Scilab Textbook Companion for Applied Physics by M. Arumugam[1]

Created by
Mathews George
B.ts
Others
M.g
College Teacher
None
Cross-Checked by
Reshma

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: Applied Physics

Author: M. Arumugam

Publisher: Anuradha Publications

Edition: 1

Year: 2015

ISBN: 9788189638726

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

Chapter 1

Bonding in Solids and Crystal Structures

Scilab code Exa 1.1 `Calculation of youngs modulus`

```
//Initialisation of variables
clc
//Variable declaration
a=7.68*10**-29;
r0=2.5*10**-10;      //radius(m)

//Calculation
b=a*(r0**8)/9;
y=((-2*a*r0**8)+(90*b))/r0**11;
E=y/r0;              //young's modulus(Pa)

//Result

printf('youngs modulus is %0.2f GPa',(E/10^9))
```

Scilab code Exa 1.2 `Find the Effective charge`

```
//Initialisation of variables
clc

d=((1.98)*10**-29)*1/3;        //dipole moment
b=(0.92);                      //bond length
EC=d/(b*10**-10);              //Effective charge

//Result
printf('Effective charge =%0.2f *10**-29 coulomb',((EC*10**19)))
```

Scilab code Exa 1.3 Find the Cohesive energy

```
//Initialisation of variables
clc

A=1.748                      //Madelung Constant
N=6.02*10**26                //Avagadro Number
e=1.6*10**-19
n=9.5
r=(0.324*10**-9)*10**3
E=8.85*10**-12
//Calculations
U=((N*A*(e)**2)/(4*%pi*E*r))*(1-1/n)        //Cohesive energy

//Result
printf('Cohesive energy =%0.2f *10**3 kJ/kmol \n",(U/10**3))
printf('//Answer varies due to rounding of numbers')
```

Scilab code Exa 1.4 Find the Coulomb energy

```
//variable declaration
I=5;                          //Ionisation energy
A=4;                          //Electron Affinity
e=(1.6*10**-19)
E=8.85*10**-12               //epsilon constant
r=0.5*10**-19                //dist between A and B

//Calculations
C=-(e**2/(4*%pi*E*r*e))/10**10     //Coulomb energy
E_c=I-A+C                                  //Energy required

//Result
printf('Coulomb energy =%0.2f eV\n',C)
printf('Energy required =%0.2f eV',E_c')
```

Scilab code Exa 1.5 Find the Distance of separation

```
//variable declaration
I=5.14;                       //Ionization energy
A=3.65;                       //Electron Affinity
e=(1.6*10**-19);
E=8.85*10**-12;
//calculations
E_c=I-A                            //Energy required
r=e**2/(4*%pi*E*E_c*e)        //Distance of separation

//Result
printf('Energy required=%0.2f eV \n",E_c)
printf('Distance of separation =%0.2f Angstrom",r/10**-10)
```

Scilab code Exa 1.6 Find the Bond Energy

```
//variable declaration
I=5.14;                          //Ionization energy
A=3.65;                          //Electron Affinity
e=(1.6*10**-19);
E=8.85*10**-12;
r=236*10**-12;

//Calculations
E_c=I-A                                //Energy required
C=-(e**2/(4*%pi*E*r*e))          //Potentential energy in eV
BE=-(E_c+C)                            //Bond Energy
//Result
printf('Energy required= %0.2f eV\n",E_c)
printf('Energy required =%0.1f eV\n",C)
printf('Bond Energy =%0.2f eV",BE)
```

Scilab code Exa 1.7 Find the density

```
//variable declaration
d=2.351                       //bond lenght
N=6.02*10**26                 //Avagadro number
n=8                           //number of atoms in unit cell
A=28.09                       //Atomin mass of silicon
m=6.02*10**26                 //1mole

//Calculations
a=(4*d)/sqrt(3)
p=(n*A)/((a*10**-10)*m)          //density

//Result
printf('a=%0.2fAngstorm\n',a)
printf('density =%0.2f kg/m**3\n',(p*10**16))
printf("//Answer given in the textbook is wrong")
```

Scilab code Exa 1.8 Find the radius of sphere

```
//Variable declaration


//Calculation
a1=4/sqrt(3);
R1=(a1/2)-1;                //radius of largest sphere
a2=4/sqrt(2);
R2=(a2/2)-1;          //maximum radius of sphere

//Result
printf('radius of largest sphere is %f*r\n',R1)
printf('maximum radius of sphere is %f*r',R2 )
```

Scilab code Exa 1.9 increase of density or the decrease of volume

```
//variable declaration
r1=1.258                //Atomic radius of BCC
r2=1.292                //Atomic radius of FCC

//calculations
a1=(4*r1)/sqrt(3)        //in BCC
b1=((a1)**3)*10**-30          //Unit cell volume
v1=(b1)/2                     //Volume occupied by one atom
a2=2*sqrt(2)*r2          //in FCC
b2=(a2)**3*10**-30                     //Unit cell volume
v2=(b2)/4                     //Volume occupied by one atom
```

```
v_c=((v1)-(v2))*100/(v1)      //Volume Change in %
d_c=((v1)-(v2))*100/(v2)      //Density Change in %

//Results
printf('a1=%0.3f Angstrom\n\n',(a1))
printf('Unit cell volume =a1**3 =%0.3f *10**-30 m**3\n',((b1)/10**-30))
printf('Volume occupied by one atom =%0.2f *10**-30 m**3\n',(v1/10**-30))
printf('a2=%0.2f\n Angstorm\n',(a2))
printf('Unit cell volume =a2**3 =%0.3f *10**-30 m**3\n',((b2)/10**-30))
printf('Volume occupied by one atom =%0.3f*10**-30 m**3\n',(v2/10**-30))
printf('Volume Change in percentage  =%0.3f\n',(v_c))
printf('Density Change in percentage =%0.3f\n',(d_c))
printf('Thus the increase of density or the decrease of volume is about 0.5 percentage")
```

