/g-atom-kelvin square
8
9 printf("\nThe electronic specific heat coefficient
     of superconductor = %5.3f mJ/g-atom-kelvin square
     ", gama/1e-03);
10
11 // Result
```

```
12 // The electronic specific heat coefficient of
   superconductor = 2.913 mJ/g-atom-kelvin square
```

Scilab code Exa 5.11 Electron phonon coupling constant for metal

```
1 // Scilab Code Ex5.11: Page-181 (2006)
2 clc; clear;
3 gamma_expt = 4.84;    // Experimental value of
   electronic specific heat of metal, mJ/g-atom/K-
   square
4 gamma_theory = 2.991;    // Theoretical value of
   electronic specific heat of metal, mJ/g-atom/K-
   square
5 L = poly(0, 'L');
6 L = roots(gamma_expt - gamma_theory*(1 + L));
7 printf("\nThe electron-phonon coupling constant for
   metal = %5.3f", L);
8
9 // Result
10 // The electron-phonon coupling constant for metal =
    0.618
```

Scilab code Exa 5.12 Pauli spin susceptibility of Mg

```
1 // Scilab Code Ex5.12: Page-181 (2006)
2 clc; clear;
3 mu_B = 9.24e-027; // Bohr's magneton, J/T
4 N_Ef = 0.826;    // Density of states at fermi energy
   , electrons/atom-eV
5 N = 6.023e+23;   // Avogadro's number
6 e = 1.6e-019;    // Energy equivalent of 1 eV, J
7 chi_Pauli = mu_B^2*N_Ef*N/e;
```

```
8  printf("\nPauli spin susceptibility of Mg = %5.2e
       cgs units", chi_Pauli/1e-03);
9
10 // Result
11 // Pauli spin susceptibility of Mg = 2.65e-07 cgs
       units
```

Chapter 6

Semiconductor Physics

Scilab code Exa 6.1 Crystal absorption wavelengths from energy gaps

```
1 // Scilab Code Ex6.1: Page-190 (2006)
2 clc; clear;
3 S = cell(4,2);       // Declare a 4X2 cell
4 // Enter material names
5 S(1,1).entries = 'Si'; S(2,1).entries = 'GaAs'; S
    (3,1).entries = 'GaP'; S(4,1).entries = 'ZnS';
6 // Enter energy band gap values
7 S(1,2).entries = 1.11; S(2,2).entries = 1.42; S(3,2)
    .entries = 2.26; S(4,2).entries = 3.60;
8 h = 6.626e-034;      // Planck's constant, Js
9 c = 3e+08;       // Speed of light, m/s
10 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
11 printf("\
    n----------------------------------------------------
    ");
12 printf("\nMaterial       E_g (eV)     Critical
    Wavelength (micron)");
13 printf("\
    n----------------------------------------------------
    ");
14 for i = 1:1:4
```

```
15      lambda = h*c/(S(i,2).entries*e);
16      printf("\n%8s          %4.2f              %5.3f", S(i
           , 1).entries, S(i, 2).entries, lambda/1e-06);
17  end
18  printf("\
       n-------------------------------------------------------
       ");
19
20  // Result
21  //
       ---------------------------------------------------------
22  // Material       E_g (eV)     Critical Wavelength (
       micron)
23  //
       ---------------------------------------------------------
24  //       Si          1.11              1.119
25  //     GaAs          1.42              0.875
26  //      GaP          2.26              0.550
27  //      ZnS          3.60              0.345
28  //
       ---------------------------------------------------------
```

Scilab code Exa 6.2 Phonon energy to lift the electron from valence band to conduc

```
1  // Scilab Code Ex6.2: Page-192 (2006)
2  clc; clear;
3  c = 3e+08;        // Speed of light, m/s
4  h = 6.626e-034;       // Planck's constant, Js
5  e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
6  omega = 2e+014;        // Wave vector involved in
       phonon energy, rad per sec
7  f = omega/(2*%pi);    // Frequency of the wave, Hz
```

```
E = h*f/e;          // Phonon energy involved in Si to
    lift the electron, eV
printf("\nThe phonon energy involved in Si = %5.3f
   eV which is insufficient to lift an electron.", E
   );

// Result
// The phonon energy involved in Si = 0.132 eV which
    is insufficient to lift an electron.
```

Scilab code Exa 6.3 Densitites of Si and GaAs

```
// Scilab Code Ex6.3: Page-192 (2006)
clc; clear;
N_A = 6.023e+023;    // Avogadro's number
// For Si
A = 28.1;    // Atomic weight of Si, g/mol
a = 5.43e-08;    // Lattice constant for Si, cm
n = 8/a^3;  // Number of atoms per unit volume,
   atoms/cc
rho = n*A/N_A;  // Density of Si, g/cc
printf("\nThe density of Si = %4.2f atoms per cc",
   rho);
// For GaAs
A = 69.7+74.9;    // Atomic weight of GaAs, g/mol
a = 5.65e-08;    // Lattice constant for Si, cm
n = 4/a^3;  // Number of atoms per unit volume,
   atoms/cc
rho = n*A/N_A;  // Density of GaAs, g/cc
printf("\nThe density of GaAs = %5.3f atoms per cc",
    rho);

// Result
// The density of Si = 2.33 atoms per cc
// The density of GaAs = 5.324 atoms per cc
```

Scilab code Exa 6.4 Intrinsic carrier concentration of GaAs at 300 K

```
1 // Scilab Code Ex6.4: Page-196 (2006)
2 clc; clear;
3 m = 9.11e-031;  // Electron Rest Mass , kg
4 k = 1.38e-023;  // Boltzmann constant, J/mol/K
5 h = 6.626e-034; // Planck's constant, Js
6 T = 300;        // Room temperature, K
7 m_e = 0.068*m;    // Mass of electron, kg
8 m_h = 0.56*m;   // Mass of hole, kg
9 E_g = 1.42*1.6e-019;     // Energy band gap for GaAs
    , J
10 n_i = 2*(2*%pi*k*T/h^2)^(3/2)*(m_e*m_h)^(3/4)*exp(-
    E_g/(2*k*T));
11 printf("\nThe Intrinsic carrier concentration of
    GaAs at 300 K = %1.0e per metre cube", n_i);
12
13 // Result
14 // The intrinsic carrier concentration of GaAs at
    300 K = 3e+012 per metre cube
```

Scilab code Exa 6.5 Position of Fermi level of Si at room temperature