Scilab code Exa 1.10 spacing between the nearest neighbouring ions

```
//variable declaration
n=4
M=58.5                    //Molecular wt. of NaCl
N=6.02*10**26             //Avagadro number
rho=2180                  //density

//Calculations
a=((n*M)/(N*rho))**(1/3)
s=a/2

//Result
printf('a=%0.3f*10**-9 metre\n",(a/10**-9))
printf('spacing between the nearest neighbouring
   ions =%0.3f nm",(s/10**-9))
```

Scilab code Exa 1.11 lattice constant

```
//variable declaration
n=4
A=63.55                      //Atomic wt. of NaCl
N=6.02*10**26                //Avagadro number
rho=8930                     //density

//Calculations
a=((n*A)/(N*rho))**(1/3)          //Lattice Constant

//Result
printf('lattice constant , a=%0.3f nm',(a*10**9))
```

Scilab code Exa 1.12 Density of iron

```
//variable declaration
r=0.123                       //Atomic radius
n=4
A=55.8                        //Atomic wt
a=2*sqrt(2)
N=6.02*10**26                //Avagadro number

//Calculations
rho=(n*A)/((a*r*10**-9)**3*N)

//Result
printf('Density of iron =%0.3fkg/m**-3",rho)
```

Chapter 2

Crystal Planes and Xray Diffraction

Scilab code Exa 2.1 Number of atoms per unit area

```
clc

//Variable declaration
a=mulf('2','R')

//Results
printf('i.Number of atoms per unit area of (100) plane= 1/(%d*R**2) ",2**2)
printf('\nii.Number of atoms per unit area of (110) plane=%f*R**2",2**2/(sqrt(2)))
printf('\niii.Number of atoms per unit area of (111) plane=%f*R**2",2**2/(sqrt(3)))
```

Scilab code Exa 2.2 Surface area of surfaces

```
clc
```

```



//Variable declaration
a=3.61*10**-7
BC=sqrt(2)/2
AD=(sqrt(6))/2
//Result
printf('i.Surface area of the face ABCD =%0.3f
   *10**-14 mm**2\n",(a**2*10**14))
printf('ii.Surface area of plane (110) =%0.3f*10**13
    atoms/mm**2\n",((2/(a*sqrt(2)*a)/10**13)))
printf('iii.Surface area of pane(111)=%0.3f*10**13
   atoms/mm**2",(2/(BC*AD*a**2)*10**-13))
```

Scilab code Exa 2.3 Calculate ratio

```
h1=1
k1=0
l1=0
h2=1
k2=1
l2=0
h3=1
k3=1
l3=1
a=1

//Calculations
d1=a/(sqrt(h1**2+k1**2+l1**2))
d2=a/(sqrt(h2**2+k2**2+l2**2))
d3=a/(sqrt(h3**2+k3**2+l3**2))

//Result
printf('d1 =%0.2f\n",d1 )
printf(’d2 =%0.2f\n”,(d2))
printf(’d3 =%0.2f\n”,(d3))
printf(’d1:d2:d3 =%0.2f:%0.2f:%0.2f”,d1,(d2),d3)
```

Scilab code Exa 2.4 Calcutate length

```

clc



//Variable declaration
h=2
k=2
l=0
a=450

//Calculations
d=a/(sqrt(h**2+k**2+l**2))

//Result
printf(’d(220) =%0.3fpm\n’,(d))
```

Scilab code Exa 2.5 Calcutate two lengths

```

clc



//Variable declaration
a=3.615
r=1.278
```

```
h=1
k=1
l=1

//Calculations
a=(4*r)/sqrt(2)
d=a/(sqrt(h**2+k**2+l**2))

//Result
printf('a =%0.3fAngstroms\n',(a))
printf('d =%0.3fAngstroms\n',(d))
```

Scilab code Exa 2.7 Calcutate lengths of two spheres

```
clc



//Variable declaration
n=1
lamda=1.54
theta=32*%pi/180
h=2
k=2
l=0

//Calculations
d=(n*lamda*10**-10)/(2*sin(theta))    //derived from 2dsin(theta)=n*l
a=d*(sqrt(h**2+k**2+l**2))

//Results
printf('d =%0.3f *10**-10 m\n',(d*10**10))
printf('a =%0.3f *10**-10 m\n',(a*10**10))
```

Scilab code Exa 2.8 `length by diffraction given by angles`

```
clc



//Variable declaration
lamda=0.58
theta1=6.45*%pi/180
theta2=9.15*%pi/180
theta3=13*%pi/180

//Calculations
dbyn1=lamda/(2*(sin(theta1)))
dbyn2=lamda/(2*sin(theta2))
dbyn3=lamda/(2*sin(theta3))

//Results
printf('i.  d/n =%0.3f Angstroms\n',(dbyn1))
printf('ii. d/n =%0.3f Angstroms\n',(dbyn2))
printf('iii.d/n =%0.3f Angstroms\n',(dbyn3))
```

Scilab code Exa 2.9 `Given wavelength calculate n`

```
clc



//Variable declaration
d=1.18
theta=90*%pi/180
lamda=1.540
```

```

//Calculations
n=(2*d*sin(theta))/lamda

//Result
printf('n =%0.3f \n',(n))
```

Scilab code Exa 2.10 Calcutate length

```
clc



//Variable declaration
lamda=0.58
theta=9.5*%pi/180
n=1
d=0.5                //d200=a/sqrt(2**2+0**2+0**2)=0.5a
//Calculations
a=n*lamda/(2*d*sin(theta))       //2*d*sin(theta)=n*lamda

//Result
printf('a =%0.3f Angstorms\n',(a))
```