```
1 // Scilab Code Ex6.5: Page-197 (2006)
2 clc; clear;
3 m = 9.11e-031;  // Electron Rest Mass , kg
4 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
5 k = 1.38e-023;  // Boltzmann constant, J/mol/K
6 T = 300;        // Room temperature, K
7 m_e = 1.1*m;    // Mass of electron, kg
8 m_h = 0.56*m;   // Mass of hole, kg
```

```
E_g = 1.1;      // Energy band gap for GaAs, J
E_F = E_g/2+3/4*k*T/e*log(m_h/m_e);    // Position of Fermi level of Si at room temperature, eV
printf("\nThe position of Fermi level of Si at room temperature = %5.3f eV", E_F);
printf("\nThe fermi level in this case is shifted downward from the midpoint (0.55 eV) in the forbiddem gap.");

// Result
// The position of Fermi level of Si at room temperature = 0.537 eV
// The fermi level in this case is shifted downward from the midpoint (0.55 eV) in the forbiddem gap.
```

Scilab code Exa 6.6 Intrinsic resistivity of Ge at room temperature

```
// Scilab Code Ex6.6: Page-197 (2006)
clc; clear;
e = 1.6e-019;    // Electronic charge, C
n_i = 2.15e+013;     // Carrier density of Ge at room temperature, per cc
mu_e = 3900;     // Mobility of electron, cm-square/V-s
mu_h = 1900;     // Mobility of hole, cm-square/V-s
sigma_i = e*(mu_e+mu_h)*n_i;     // Intrinsic conductivity of Ge, mho per m
rho_i = 1/sigma_i;  // Intrinsic resistivity of Ge at room temperature, ohm-m
printf("\nThe intrinsic resistivity of Ge at room temperature = %2d ohm-cm", rho_i);


// Result
// The intrinsic resistivity of Ge at room
```

temperature = 50 ohm-cm

Scilab code Exa 6.7 Conductivity in CdS

```
1 // Scilab Code Ex6.7: Page-197 (2006)
2 clc; clear;
3 m = 9.1e-031;    // Mass of an electron, kg
4 e = 1.6e-019;    // Electronic charge, C
5 k = 1.38e-023;   // Boltzmann constant, J/mol/K
6 T = 30;     // Given temperature, K
7 n = 1e+22;     // Carrier density of CdS, per metre
    cube
8 mu = 1e-02;      // Mobility of electron, metre-
    square/V-s
9 sigma = e*mu*n;     // Conductivity of CdS, mho per m
10 printf("\nThe conductivity of CdS sample = %2d mho
    per m", ceil(sigma));
11 m_eff = 0.1*m;  // Effective mass of the charge
    carries, kg
12 t = m_eff*sigma/(n*e^2);  // Average time between
    successive collisions, s
13 printf("\nThe average time between successive
    collisions = %4.2e sec", t);
14 // We have 1/2*m_eff*v^2 = 3/2*k*T, solving for v
15 v = sqrt(3*k*T/m_eff);     // Velocity of charrge
    carriers, m/s
16 l = v*t;     // Mean free distance travelled by the
    carrier, m
17 printf("\nThe mean free distance travelled by the
    carrier = %4.2e m", l);
18
19 // Result
20 // The conductivity of CdS sample = 16 mho per m
21 // The average time between successive collisions =
    5.69e-015 sec
```

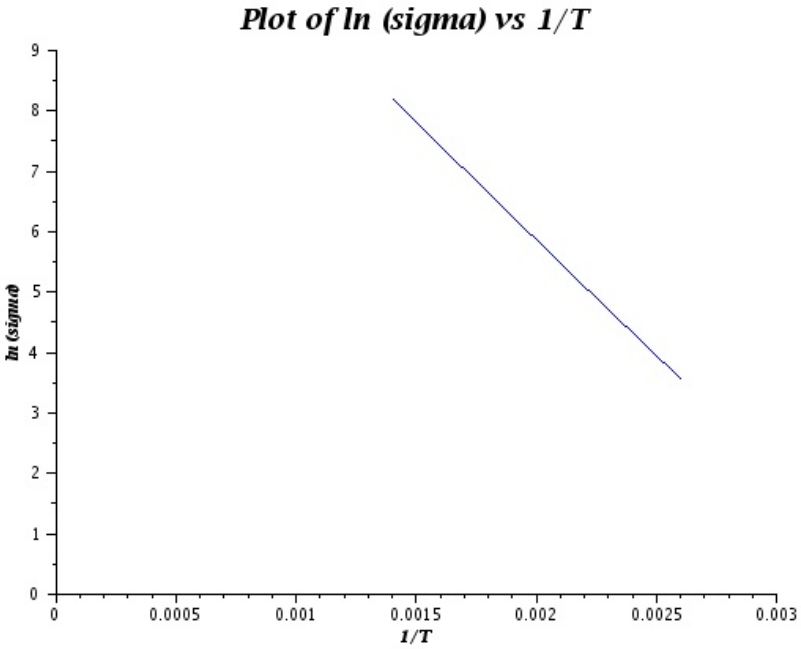

Figure 6.1: Energy Gap of Ge

```
22 // The mean free distance travelled by the carrier =
       6.64e-010 m
```

Scilab code Exa 6.8 Energy Gap of Ge

```
1 // Scilab Code Ex6.8: Page-199 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/mol/K
4 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
5 T = [385 455 556 714];      // Temperatures of Ge, K
6 rho = [0.028 0.0061 0.0013 0.000274];   //
```

```
    Electrical resistivity , ohm-m
Tinv = zeros(4);    // Create an empty row matrix
    for 1/T
ln_sigma = zeros(4);  // Create the empty row matrix
    for log(sigma)
for i = 1:1:4
    Tinv(i) = 1/T(i);
    log_sigma(i) = log(1/rho(i));
end
// Plot the graph
plot(Tinv, log_sigma);
a=gca(); // Handle on axes entity
a.box="off";
a.x_location = "origin";
a.y_location = "origin";
a.x_label
a.y_label
a.title
type(a.title);
x_label=a.x_label;
x_label.text="1/T";
x_label.font_style= 5;
y_label=a.y_label;
y_label.text="ln (sigma)";
y_label.font_style= 5;
t=a.title;
t.foreground=9;
t.font_size=4;
t.font_style=5;
t.text="Plot of ln (sigma) vs 1/T";
// Calculate slope
slope = (log_sigma(2)-log_sigma(1))/(Tinv(2)-Tinv(1)
    );
E_g = abs(2*slope*k);          // Energy gap of Ge, J
printf("\nThe energy gap of Ge = %5.3f eV", E_g/e);

// Result
// The energy gap of Ge = 0.658 eV
```

Scilab code Exa 6.9 Energy gap and emission wavelength of Al doped GaAs

```
1 // Scilab Code Ex6.9: Page-199 (2006)
2 clc; clear;
3 h = 6.626e-34;  // Planck's constant, Js
4 c = 3e+08;      // Speed of light, m/s
5 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
6 x = 0.07;   // Al concentration in host GaAs
7 E_g = 1.424 + 1.266*x + 0.266*x^2;    // Band gap of
    GaAs as a function of x, eV
8 // As E_g = h*c/lambda, solving for lambda
9 lambda = h*c/(E_g*e);   // Emission wavelength of
    light, m
10 printf("\nThe energy band gap of Al doped GaAs = %4
    .2f eV", E_g);
11 printf("\nThe emission wavelength of light = %4.2f
    micron", lambda/1e-06);
12 printf("\nThe Al atoms go as substitutional impurity
    in the host material.");
13
14 // Result
15 // The energy band gap of Al doped GaAs = 1.51 eV
16 // The emission wavelength of light = 0.82 micron
17 // The Al atoms go as substitutional impurity in the
    host material.
```

Scilab code Exa 6.10 Energy gap of Al doped GaAs

```
1 // Scilab Code Ex6.10: Page-200 (2006)
2 clc; clear;
3 x = 0.38;   // Al concentration in host GaAs
```

```
4 E_g = 1.424 + 1.266*x + 0.266*x^2;     // Band gap of
      GaAs as a function of x, eV
5 printf("\nThe energy band gap of 38 percent Al doped
      in GaAs = %5.3f eV", E_g);
6
7 // Result
8 // The energy band gap of 38 percent Al doped in
     GaAs = 1.943 eV
```

Scilab code Exa 6.11 Resistivity of Ge at 20 degree celsius

```
1 // Scilab Code Ex6.11: Page-200 (2006)
2 clc; clear;
3 k = 1.38e-023;  // Boltzmann constant, J/mol/K
4 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
5 rho_40 = 0.2;   // Resistivity of Ge at 40 degree
     celsius, ohm-m
6 T1 = 40+273;    // Temperature at which resistivity
     of Ge becomes 0.2 ohm-m, K
7 T2 = 20+273;    // Temperature at which resistivity
     of Ge is to be calculated, K
8 E_g = 0.7;  // Band gap of Ge, eV
9 // As rho = exp(E_g/(2*k*T)), so for rho_20
10 rho_20 = rho_40*exp(E_g/(2*k/e)*(1/T2-1/T1));  //
     Resistivity of Ge at 20 degree celsius, ohm-m
11 printf("\nThe resistivity of Ge at 20 degree celsius
      = %3.1f ohm-m", rho_20);
12
13 // Result
14 // The resistivity of Ge at 20 degree celsius = 0.5
     ohm-m
```

Scilab code Exa 6.12 Donor ionization energies at room temperature

```
// Scilab Code Ex6.12: Page-203 (2006)
clc; clear;
k = 1.38e-023;  // Boltzmann constant, J/mol/K
e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
T = 300;        // Room temperature of the material, K
K_Si = 11.7;    // Dielectric constant of Si
K_Ge = 15.8;    // Dielectric constant of Ge
m = 9.1e-031;   // Mass of an electron, kg
m_eff = 0.2;    // Effective masses of the electron in both Si and Ge, kg
E_ion_Si = 13.6*m_eff/K_Si^2;   // Donor ionization energy of Si, eV
E_ion_Ge = 13.6*m_eff/K_Ge^2;   // Donor ionization energy of Ge, eV
E = k*T/e;      // Energy available for electrons at 300 K, eV
printf("\nThe donor ionization energy of Si = %6.4f eV", E_ion_Si);
printf("\nThe donor ionization energy of Ge = %6.4f eV", E_ion_Ge);
printf("\nThe energy available for electrons at 300 K = %5.3f eV", E);

// Result
// The donor ionization energy of Si = 0.0199 eV
// The donor ionization energy of Ge = 0.0109 eV
// The energy available for electrons at 300 K = 0.026 eV
```

Scilab code Exa 6.13 Radius of the orbit of the fifth valence electron of the acce

```
// Scilab Code Ex6.13: Page-203 (2006)
clc; clear;
e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
```

```
epsilon = 15.8;  // Dielectric constant of Ge
m = 9.1e-031;   // Mass of an electron, kg
m_e = 0.2*m;     // Effective masses of the electron in Ge, kg
a_Ge = 5.65;    // Lattice parameter of Ge, angstrom
A_d = 0.53*epsilon*(m/m_e);    // Radius of donor atom, angstrom
printf("\nThe radius of the orbits of fifth valence electron of acceptor impurity = %2d angstrom", ceil(A_d));
printf("\nThis radius is %d times the lattice constant of Ge", ceil(A_d/a_Ge));

// Result
// The radius of the orbits of fifth valence electron = 42 angstrom
// This radius is 8 times the lattice constant of Ge
```

Scilab code Exa 6.14 Mobility of electron and hole concentration in Ge

```
// Scilab Code Ex6.14: Page-203 (2006)
clc; clear;
e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
tau = 1e-012;   // Life time of electron in Ge, s
m = 9.1e-031;   // Mass of an electron, kg
m_e = 0.5*m;     // Effective masses of the electron in Ge, kg
mu = e*tau/m_e;   // Mobility of electron in Ge, m-square/V-s
n_i = 2.5e+019;     // Intrinsic carrier concentration of Ge at room temperature, per metre cube
n_Ge = 5e+028;  // Concentration of Ge atoms, per metre cube
n_e = n_Ge/1e+06;    // Concentration of impurity
```

```
    atoms, per metre cube
11 // From law of mass action, n_e*n_h = n_i^2, solving
    for n_h
12 n_h = n_i^2/n_e;     // Concentration of holes, per
    metre cube
13
14 printf("\nThis mobility of electron in Ge = %4d cm-
    square/V-s", mu/1e-04);
15 printf("\nThis concentration of holes in Ge = %4.2e
    per metre cube", n_h);
16
17 // Result
18 // This mobility of electron in Ge = 3516 cm-square/
    V-s
19 // This concentration of holes in Ge = 1.25e+016 per
    metre cube
```

Scilab code Exa 6.15 Hole concentration in Ge at room temperature