Scilab code Exa 2.11 Calculate the angle

```

clc



//Variable declaration
```

```
lamda=0.842
n1=1
q=(8+(35/60))*(%pi/180)
n2=3
d=1
//Calculations
//n*lamda=2*d*sin(theta)
//n1*0.842=2*d*sin(q)
//n3*0.842=2*d*sin(theta3)
//Dividing both the eauations, we get
//(n2*lamda)/(n1*lamda)=2*d*sin(theta3)/2*d*sin(q)
theta3=asin((((n2*lamda)/(n1*lamda))*(2*d*sin(q)))/(2*d))
d=theta3*180/%pi;
a_d=int(d);
a_m=(d-int(d))*60

//Result
printf('sin(theta3) =%0.3f %0.3f',a_d,a_m)
```

Scilab code Exa 2.12 Calculate h and k values

```
clc
//Variable declaration
a=3.16
lamda=1.54
n=1
theta=20.3*%pi/180

//Calculations
d=(n*lamda)/(2*sin(theta))
x=a/d                                //let sqrt(h**2+k**2+l**2)=x

//Result
```

```
printf('d =%0.3f Angstorms\n',(d))
printf('sqrt(h**2+k**2+l**2) =%0.3f \n',(x))
printf('Therefore, h**2+k**2+l**2 =sqrt(2)\n')
printf('h =1, k=1')
```

Scilab code Exa 2.13 Calculate wavelength and energy

```

//Variable declaration
n=4
A=107.87
rho=10500
N=6.02*10**26
h=1;
k=1;
l=1;
H=6.625*10**-34
e=1.6*10**-19
theta=(19+(12/60))*%pi/180
C=3*10**8
//Calculations
a=((n*A)/(rho*N))**(1/3)*10**10
d=a/sqrt(h**2+k**2+l**2)
lamda=2*d*sin(theta)
E=(H*C)/(lamda*10**-10*e)

//Result
printf('a =%0.3f Angstroms \n',(a))
printf('d =%0.3f Angstroms\n',(d))
printf('lamda =%0.3f Angstroms\n',(lamda))
printf('E =%0.3f *10**3 eV\n',(E/10**3))
```

Scilab code Exa 2.14 Calculate wavelength and angle

```

//Variable declaration
a=4.57
h=1
k=1
l=1
lamda=1.52
twotheta=33.5*%pi/180
r=5                   //radius
//Calculations
d=a/(h**2+k**2+l**2)**(1/2)
sintheta=lamda/(2*d)
X=r/tan(twotheta)

//Result
printf('d =%0.3f Angstorms\n',(d))
printf('sin(theta)=%0.3f \n',(sintheta))
printf('X =%0.3f cm\n',(X))
```

Chapter 3

Defects In Solids

Scilab code Exa 3.1 `The number of vacancies per kilomole of copper`

```
//Variable declaration
N=6.023*10**26
deltaHv=120
B=1.38*10**-23
k=6.023*10**23

//Calculations
n0=0                                          // 0 in denominator
n300=N*exp(-deltaHv*10**3/(k*B*300))      //The number of vacancies per kilomole of copper
n900=N*exp(-(deltaHv*10**3)/(k*B*900))

//Results
printf('at 0K, The number of vacancies per kilomole of copper is %0.3f' ,n0)
printf('at 300K, The number of vacancies per kilomole of copper is %0.3f *10**5\n',(n300/10**5))
printf('at 900K, The numb ber of vacancies per kilomole of copper is %0.3f *10**19\n',(n900
   /10**19))
```

Scilab code Exa 3.2 Fraction of vacancies at 1000 degree

```

//Variable declaration
F_500=1*10**-10

T1=500+273
T2=1000+273


//Calculations
lnx=log(F_500)*T1/T2;
x=exp(lnx)

printf('Fraction of vacancies at 1000 degrees C =%0
   .3f *10**-7\n',(x*10**7))
```

Scilab code Exa 3.3 The concentration of Schottky defects

```
//Variable declaration
a=(2*2.82*10**-10)
delta_Hs=1.971*1.6*10**-19
k=1.38*10**-23
T=300
e=2.718281
//Calculations
V=a**3                                  //Volume of unit
   cell of NaCl
N=4/V                                   //Total number of
   ion pairs
n=N*e**-(delta_Hs/(2*k*T))
```

```

//Result
printf('Volume of unit cell of NaCl =%0.3f *10**-28 m**3 \n',(V*10**28))
printf('Total number of ion pairs N =%0.3f *10**28\n',(N/10**28))
printf('The concentration of Schottky defects per m**3 at 300K =%0.3f *10**11\n',(n/10**11))
```

Scilab code Exa 3.4 amount of climb down by the dislocation

```
//Variable declaration
N=6.023*10**23
delta_Hv=1.6*10**-19
k=1.38*10**-23
T=500
mv=5.55;       //molar volume
x=2*10**-8;      //numbber of cm in 1 angstrom

//Calculations
n=N*exp(-delta_Hv/(k*T))/mv
a=(n/(5*10**7*10**6))*x;

//Result
printf('The number that must be created on heating from 0 to 500K is n=%0.3f *10**12 per cm**3\n',(n/10**12)) //into cm**3
printf('As one step is 2 Angstorms , 5*10**7 vacancies are required for 1cm')
printf('The amount of climb down by the dislocation is %0.3f cm',a*10**8)
```

Scilab code Exa 3.5 Velocity and wavelength

```
//Variable declaration
KE=10                  //Kinetic Energy of neutron in keV
m=1.675*10**-27
h=6.625*10**-34
//Calculations
KE=10**4*1.6*10**-19       //in joule
v=((2*KE)/m)**(1/2)        //derived from KE=1/2*m*v**2
lamda=h/(m*v)
//Results
printf('Velocity =%0.3f  *10**6 m/s \n  ',(v/10**6))
printf('Wavelength =%0.3f  Angstorm \n ',(lamda*10**10))
```

Scilab code Exa 3.6 Momentum and de Brolie wavelength

```

//Variable declaration
E=2*1000*1.6*10**-19        //in joules
m=9.1*10**-31
h=6.6*10*10**-34

//Calculations
p=sqrt(2*m*E)
lamda= h/p

//Result
printf('Momentum%0.3f   \n ',(p*10**23))
printf('de Brolie wavelength =%0.3f *10**-11 m  \n ',(lamda*10**10))
```