```
1 // Scilab Code Ex6.15: Page-204 (2006)
2 clc; clear;
3 n_i = 2.5e+019;       // Intrinsic carrier
    concentration of Ge at room temperature, per
    metre cube
4 n_Ge = 5e+028;  // Concentration of Ge atoms, per
    metre cube
5 delta_d = 1e+06;      // Rate at which pentavalent
    impurity is doped in pure Ge, ppm
6 n_e = n_Ge/delta_d;     // Concentration of impurity
    atoms, per metre cube
7 // From law of mass action, n_e*n_h = n_i^2, solving
    for n_h
8 n_h = n_i^2/n_e;     // Concentration of holes, per
    metre cube
9
```

```
10 printf("\nThis concentration of holes in Ge = %4.2e
     per metre cube", n_h);
11
12 // Result
13 // This concentration of holes in Ge = 1.25e+016 per
     metre cube
```

Scilab code Exa 6.16 Hall effect in n type semiconductor

```
1  // Scilab Code Ex6.16: Page-205 (2006)
2  clc; clear;
3  e = 1.6e-019;    // Charge on an electron, C
4  mu = 1400e-04;   // Mobility of electron, metre-
     square per volt per sec
5  l = 300-06;      // Length of the n-type semiconductor
     , m
6  w = 100-06;      // Width of the n-type semiconductor,
     m
7  t = 20-06;       // Thickness of the n-type
     semiconductor, m
8  N_D = 4.5e+021; // Doping concentration of donor
     impurities, per metre-cube
9  V = 10;          // Biasing voltage for semiconductor, V
10 B_prep = 1; // Perpendicular magnetic field to which
     the semiconductor is subjected, tesla
11
12 // Part (a)
13 n = N_D;      // Electron concentration in
     semiconductor, per cc
14 R_H = -1/(n*e); // Hall Co-efficient, per C per
     metre cube
15
16 // Part (b)
17 rho = 1/(n*e*mu);          // Resistivity of
     semiconductor, ohm-m
```

```
18 R = rho*l/(w*t);     // Resistance of the
      semiconductor, ohm
19 I = V/R;     // Current through the semiconductor, A
20 V_H = R_H*I*B_prep/t;    // Hall voltage, V
21
22 // Part (c)
23 theta_H = atand(-mu*B_prep);     // Hall angle,
      degrees
24
25
26 printf("\nHall coefficient, R_H = %4.2e per C metre
      cube", R_H);
27 printf("\nHall voltage, V_H = %4.2f V", abs(V_H));
28 printf("\nHall angle, theta_H = %4.2f degree",
      theta_H);
29
30 // Result
31 // Hall coefficient, R_H = -1.39e-003 per C metre
      cube
32 // Hall voltage, V_H = 0.45 V
33 // Hall angle, theta_H = -7.97 degree
```

Chapter 8

Magnetism

Scilab code Exa 8.1 Spontaneous magnetization of iron

```
1 // Scilab code Ex8.1 Page:241 (2006)
2 clc; clear;
3 rho = 7.9e+03;     // Density of iron, kg per cubic
    meter
4 A = 56e-03;     // Atomic weight of iron, g/mol
5 N_A = 6.02e+023;     // Avogadro's number, atoms per
    mole
6 mu_B = 9.3e-024;     // Bohr magneton;     // Ampere
    meter square
7 n = rho*N_A/A;     // Total number of atoms per unit
    cell, per cubic meter
8 M = 2.2*n*mu_B;     // Spontaneous magnetization of
    iron, Ampere per meter
9 printf("\nSpontaneous magnetization of iron = %4.2e
    Ampere per meter", M);
10
11 // Result
12 // Spontaneous magnetization of iron = 1.74e+006
    Ampere per meter
```

Scilab code Exa 8.2 Saturation magnetization of a ferromagnetic material

```
1 // Scilab code Ex8.2 Page:241 (2006)
2 clc; clear;
3 n = 3e+028;     // Spin density of electrons in a
     ferromagnetic material, per cubic meter
4 mu = 3e-023;    // spin magnetic moment of a
     ferromagnetic material, Square Ampere
5 M_s = n*mu;     // Saturation magnetization of a
     ferromagnetic material, Per Ampere
6 printf("\nSaturation magnetization of a
     ferromagnetic material = %1.0e ampere per meter",
      M_s);
7
8 // Result
9 // Saturation magnetization of a ferromagnetic
     material = 9e+005 ampere per meter
```

Scilab code Exa 8.3 Magnetic susceptibility of Lithium

```
1 // Scilab code Ex8.3 Page:241 (2006)
2 clc; clear;
3 h_bar = 6.58e-016;      // Planck's constant, eV.s
4 m = 0.511e+06;          // Mass of an electron, eV
5 e = 1.6e-012;           // Energy equivalent of 1 eV,
     erg/eV
6 c = 3.0e+010;           // Speed of light, cm/s
7 N = 4.7e+022;     // Free electron gas concentration
     of Lithium, per cubic cm
8 mu_B = 9.27e-021;     // Bohr magneton, Ampere cm-
     square
```

```
9  E_F = (h_bar*c)^2/(2*m)*(3*%pi^2*N)^(2/3);     //
     Fermi energy, eV
10 chi = 3*N*mu_B^2/(2*E_F*e);     // Magnetic
     susceptibility of Lithium, cgs units
11 printf("\nMagnetic susceptibility of Lithium = %2.0e
     cgs units", chi);
12
13 // Result
14 // Magnetic susceptibility of Lithium = 8e-007 cgs
     units
```

Scilab code Exa 8.4 Diamagnetic susceptibility of helium atom in ground state

```
1  // Scilab code Ex8.4 Page:241 (2006)
2  clc; clear;
3  a_B = 0.53e-08;     // Bohr radius, cm
4  N = 27e+023;     // Atomic density of He gas, per
     cubic cm
5  c = 3e+010;     // Speed of light, cm/sec
6  e = 1.6e-019;     // Charge of an electron, Coulomb
7  m = 9.1e-028;     // Mass of an electron, g
8  // As r_classic = e^2/(m*c^2), Classical radius of
     an electron
9  r_classic = 2.8e-013;    // Classical radius of the
     electron, cm
10 chi = -2*N*r_classic/6*a_B^2;     // Magnetic
     susceptibility of Helium, cgs units
11
12 printf("\nDiamagnetic susceptibility of helium atom
     in ground state = %3.1e emu", chi);
13
14 // Result
15 // Diamagnetic susceptibility of helium atom in
     ground state = -7.1e-006 emu
```

Scilab code Exa 8.5 Atomic radii of helium and copper from atomic susceptibilities

```
1 // Scilab code Ex8.5 Page:242 (2006)
2 clc; clear;
3 chiA_He = 1.9e-06;     // Atomic susceptibility of
     helium, cm cube per mole
4 chiA_Cu = 18e-06;     // Atomic susceptibility of
     Copper, cm cube per mole
5 Q_sp = 1.77e+07;     // Specific charge of an
     electron, emu
6 Ne = 9650;     // Charge of a gram ion, emu
7 Z_He = 2;     // Atomic number of helium atom
8 Z_Cu = 29;     // Atomic number of copper atom
9 R_He = sqrt(abs(-6*chiA_He/(Ne*Z_He*Q_sp)));     //
     Magnetic susceptibility of helium atom, cgs units
10 R_Cu = sqrt(abs(-6*chiA_Cu/(Ne*Z_Cu*Q_sp)));     //
     Magnetic susceptibility of copper atom, cgs units
11 printf("\nAtomic radius of helium = %4.2e cm", R_He)
     ;
12 printf("\nAtomic radius of copper = %4.2e cm", R_Cu)
     ;
13
14 // Result
15 // Atomic radius of helium = 5.78e-009 cm
16 // Atomic radius of copper = 4.67e-009 cm
```

Scilab code Exa 8.6 Atomic susceptibility of Ne atom

```
1 // Scilab code Ex8.6 Page:242 (2006)
2 clc; clear;
3 N = 6.039e+022;     // Atomic density of Neon gas,
     per cubic cm
```

```
4 // As r_classic = e^2/(m*c^2), Classical radius of
    an electron
5 r_classic = 2.8e-013;   // Classical radius of the
    electron, cm
6 Z = 10;    // Atomic number of helium atom
7 a0 = 0.53e-08;      // Bohr's radius, cm
8 n1 = 2, n2 = 2, n3 = 6;     // Occupation numbers
    for 1s, 2s and 2p states of Ne
9 r_sq_1s = 0.031;    // Expectation value for 1s
    state
10 r_sq_2s = 0.905;    // Expectation value for 2s
    state
11 r_sq_2p = 1.126;    // Expectation value for 2p
    state
12 mean_r_sq = n1*r_sq_1s + n2*r_sq_2s + n3*r_sq_2p;
    // Mean square radius, cm-square
13 Chi_A = -1/6*N*Z*r_classic*mean_r_sq*a0^2;    //
    Magnetic susceptibility of helium atom, cgs units
14 printf("\nAtomic susceptibility of Ne atom = %6.4e
    emu/mole", Chi_A);
15
16 // Result
17 // Atomic susceptibility of Ne atom = -6.8302e-006
    emu/mole
```

Scilab code Exa 8.7 Langevin approximation for paramagnetism

```
1 // Scilab code Ex8.7: Page:249 (2006)
2 clc; clear;
3 e = 1.6e-019;   // Energy equivalent of 1 eV, J/eV
4 h = 6.626e-034; // Planck's constant, Js
5 h_cross = h/(2*%pi);    // Reduced Planck's constant
    , Js
6 m = 9.1e-031;   // Mass of an electron, kg
7 mu = e*h_cross/(2*m);    // Bohr magneton, J/T
```

```
8 mu_H = mu/e;     // Magnetic energy, eV
9 kT = 0.025;     // Energy associated with two degrees
      of freedom, eV
10 E_ratio = mu_H/kT;  // Exceptional terms in Langevin
    's function
11 printf("\nThe magnitude of mu*H/(k*T) = %3.1e",
    E_ratio);
12
13 // Result
14 // The magnitude of mu*H/(k*T) = 2.3e-003
```

Scilab code Exa 8.8 Paramagnetic susceptibility of Mg

```
1 // Scilab code Ex8.8 Page:249 (2006)
2 clc; clear;
3 mu = 5.78e-005;     // Bohr magneton, eV/T
4 NE_F = 0.826;     // Density of states at fermi level
    , electrons/atom-J
5 chi_Pauli = mu^2*NE_F/1e-004;     // Pauli
    diamagnetism, cgs units
6 chi_Core = -4.2e-06;     // Core diamagnetism, cgs
    units
7 chi_Landau = -1/3*chi_Pauli;     // Landau
    diamagnetism, cgs units
8 chi_Total = chi_Core+ chi_Pauli+chi_Landau;     //
    Paramagnetic susceptibility of Mg, cgs units
9
10 printf("\nThe paramagnetic susceptibility of Mg =
    %5.2e cgs units",chi_Total);
11
12 // Result
13 // The paramagnetic susceptibility of Mg = 1.42e-05
      cgs units
```

Scilab code Exa 8.9 Pauli spin susceptibility and diamagnetic contribution in Alum

```
// Scilab code Ex8.9 Page:250 (2006)
clc; clear;
e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
mu = 9.29e-024;     // Bohr magneton, J/T
mu_0 = 1.26e-006;     // Permeability of free space,
   Sq. tesla cubic meter per joule
E_F= 11.63*e;     // Fermi energy, J
N = 6.02e+028;     // Atomic concentration, atoms per
    cubic meter
chi_Total = 2.2e-005;    // Paramagnetic
   susceptibility of Mg, S.I. units
chi_Pauli = 3*N*mu^2*mu_0/(2*E_F);     // Pauli
   diamagnetism, S.I. units
chi_dia = chi_Total - chi_Pauli;     // Diamagnetic
   contribution to magnetic susceptibility

printf("\nThe Pauli spin susceptibility of Al = %5.3
   e S.I. units", chi_Pauli);
printf("\nThe diamagnetic contribution to magnetic
   susceptibility of Al = %5.3e S.I. units", chi_dia
   );

// Result
// The Pauli spin susceptibility of Al = 5.277e-06 S
   .I. units
// The diamagnetic contribution to magnetic
   susceptibility of Al = 1.672e-05 S.I. units
```

Scilab code Exa 8.10 Pauli spin susceptibility for Na

```
// Scilab code Ex8.10 Page:250 (2006)
clc; clear;
a0 = 5.3;     // Bohr radius, nm
rs_a0_ratio = 3.93;      // Ratio of solid radius to
   the lattice parameter
chi_Pauli = 2.59/rs_a0_ratio;     // Pauli's spin
   susceptibility, cgs units

printf("\nThe Pauli spin susceptibility for Na in
   terms of free electron gas parameter = %4.2f",
   chi_Pauli);

// Result
// The Pauli spin susceptibility for Na in terms of
   free electron gas parameter = 0.66
```

Scilab code Exa 8.11 Effective magneton number of Mn ion