Scilab code Exa 3.7 wavelength

```
//Variable declaration
M=1.676*10**-27              //Mass of neutron
m=0.025
v=1.602*10**-19
h=6.62*10**-34

//Calculations
mv=(2*m*v)**(1/2)
lamda=h/(mv*M**(1/2))

//Result
printf('wavelength =%0.3f  Angstorm \n ',(lamda
   *10**10))
```

Scilab code Exa 3.8 Wavelength

```
//Variable declaration
V=10000

//Calculation
lamda=12.26/sqrt(V)

//Result
printf('Wavelength =%0.3f Angstorm' ,lamda)
```

Scilab code Exa 3.9 The permitted electron energies

```
//Variable declaration
e=1.6*10**-19;   //charge of electron(coulomb)
L=10**-10             //1Angstrom=10**-10 m
n1=1;
n2=2;
n3=3;
```

```
h=6.626*10**-34
m=9.1*10**-31
L=10**-10

//Calculations
E1=(h**2)/(8*m*L**2*e)
E2=4*E1
E3=9*E1
//Result
printf('The permitted electron energies =%0.3f  *n**2 eV \n ',(E1))
printf('E1=%0.3f  eV \n ',(E1))
printf('E2=%0.3f  eV \n ',(E2))
printf('E3=%0.3f  eV \n ',(E3))
printf('//Answer varies due to ing of numbers")
```

Scilab code Exa 3.10 Calculate delta

```
//Variable declaration
i=1*10**-10;     //interval
L=10*10**-10;    //width

//Calculations
si2=2*i/L;

//Result
printf('si**2 delta(x)=%0.3f ' ,si2)
```

Scilab code Exa 3.11 Calculate energy difference

```
//Variable declaration
nx=1
ny=1
```

```
nz=1
a=1
h=6.63*10**-34
m=9.1*10**-31

//Calculations
E1=h**2*(nx**2+ny**2+nz**2)/(8*m*a**2)
E2=(h**2*6)/(8*m*a**2)                    //nx**2+ny**2+nz**2=6
diff=E2-E1
//Result
printf('E1 =%0.3f  *10**-37 Joule \n ',(E1*10**37))
printf('E2 =%0.3f  *10**-37 Joule \n ',(E2*10**37))
printf('E2-E1 =%0.3f *10**-37 J  \n ',(diff*10**37))
```

Scilab code Exa 3.12 `Calculate energy`

```
//Variable declaration
m=1.67*10**-27
a=10**-14
h=1.054*10**-34

//Calculations
E1=(1*%pi*h)**2/(2*m*a**2)

//Result
printf('E1 =%0.3f *10**-13 J  \n ',(E1*10**13))
```

Scilab code Exa 3.13 `Integration`

```
//Variable declarations
k=1;

```

```
//Calculations

a=integrate('2*k*exp(-2*k*x)','x',2/k,3/k)
//Result
printf('a=%0.3f \n ',(a))
```

Chapter 4

Electron Theory of Metals

Scilab code Exa 4.1 energy difference

```


//Variable declaration
m=9.1*10**-31;       //mass(kg)
nx=1;
ny=1
nz=1
n=6;
a=1;       //edge(m)
h=6.63*10**-34;       //planck's constant
k=1.38
//Calculation
E1=h**2*(nx**2+ny**2+nz**2)/(8*m*a**2);
E2=h**2*n/(8*m*a**2);
E=E2-E1;                  //energy difference(J)
T=(2*E2*10**37)/(3*k*10**-23)
//Result
printf('energy difference is%0.3f  *10**-37 J \n ',(
   E*10**37))
printf('3/2*k*T = E2 =%0.3f  *10**-37 J \n ',(E2
   *10**37))
```

```
printf('T =%0.3f *10**-14 K  \n ',(T/10**23))
```

Scilab code Exa 4.2 Calculate temperature

```
//Variable declaration
y=1/100;     //percentage of probability
x=0.5*1.6*10**-19;      //energy(J)
k=1.38*10**-23;        //boltzmann constant

//Calculation
xbykT=log((1/y)-1);
T=x/(k*xbykT);        //temperature(K)

//Result
printf('temperature is %0.3f K ',int(T))
printf('answer varies due to ing off errors')
```

Scilab code Exa 4.3 Calculate fermi energy

```
//Variable declaration
d=970;      //density(kg/m**3)
Na=6.02*10**26;     //avagadro number
w=23;     //atomic weight
m=9.1*10**-31;      //mass(kg)
h=6.62*10**-34;      //planck's constant

//Calculation
N=d*Na/w;     //number of atoms/m**3
x=h**2/(8*m);
y=(3*N/%pi)**(2/3)
EF=x*y;      //fermi energy(J)

//Result
```

```
printf('fermi energy is %0.3f eV \n ',(EF/(1.6*10**-19)))
```

Scilab code Exa 4.4 Find energy

```
//Variable declaration
kT=1;
E_EF=1;

//Calculations
p_E=1/(1+exp(E_EF/kT))

//Result
printf('p(E) =%0.3f  \n ',(p_E))
```

Scilab code Exa 4.5 Number of states

```
//Variable declarations
m=9.1*10**-31
h=6.626*10**-34
Ef=3.1
Ef1=Ef+0.02
e=1.6*10**-19
//Calculations

    N=integrate('%pi*((8*m)**(3/2))*(E**(1/2)*e**(3/2))/(2*(h**3))','E',Ef,Ef1)

//Result
printf('N =%0.3f *10**26 states \n ',(N*10**-26))
```