```
// Scilab code Ex8.11 Page:264 (2006)
clc; clear;
S = 2;  // Spin quantum number
J = 0;  // Total quantum number
L = 2;  // Orbital quantum number
g = 2;  // Lande splitting factor
printf("\nThe spectroscopic term value of Mn3+ ion =
    %d_D_%d", 2*S+1, J);
// For J = L - S
J = L - S;
mu_N = g*sqrt(J*(J+1)); // Effective magneton number
printf("\nThe effective magneton number for J = L -
   S is %d", mu_N);
// For J = S, L = 0 so that
L = 0;
J = L+S;
mu_N = g*sqrt(J*(J+1)); // Effective magneton number
```

```
16 printf("\nThe effective magneton number for J = S is
       %3.1f.\nIt is in agreement with the experimental
       value of 5.0.", mu_N);
17
18 // Result
19 // The spectroscopic term value of Mn3+ ion = 5_D_0
20 // The effective magneton number for J = L - S is 0
21 // The effective magneton number for J = S is 4.9.
22 // It is in agreement with the experimental value of
       5.0.
```

Scilab code Exa 8.12 Magnetic moment of 3d electrons of Fe using Hunds rule

```
1 // Scilab code Ex8.12 Page:264 (2006)
2 clc; clear;
3 mu = 9.27e-024; // Bohr's magneton, J/T
4 N_up = 5;   // Number of electrons with spin up as
     per Hunds Rule
5 N_down = 1; // Number of electrons with spin down as
       per Hunds Rule
6 M = mu*(N_up-N_down);      // Net magnetic moment
     associated with six electrons in the 3d shell, J/
     T
7
8 printf("\nThe magnetic moment of 3d electrons of Fe
     using Hunds rule = %d Bohr magnetons", M/mu);
9
10 // Result
11 // The magnetic moment of 3d electrons of Fe using
     Hunds rule = 4 Bohr magnetons
```

Scilab code Exa 8.13 Magnetic moment of compounds using Hunds rule

```
// Scilab code Ex8.13 Page:264 (2006)
clc; clear;
C = cell(3,4);
// Enter compound names
C(1,1).entries = 'LaCrO3';
C(2,1).entries = 'LaMnO3';
C(3,1).entries = 'LaCoO3';
// Enter Magnetic moments from Hunds rule
C(1,2).entries = 3.0;
C(2,2).entries = 4.0;
C(3,2).entries = 5.0;
// Enter Magnetic moments from Band theory
C(1,3).entries = 2.82;
C(2,3).entries = 3.74;
C(3,3).entries = 4.16;
// Enter Magnetic moments from the Experiment
C(1,4).entries = 2.80;
C(2,4).entries = 3.90;
C(3,4).entries = 4.60;
printf("\
    n----------------------------------------------------
    ");
printf("\nCompound  Magnetic moment per formula unit
    (in BM)  ");
printf("\n
    ------------------------------------------");
printf("\n           Hunds Rule      Band Theory
    Experiment");
printf("\
    n----------------------------------------------------
    ");
for i = 1:1:3
    printf("\n%s       %3.2f                %4.2f
                %4.2f", C(i,1).entries, C(i,2).
        entries, C(i,3).entries, C(i,4).entries);
end
printf("\
    n----------------------------------------------------
```

```
    ");
29
30 // Result
31 //
    ---------------------------------------------------
32 // Compound  Magnetic moment per formula unit (in BM
    )
33 //
    ------------------------------------------
34 //            Hunds Rule      Band Theory
    Experiment
35 //
    ---------------------------------------------------
36 // LaCrO3       3.00              2.82            2.80
37 // LaMnO3       4.00              3.74            3.90
38 // LaCoO3       5.00              4.16            4.60
39 //
    ---------------------------------------------------
```

Scilab code Exa 8.14 Magnetic structure of the solids from total energy

```
1 // Scilab code Ex8.14 Page:268 (2006)
2 clc; clear;
3 C = cell(4,4);
4 // Enter compound names
5 C(1,1).entries = 'LaTiO3';
6 C(2,1).entries = 'LaCrO3';
7 C(3,1).entries = 'LaFeO3';
8 C(4,1).entries = 'LaCoO3';
9 // Enter total energy difference w.r.t. ground state
    for Paramagnetics, mRyd
10 C(1,2).entries = 0.014;
```

```
11 C(2,2).entries = 158.3;
12 C(3,2).entries = 20.69;
13 C(4,2).entries = 0.000;
14 // Enter total energy difference w.r.t. ground state
       for Ferromagnetics, mRyd
15 C(1,3).entries = 0.034;
16 C(2,3).entries = 13.99;
17 C(3,3).entries = 0.006;
18 C(4,3).entries = 0.010;
19 // Enter total energy difference w.r.t. ground state
       for Antiferromagnetics, mRyd
20 C(1,4).entries = 0.000;
21 C(2,4).entries = 0.000;
22 C(3,4).entries = 0.000;
23 C(4,4).entries = 0.003;
24 printf("\
      n-------------------------------------------------------------
      ");
25 printf("\nSolid      Total energy difference (mRyd) (
      w.r.t. ground state)");
26 printf("\n
      ------------------------------------------------------
      ");
27 printf("\n             Paramagnetic     Ferromagnetic
       Antiferromagnetic ");
28 printf("\
      n-------------------------------------------------------------
      ");
29 for i = 1:1:4
30     printf("\n%s       %10.3f       %10.3f       %10.3f
          ", C(i,1).entries, C(i,2).entries, C(i,3).
          entries, C(i,4).entries);
31 end
32 printf("\
      n-------------------------------------------------------------
      ");
33 printf("\nAll the solids given above crystallize in
      the antiferromagnetic state except that of LaCoO3
```

```
    .");
34
35 // Result
36 //
    ------------------------------------------------------------------

37 // Solid      Total energy difference (mRyd) (w.r.t.
    ground state)
38 //
    ------------------------------------------------------

39 //              Paramagnetic     Ferromagnetic
    Antiferromagnetic
40 //
    ------------------------------------------------------------------

41 // LaTiO3            0.014            0.034
    0.000
42 // LaCrO3          158.300           13.990
    0.000
43 // LaFeO3           20.690            0.006
    0.000
44 // LaCoO3            0.000            0.010
    0.003
45 //
    ------------------------------------------------------------------

46 // All the solids given above crystallize in the
    antiferromagnetic state except that of LaCoO3.
```

Chapter 9

Superconductivity

Scilab code Exa 9.1 Critical field required to destroy superconductivity