Scilab code Exa 4.6 `mean free collision time`

```
//Variable declaration
N=6.023*10**26                          //Avagadro number
D=8960                                  //density
F_e=1                                   //no.of free electrons per atom
W=63.54                                 //Atomic weight
i=10
e=1.602*10**-19
m=9.1*10**-31
rho=2*10**-8
Cbar=1.6*10**6                          //mean thermal velocity(m/s)

//Calculations
n=(N*D*F_e)/W
A=%pi*0.08**2*10**-4
Vd=i/(A*n*e)                             //Drift speed
Tc=m/(n*(e**2)*rho)
lamda=Tc*Cbar

//Result
printf('n =%0.3f  *10**28 /m**3 \n ',(n/10**28))
printf('The drift speed Vd =%0.3f *10**-5 m/s  \n ',(Vd*10**5))
printf('The mean free collision time Tc =%0.3f *10**-14 seconds \n ',(Tc*10**14))
printf('Mean free path =%0.3f *10**-8 m""(answer varies due to ing off errors)  \n ',(lamda*10**8))
```

Scilab code Exa 4.7 `The mean free collision time`

```
//Variable declaration
```

```
n=8.5*10**28
e=1.602*10**-19
t=2*10**-14
m=9.1*10**-31

//Calculations
Tc=n*(e**2)*t/m

//Result
printf('The mean free collision time =%0.3f *10**7 ohm**-1 m**-1 \n ',(Tc/10**7))
```

Scilab code Exa 4.8 Relaxation time

```
//Variable declaration
e=1.6*10**-19
E=1                         //(V/m)
rho=1.54*10**-8
n=5.8*10**28
m=9.1*10**-31
//Calculations
T=m/(rho*n*e**2)
Me=(e*T)/m
Vd=Me*E

//Result
printf('Relaxation time =%0.3f *10**-14 second \n ',(T*10**14))
printf('Mobility =%0.3f *10**-3 m**2/volt-s \n ',(Me*10**3))
printf('Drift Velocity=%0.3f m/s \n ',(Vd*100))
```

Scilab code Exa 4.9 Temperature coefficient of resistivity

```

//Variable declaration
rho_r=0
T=300
rho=1.7*10**-18

//Calculations
a=rho/T
rho_973=a*973

//Results
printf('Temperature coefficient of resistivity,a =%0.3f   \n',(a*10**21))
printf('rho_973 =%0.3f *10**-8 ohm-m  \n',(rho_973*10**18))
```

Scilab code Exa 4.10 `Increase in resistivity in copper`

```
//Variable declaration
rho1=1.2*10**-8
p1=0.4
rho2=0.12*10**-8
p2=0.5
rho3=1.5*10**-8
//Calculations
R=(rho1*p1)+(rho2*p2)
R_c=R+rho3

//Results
printf('Increase in resistivity in copper =%0.3f *10**-8 ohm m  \n',(R*10**8))
printf('Total resistivity of copper alloy =%0.3f *10**-8 ohm m  \n',(R_c*10**8))
printf('The resistivity of alloy at 3K =%0.3f *10**-8 ohm m \n',(R*10**8))
```

Chapter 5

Dielectric Properties and Magnetic Properties

Scilab code Exa 5.1 `insulation resistance`

```
//Variable declaration
rho=5*10**16;    //resistivity(ohm m)
l=5*10**-2;      //thickness(m)
b=8*10**-2;      //length(m)
w=3*10**-2;      //width(m)

//Calculation
A=b*w;     //area(m**2)
Rv=rho*l/A;
X=l+b;       //length(m)
Y=w;       //perpendicular(m)
Rs=Rv*X/Y;
Ri=Rs*Rv/(Rs+Rv);        //insulation resistance(ohm)


printf('insulation resistance is %0.3f *10**18 ohm'
   ,(Ri/10**18))
printf('answer varies due to rounding off errors')
```

Scilab code Exa 5.2 polarisability of He

```
//Variable declaration
epsilon0=8.84*10**-12;
R=0.55*10**-10;     //radius(m)
N=2.7*10**25;       //number of atoms

//Calculation
alpha_e=4*%pi*epsilon0*R**3;     //polarisability of He(farad m**2)
epsilonr=1+(N*alpha_e/epsilon0);       //relative permittivity

//Result
printf('polarisability of He is  %0.3f *10**-40 farad m**2\n',(alpha_e*10**40))
printf('relative permittivity is  %0.3f \n',(epsilonr))
printf('answer varies due to ing off errors')
```

Scilab code Exa 5.3 total dipole moment

```
//Variable declaration
A=360*10**-4;     //area(m**2)
V=15;     //voltage(V)
C=6*10**-6;      //capacitance(farad)
epsilonr=8;
epsilon0=8.84*10**-12;

//Calculation
E=V*C/(epsilon0*epsilonr*A);      //field strength(V/m)
```

```
dm=epsilon0*(epsilonr-1)*V*A;      //total dipole moment(Cm)

//Result
printf('field strength is %0.3f *10**7 V/m\n',(E/10**7))
printf('total dipole moment is %0.3f *10**-12 Cm\n',(dm*10**12))
```

Scilab code Exa 5.4 the complex polarizability

```
//Variable declaration
epsilonr=4.36;          //dielectric constant
t=2.8*10**-2;           //loss tangent(t)
N=4*10**28;             //number of electrons
epsilon0=8.84*10**-12;

//Calculation
epsilon_r = epsilonr*t;
epsilonstar = (complex(epsilonr,-epsilon_r));
alphastar = (epsilonstar-1)/(epsilonstar+2);
alpha_star = 3*epsilon0*alphastar/N;              //complex polarizability(Fm**2)

//Result
printf('the complex polarizability is  %0.3f *10**-40 F-m**2    \n',alpha_star*10**40)
printf('answer cant be rouned off to 2 decimals as given in the textbook. Since it is a complex number and complex cant be converted to float')
```

Scilab code Exa 5.5 temperature rise

```
//Variable declaration
El=10**-2*50;        //energy loss(J)
H=El*60;       //heat produced(J)
d=7.7*10**3;     //iron rod(kg/m**3)
s=0.462*10**-3;     //specific heat(J/kg K)

//Calculation
theta=H/(d*s);      //temperature rise(K)

//Result
printf('temperature rise is %0.3f K \n',(theta))
```

Scilab code Exa 5.6 `magnetic field at the centre`

```
//Variable declaration
e=1.6*10**-19;     //charge(coulomb)
new=6.8*10**15;    //frequency(revolutions per second)
mew0=4*%pi*10**-7;
R=5.1*10**-11;      //radius(m)

//Calculation
i=(e*new);            //current(ampere)
B=mew0*i/(2*R);     //magnetic field at the centre(weber/m**2)
A=%pi*R**2;
d=i*A;        //dipole moment(ampere/m**2)

//Result
printf('magnetic field at the centre is %0.3f weber/m**2\n',(B))
printf('dipole moment is %0.3f *10**-24 ampere/m**2\n',(d*10**24))
```

Scilab code Exa 5.7 intensity of magnetisation

```
//Variable declaration
chi=0.5*10**-5;     //magnetic susceptibility
H=10**6;      //field strength(ampere/m)
mew0=4*%pi*10**-7;

//Calculation
I=chi*H;      //intensity of magnetisation(ampere/m)
B=mew0*(I+H);     //flux density in material(weber/m**2)

//Result
printf('intensity of magnetisation is %0.3f ampere/m \n',I)
printf('flux density in material is %0.3f weber/m**2 \n',(B))
```

Scilab code Exa 5.8 number of Bohr magnetons

```
//Variable declaration
B=9.27*10**-24;       //bohr magneton(ampere m**2)
a=2.86*10**-10;       //edge(m)
Is=1.76*10**6;        //saturation value of magnetisation(ampere/m)

//Calculation
N=2/a**3;
mew_bar=Is/N;       //number of Bohr magnetons(ampere m**2)
mew_bar=mew_bar/B;       //number of Bohr magnetons(bohr magneon/atom)
```

```

//Result
printf('number of Bohr magnetons is %0.3f bohr magneon/atom\n',(mew_bar))
```

Scilab code Exa 5.9 average magnetic moment

```
//Variable declaration
mew0=4*%pi*10**-7;
H=9.27*10**-24;       //bohr magneton(ampere m**2)
beta=10**6;       //field(ampere/m)
k=1.38*10**-23;     //boltzmann constant
T=303;     //temperature(K)

//Calculation
mm=mew0*H*beta/(k*T);     //average magnetic moment(bohr magneton/spin)

//Result
printf('average magnetic moment is %0.3f *10**-3 bohr magneton/spin\n',(mm*10**3))
```

Scilab code Exa 5.10 hysteresis loss

```
//Variable declaration
A=94;         //area(m**2)
vy=0.1;       //value of length(weber/m**2)
vx=20;        //value of unit length
n=50;         //number of magnetization cycles
d=7650;       //density(kg/m**3)

//Calculation
h=A*vy*vx;        //hysteresis loss per cycle(J/m**3)
```

```
hs=h*n;         //hysteresis loss per second(watt/m**3)
pl=hs/d;        //power loss(watt/kg)

//Result
printf('hysteresis loss per cycle is  %0.3f    J/m**3 \n',h)
printf('hysteresis loss per second is  %0.3f    watt/m**3  \n',hs)
printf('power loss is %0.3f watt/kg\n',(pl))
```

Chapter 6

Semiconductors and Superconductivity

Scilab code Exa 6.1 `number of electron hole pairs`

```
//Variable declaration
ni1=2.5*10**19;     //number of electron hole pairs
T1=300;      //temperature(K)
Eg1=0.72*1.6*10**-19;     //energy gap(J)
k=1.38*10**-23;      //boltzmann constant
T2=310;     //temperature(K)
Eg2=1.12*1.6*10**-19;     //energy gap(J)

//Calculation
x1=-Eg1/(2*k*T1);
y1=(T1**(3/2))*exp(x1);
x2=-Eg2/(2*k*T2);
y2=(T2**(3/2))*exp(x2);
ni=ni1*(y2/y1);           //number of electron hole
   pairs

//Result
printf('number of electron hole pairs is %0.3f
   *10**16 per cubic metre    \n',(ni/10**16))
```

```
printf('answer varies due to ing off errors')
```

Scilab code Exa 6.2 `intrinsic conductivity`

```
//Variable declaration
w=72.6;     //atomic weight
d=5400;     //density(kg/m**3)
Na=6.025*10**26;     //avagadro number
mew_e=0.4;     //mobility of electron(m**2/Vs)
mew_h=0.2;     //mobility of holes(m**2/Vs)
e=1.6*10**-19;
m=9.108*10**-31;     //mass(kg)
ni=2.1*10**19;       //number of electron hole pairs
Eg=0.7;     //band gap(eV)
k=1.38*10**-23;     //boltzmann constant
h=6.625*10**-34;     //plancks constant
T=300;      //temperature(K)

//Calculation
sigmab=ni*e*(mew_e+mew_h);     //intrinsic conductivity(ohm-1 m-1)
rhob=1/sigmab;      //resistivity(ohm m)
n=Na*d/w;      //number of germanium atoms per m**3
p=n/10**5;    //boron density
sigma=p*e*mew_h;
rho=1/sigma;

//Result
printf('intrinsic conductivity is %0.3f   *10**4 ohm-1 m-1     \n',(sigma/10**4))
printf('intrinsic resistivity is %0.3f    *10**-4 ohm m    \n',(rho*10**4))
printf('answer varies due to ing off errors')
printf('number of germanium atoms per m**3 is %0.3f *10**28     \n',(n/10**28))
```

Scilab code Exa 6.3 charge carrier density

```

//Variable declaration
e=1.6*10**-19;
RH=3.66*10**-4;      //hall coefficient(m**3/coulomb)
sigma=112;        //conductivity(ohm−1 m−1)

//Calculation
ne=3*%pi/(8*RH*e);     //charge carrier density(per m**3)
mew_e=sigma/(e*ne);        //electron mobility(m**2/Vs)

//Result
printf('charge carrier density is %0.3f  *10**22 per m**3   \n',int(ne/10**22))
printf('electron mobility is %0.3f    m**2/Vs    \n',(mew_e))
```