```
1 // Scilab code Ex9.1 Page:278 (2006)
2 clc; clear;
3 H_c0 = 0.0803;     // Critical field at absolute zero
    , Tesla
4 T_c = 7.19;     // Transition temperature of specimen
     lead, Kelvin
5 T = 5;     // Temperature at which destruction of
    superconductivity is to be found, Kelvin
6 H_c = H_c0*[1-(T/T_c)^2];     // Critical field
    required to destroy superconductivity, Tesla
7 printf("\nCritical field required to destroy
    superconductivity = %6.4f T", H_c);
8
9 // Result
10 // Critical field required to destroy
    superconductivity = 0.0415 T
```

Scilab code Exa 9.2 Limiting magnetic field of Nb to serve as superconductor

```
// Scilab Code Ex9.2 Page:278 (2006)
clc; clear;
H0 = 1970;     // Critical field at absolute zero, Oe
T_c = 9.25;     // Transition temperature of specimen
     Nb, Kelvin
T = 4;     // Temperature at which destruction of
   superconductivity is to be found, Kelvin
H_c = H0*[1-(T/T_c)^2];     // Limiting magnetic
   field, Oe
printf("\nLimiting magnetic field of Nb to serve as
   superconductor = %4d Oe", round(H_c));

// Result
// Limiting magnetic field of Nb to serve as
   superconductor = 1602 Oe
```

Scilab code Exa 9.3 Transition temperature of a specimen

```
// Scilab Code Ex9.3 Page:278 (2006)
clc;clear;
T_1 = 14;     // Temperature, K
T_2 = 13;     // Temperature, K
H_c1 = 1.4e+05;     // Critical field at T_1, K
H_c2 = 4.2e+05;     // Critical field at T_2, K//As
   H_c1/H_c2 = (T_c^2-T_1^2)/(T_c^2-T_2^2), solving
   for T_c
T_c = sqrt((H_c2/H_c1*T_1^2 - T_2^2)/2);  // The
   superconducting transition temperature of a
   specimen, K
printf("\nTransition temperature of a specimen = %5
   .2f K", T_c);

// Result
// Transition temperature of a specimen = 14.47 K
```

Scilab code Exa 9.4 Coherence length of aluminium

```
1 // Scilab Code Ex9.4 Page:280 (2006)
2 clc;clear;
3 e = 1.6e-019;    // Energy equivalent of 1 eV, J/eV
4 E_g = 3.4e-04;     // Energy gap of aluminium, eV
5 v_F = 2.02e+08;     // Fermi velocity of aluminium,
    cm/sec
6 h_bar = 1.05e-034;     // Planck's constant
7 L = h_bar*v_F/(2*E_g*e);     // Coherence Length of
    aluminium, cm
8
9 printf("\nThe coherence length of aluminium = %4.2e
    cm", L);
10
11 // Result
12 // The coherence length of aluminium = 1.95e-04 cm
```

Scilab code Exa 9.6 Wavelength of photon required to break a Cooper pair

```
1 // Scilab Code Ex9.6 Page:284 (2006)
2 clc; clear;
3 h = 6.6e-034;      // Planck's constant, Js
4 e = 1.6e-019;        // Energy eqivalent of 1 eV, eV/
    J
5 k = 0.86e-004;  // Boltzmann constant, eV/K
6 T_c = 0.56;      // Critical temperature for
    superconducting Zr, K
7 E_g = 3.52*k*T_c;     // Energy gap of aluminium, J
8 c = 3e+08;       // Speed of light, m/s
9 lambda = h*c/(E_g*e);        // Wavelength of photon
    required to break a Cooper pair, m
```

```
10
11 printf("\nThe wavelength of photon required to break
       a Cooper pair = %3.1e m", lambda);
12
13 // Result
14 // The wavelength of photon required to break a
     Cooper pair = 7.3e-03 m (Answer given in the
     textbook is wrong)
```

Scilab code Exa 9.7 London penetration depth in Pb

```
1 // Scilab Code Ex9.7 :Page:285 (2006)
2 clc; clear;
3 Lambda_0 = 390;     // Penetration depth at absolute
     zero, angstorm
4 T_c = 7;     // Transition temperature of Pb, K
5 T = 2;       // Givn temperature, K
6 Lambda = Lambda_0*[1-(T/T_c)^2]^(-1/2);     // London
      penetration depth in Pb at 2K, angstorm
7 printf("\nThe London penetration depth in Pb at 2K =
      %7.4f angstorm", Lambda);
8 printf("\nThe London penetration depth at T = T_c
     becomes %d", %inf);
9
10 // Result
11 // The London penetration depth in Pb at 2K =
     406.9644 angstorm
12 // The London penetration depth at T = T_c becomes
     Inf
```

Scilab code Exa 9.8 Isotopic exponent in Isotopic effect of Hg

```
1 // Scilab Code Ex9.8: Page:286 (2006)
```

```
clc; clear;
M = [199.5 200.7 202.0 203.3];  // Isotopic mass of Hg, amu
T_c = [4.185 4.173 4.159 4.146];    // Critical temperature of Hg, kelvin
alpha = 0.5;    // Trial value of Isotopic exponent
// Accroding to isotopic effect, T_c = K*M^(-alpha), solving for K
K = T_c(1)/M(1)^(-alpha);  // Isoptopic coefficent
Tc = zeros(3);
for i = 2:1:4
    Tc(i-1) = K*M(i)^(-alpha);
    printf("\nTc(%d) = %5.3f", i, Tc(i-1));
end
if T_c(2)-Tc(1)<0.001 & T_c(3)-Tc(2)<0.001 & T_c(4)-Tc(3)<0.001 then
    printf("\nThe isotopic exponent in Isotopic effect of Hg = %3.1f", alpha);
end

// Result
// Tc(2) = 4.172
// Tc(3) = 4.159
// Tc(4) = 4.146
// The isotopic exponent in Isotopic effect of Hg = 0.5
```

Scilab code Exa 9.9 Transition temperature of isotope of Hg whose mass number is 1

```
// Scilab code Ex9.9 Page:286 (2006)
clc; clear;
M_1 = 202;    // Mass of first isotope of mercury, amu
M_2 = 199;    // Mass of second isotope of mercury, amu
```

```
T_c1 = 4.153;     // Transition temperature of first
   isotope of mercury, K
//As T_c1/T_c2 = (M_2/M_1)^1/2, solving for T_c2
T_c2 = sqrt(M_1/M_2)*T_c1;  //
printf("\nThe transition temperature of isotope of
   Hg whose mass number is %d = %5.3f K", M_2, T_c2)
   ;

// Result
// The transition temperature of isotope of Hg whose
    mass number is 199 = 4.184 K
```

Scilab code Exa 9.10 Constant of proportionality in Isotopc effect

```
// Scilab code Ex9.10 Page:287 (2006)
clc; clear;
alpha = 0.5;     // Isotopic exponent of Osmium
T_c = 0.655;     // Transition temperature of Osmium,
    K
M = 190.2;     // Mass of Osmium, amu
K = T_c*M^alpha;     // K is the constant of
   proportionality

printf("\nThe value of constant of proportionality =
    %4.2f ", K);

// Result
// The value of constant of proportionality = 9.03
```

Scilab code Exa 9.11 Transition temperature and energy gap of a material

```
// Scilab code Ex9.11 Page:298 (2006)
clc; clear;
```

```
3 k = 1.38e-023;    // Boltzmann constant, J/mol/K
4 e = 1.6e-019;       // Energy equivalent of 1 eV, eV/
    J
5 Theta_D = 96;     // Debye temperature, kelvin
6 N0 = 0.3678;     // Density of states at Fermi energy
7 V = 1;           // Volume of the material, metre
    cube
8 T_c = 1.14*Theta_D*exp(-1/(N0*V));     // Critical
    temperature of the material, K
9 Delta_0 = k*Theta_D/sinh(1/(N0*V)); // Energy gap at
     absolute zero, J
10 printf("\nThe transition temperature of a material =
    %4.2f K", T_c);
11 printf("\nThe energy gap of a material = %5.3e eV",
    Delta_0/e);
12
13 // Result
14 // The transition temperature of a material = 7.22 K
15 // The energy gap of a material = 1.097e-03 eV
```

Scilab code Exa 9.12 Transition temperature of a superconductor using McMillan for

```
1 // Scilab code Ex9.12 Page:298 (2006)
2 clc; clear;
3 Theta_D = 350;     // Debye temperature, kelvin
4 Lambda = 0.828;     // Electron-phonon coupling
    constant
5 mu_prime = 0.1373;     // Reduced mass of a
    superconductor, amu
6 T_c = Theta_D/1.45*exp(-1.04*(1+Lambda)/(Lambda-
    mu_prime*(1+0.62*Lambda)));     // Transition
    temperature of a superconductor using McMillan
    formula, K
7
8 printf("\nThe transition temperature of the
```

```
    superconductor using McMillan formula = %5.2f K",
     T_c);
9
10 // Result
11 // The transition temperature of the superconductor
    using McMillan formula = 11.26 K
```

Scilab code Exa 9.13 Superconducting transition temperature of a superconductor us

```
1 // Scilab code Ex9.13 : Page:298 (2006)
2 clc; clear;
3 Theta_D = 350;     // Debye temperature, kelvin
4 Lambda = 0.641;     // Electron-phonon coupling
    constant
5 mu_prime = 0.143;     // Reduced mass of a
    superconductor, amu
6 T_c = Theta_D/1.45*exp(-1.04*(1+Lambda)/(Lambda-
    mu_prime*(1+0.62*Lambda)));    // Superconducting
    transition temperature of a superconductor using
    mcMillan's formula, K
7
8 printf("\nThe superconducting transition temperature
     of a superconductor using McMillan formula = %5
    .3f K", T_c);
9
10 // Result
11 // The superconducting transition temperature of a
    superconductor using McMillan formula = 5.043 K
```

Scilab code Exa 9.15 Superconducting transition temperature of a borocarbide super

```
1 // Scilab code Ex9.15 Page:314 (2006)
2 clc; clear;
```

```
3 Theta_D = 490;     // Debye temperature, Kelvin
4 Lambda = 0.8;     // wavelength of a superconductor,
    angstorm
5 mu_prime = 0.13;     // Reduced mass of a
    superconductor, amu
6 T_c = Theta_D/1.45*exp(-1.04*(1+Lambda)/(Lambda-
    mu_prime*(1+0.62*Lambda)));
7 printf("\nThe superconducting transition temperature
     of a borocarbide superconductor = %4.1f K", T_c)
    ;
8
9 // Result
10 // The superconducting transition temperature of a
    borocarbide superconductor = 15.4 K
```

Scilab code Exa 9.16 Electron phonon coupling constant for a superconductor

```
1 // Scilab code Ex9.16 Page:314 (2006)
2 clc; clear;
3 T_c = 16.5;     // Transition temperature of a
    superconductor, K
4 Lambda = [0.7 0.8 0.9 1.0];     // Electron-phonon
    coupling constants at different Tc values
5 Theta_D = 503;     // Debye temperature, kelvin
6 mu_prime = 0.13;     // Reduced mass of a
    superconductor, amu
7 Tc = zeros(4);
8 printf("\n----------------------");
9 printf("\nLambda           Tc");
10 printf("\n----------------------");
11 for i = 1:1:4
12     Tc(i) = Theta_D/1.45*exp(-1.04*(1+Lambda(i))/(
        Lambda(i)-mu_prime*(1+0.62*Lambda(i))));
13     if abs(Tc(i) - 16.5) < 1.0 then
14         best_Lvalue = Lambda(i);
```

```
15      end
16      printf("\n%3.1f        %8.1f K", Lambda(i), Tc(i))
           ;
17  end
18  printf("\n--------------------");
19
20  printf("\nThe best electron-phonon coupling constant
        should be slightly above %3.1f ", best_Lvalue);
21
22  // Result
23  // ----------------------
24  // Lambda           Tc
25  // ----------------------
26  // 0.7              11.1 K
27  // 0.8              15.8 K
28  // 0.9              20.4 K
29  // 1.0              24.9 K
30  // ----------------------
31  // The best electron-phonon coupling constant should
        be slightly above 0.8
```

Scilab code Exa 9.17 Debye temperature of a BCS superconductor

```
1  // Scilab code Ex9.17 Page:317 (2006)
2  clc; clear;
3  T_c = 39.4;     // Transition temperature of a
      superconductor, K
4  Lambda = 1;     // Electron-phonon coupling constant
      for a superconductor
5  mu_prime= 0.15;     // Reduced mass of a
      superconductor, amu
6  // As T_c = Theta_D/1.45*exp(-1.04*(1+Lambda)/(
      Lambda-mu_prime*(1+0.62*Lambda))), solving for
      Theta_D
7  Theta_D = T_c*1.45*exp(1.04*(1+Lambda)/(Lambda-
```

```
    mu_prime*(1+0.62*Lambda)));
8
9 printf("\nThe Debye temperature of a BCS
    superconductor = %3d K", Theta_D);
10
11 // Result
12 // The Debye temperature of a BCS superconductor =
    891 K
```