Scilab code Exa 6.4 conductivity during donor impurity

```
//Variable declaration
mew_e=0.13;     //mobility of electron(m**2/Vs)
mew_h=0.05;     //mobility of holes(m**2/Vs)
e=1.6*10**-19;
ni=1.5*10**16;       //number of electron hole pairs
N=5*10**28;

//Calculation
sigma1=ni*e*(mew_e+mew_h);     //intrinsic conductivity(ohm−1 m−1)
```

```
ND=N/10**8;
n=ni**2/ND;
sigma2=ND*e*mew_e;      //conductivity(ohm-1 m-1)
sigma3=ND*e*mew_h;      //conductivity(ohm-1 m-1)

//Result
printf('intrinsic conductivity is %0.3f *10**-3 ohm-1 m-1 %0.3f \n',(sigma1*10**3),sigma2)
printf('conductivity during donor impurity is %0.3f ohm-1 m-1 \n',sigma2)
printf('conductivity during acceptor impurity is %0.3f ohm-1 m-1',int(sigma3))
```

Scilab code Exa 6.5 conductivity

```
//Variable declaration
e=1.6*10**-19;
Eg=0.72;     //band gap(eV)
k=1.38*10**-23;     //boltzmann constant
T1=293;      //temperature(K)
T2=313;      //temperature(K)
sigma1=2;     //conductivity(mho m-1)

//Calculation
x=(Eg*e/(2*k))*((1/T1)-(1/T2));
y=(x/2.303);
z=(log10(sigma1));
log_sigma2=y+z;
sigma2=10**log_sigma2;      //conductivity(mho m-1)

//Result
printf('conductivity is %0.3f   mho m-1      \n',(sigma2))
```

Scilab code Exa 6.6 Concentration

```
//Variable declaration
ni=1.5*10**16
mu_n=1300*10**-4
mu_p=500*10**-4
e=1.6*10**-19
sigma=3*10**4

//Calculations
//Concentration in N-type
n1=sigma/(e*mu_n)
p1=ni**2/n1
//Concentration in P-type
p=sigma/(e*mu_p)
n2=(ni**2)/p

//Result
printf('a)Concentration in N-type\n ')
printf('n = %0.3f      *10**24 m**-3    \n',(n1
    *10**-24))
printf('Hence p = %0.3f      *10**8 m**-3    \n',(p1
    /10**8))
printf('b)Concentration in P-type\n')
printf('p = %0.3f       *10**24 m**-3   \n',(p/10**24)
    )
printf('Hence n = %0.3f    *10**8 m**-3      \n',(n2
    /10**8))
```

Scilab code Exa 6.7 Current density

```
//Variable declaration
```

```
i=10**-2
A=0.01*0.001
RH=3.66*10**-4
Bz=0.5

//Calculations
Jx=i/A
Ey=RH*(Bz*Jx)
Vy=Ey*0.01

//Result
printf('Jx = %0.3f    ampere/m**2  \n',Jx)
printf('Ey = %0.3f      V/m    \n',(Ey))
printf('Vy = %0.3f      mV    \n',(Vy*10**3))
```

Scilab code Exa 6.8 Position of fermi level

```
//Variable declaration
Ev=0
Ec=1.12
k=1.38*10**-23
T=300
mh=0.28
mc=0.12
e=1.6*10**-19
//Calculations
Ef=((Ec+Ev)/2)+((3*k*T)/(4*e))*log(mh/mc)

//Result
printf('Position of fermi level = %0.3f      eV    \n',(Ef))
```

Scilab code Exa 6.9 Conductivity of intrinsic germanium at 300K

```
//Variable declaration
ni=2.5*10**19
mu_e=0.38
mu_h=0.18
e=1.6*10**-19

//Calculations
sigmai=ni*e*(mu_e+mu_h)

//Result
printf('Conductivity of intrinsic germanium at 300K
    = %0.3f    ohm**-1 m**-1    \n',(sigmai))
```

Scilab code Exa 6.10 Conductivity

```
//Variable declaration
m=9.1*10**-31
k=1.38*10**-23
T=300
h=6.626*10**-34
Eg=1.1
e=1.6*10**-19
mu_e=0.48
mu_h=0.013
//Calculations
ni=2*((2*%pi*m*k*T)/h**2)**(3/2)*exp(-(Eg*e)/(2*k*T)
    )
sigma=ni*e*(mu_e+mu_h)

//Result
printf('Conductivity = %0.3f    *10**-3 ohm**-1 m
    **-1   \n',(sigma*10**3))
```

Scilab code Exa 6.11 The electron concentration

```
//Variable declaration
Na=5*10**23
Nd=3*10**23
ni=2*10**16
//Calculations
p=((Na-Nd)+(Na-Nd))/2

//Result
printf('p = %0.3f     *10**23 m**-3 \n',p*10**-23)
printf('The electron concentration is given by n = %0.3f    *10**9 m**-3   \n',ni**2/p*10**-9)
```

Scilab code Exa 6.12 resistance

```

//Variable declaration
Vh=37*10**-6
thick=1*10**-3
width=5
Iy=20*10**-3
Bz=0.5

//Calculations
Rh=(Vh*width*thick)/(width*Iy*Bz)

//Result
printf('Rh = %0.3f       *10**-6 C**-1 m**3    \n',(Rh*10**6))
```

Scilab code Exa 6.13 Calculate Dn and Dp

```

//Variable declaration
Vt=0.0258
mu_n=1300
mu_p=500

//Calculations
Dn=Vt*mu_n
Dp=Vt*mu_p

//Result
printf('Dn = %0.3f    cm**2 s**-1   \n',Dn)
printf('Dp = %0.3f cm**2 s**-1     \n',Dp)
```

Scilab code Exa 6.14 Electrical Conductivity

```

//Variable declaration
ni=1.5*10**16
Nd=2*10**19
e=1.602*100**-19
mu_n=0.12

//Calculations
p=ni**2/Nd
E_c=e*Nd*mu_n

//Result
printf('The hole concentration p = %0.3f      *10**13/m**3    \n',(p*10**-13))
printf('n= Nd = %0.3f      *10**19    \n',(Nd*10**-19))
printf('Electrical Conductivity = %0.3f       ohm**-1 m**-1   \n',(E_c*10**19))
```

Scilab code Exa 6.15 Current density

```

//Variable declaration
N=1/60
e=1.6*10**-19
ni=2.5*10**13
b=5*10**13
E=2

//Calculations
n=(b+sqrt(2*b**2))/2
mu_p=N/(3*e*ni)
mu_i=2*mu_p
np=ni**2
p=(ni**2)/n
e=1.6*10**-19
E=2
J=(e*E)*((n*mu_i)+(p*mu_p))
//Result
printf('mu_p= %0.3f    cm**2/V-s      \n',(mu_p))
printf('n= %0.3f      *10**13/cm**3    \n',(n/10**13))
printf('p= %0.3f      *10**13/cm**3    \n',(p*10**-13))
printf('J= %0.3f      A/m**2      \n',(J*10**4))
printf('//Answer varies due to ing of numbers')
```

Scilab code Exa 6.16 Drift velocity

```

//Variable declaration
rho=47*10**-2
```

```
e=1.6*10**-19
mu_n=0.39
mu_p=0.19
E=10**4

//Calculations
ni=1/(rho*e*(mu_n+mu_p))
Dh=mu_p*E
De=mu_n*E

//Results
printf('ni = %0.3f     *10**19 /m**3   \n',(ni/10**19))
printf('Drift velocity of holes %0.3f   ms**-1  \n',Dh)
printf('Drift velocity of electrons= %0.3f    ms**-1 \n',De)
```

Scilab code Exa 6.17 `critical field`

```
//Variable declaration
H0=64*10**3;     //initial field(ampere/m)
T=5;     //temperature(K)
Tc=7.26;    //transition temperature(K)

//Calculation
H=H0*(1-(T/Tc)**2);      //critical field(ampere/m)

//Result
printf('critical field is %0.3f     *10**3 ampere/m \n',(H/10**3))
```

Scilab code Exa 6.18 `Frequency of generated microwaves`

```
//Variable declaration
e=1.6*10**-19
V=1*10
h=6.625*10**-34

//Calculations
v=(2*e*V**-3)/h

//Result
printf('Frequency of generated microwaves= %0.3f*10**9 Hz      \n',(v/10**9))
```

Scilab code Exa 6.19 Penetration depth

```
//Variable declaration
d=7300                        //density in (kg/m**3)
N=6.02*10**26                 //Avagadro Number
A=118.7                       //Atomic Weight
E=1.9                       //Effective mass
e=1.6*10**-19

//Calculations
n=(d*N)/A
m=E*9.1*10**-31
x=4*%pi*10**-7*n*e**2
lamda_L=sqrt(m/x)

//Result
printf('Number of electrons per unit volume = %0.3f*10**28/m**3     \n',(n/10**28))
printf('Effective mass of electron m = %0.3f*10*-31 kg    \n',(m*10**31))
printf('Penetration depth = %0.3f    Angstroms   \n',lamda_L*10**8)
printf('//The answer given in the text book is wrong
')
```

Scilab code Exa 6.20 Calculate wavelength

```

//Variable declaration
lamda_L1=39.6*10**-9
lamda_L2=173*10**-9
T1=7.1
T2=3

//Calculations
x=(lamda_L1/lamda_L2)**2
Tc4=(T1**4)-((T2**4)*x)/(1-x)
Tc=(Tc4)**(1/4)
printf('Tc = %0.3f K \n',(Tc))
printf('lamda0= %0.3f nm \n',((sqrt(1-(T2/Tc)**4)*lamda_L1)*10**9))
```

Scilab code Exa 6.21 Critical current density

```

//Variable declaration
H0=6.5*10**4                //(ampere/metre)
T=4.2                       //K
Tc=7.18                     //K
r=0.5*10**-3

//Calculations
Hc=H0*(1-(T/Tc)**2)
Ic=(2*%pi*r)*Hc
A=%pi*r**2
Jc=Ic/A                     //Critical current density
```

```

//Result
printf('Hc = %0.3f         *10**4 \n',(Hc/10**4))
printf('Critical current density,Jc = %0.3f  *10**8 ampere/metre**2        \n',(Jc/10**8))
```

Scilab code Exa 6.22 New critical temperature for mercury

```

//Variable declaration
Tc1=4.185
M1=199.5
M2=203.4

//Calculations
Tc2=Tc1*(M1/M2)**(1/2)

//Result
printf('New critical temperature for mercury = %0.3f K    \n',(Tc2))
```

Chapter 7

Lasers

Scilab code Exa 7.1 Calcutate Divergence

```

//variable declaration
r1 = 7;                    //in radians
r2 = 3;                    //in radians
d1 = 4;                    //Converting from mm to radians
d2 = 6;                    //Converting from mm to radians

//calculations
D = (r2-r1)/(d2*10**3-d1*10**3)    //Divergence

//Result
printf('Divergence = %0.3f    *10**-3 radian      \n',(D*10**3))
```

Scilab code Exa 7.2 Relative Population

```

//variable declaration
C=3*10**8                      //The speed of light
Lamda=6943                     //Wavelength
T=300                          //Temperature in Kelvin
h=6.626*10**-34                //Planck constant
k=1.38*10**-23                 //Boltzmann's constant

//Calculations

V=(C)/(Lamda*10**-10)         //Frequency
R=exp(h*V/(k*T))         //Relative population

//Result
printf('Frequency (V) = %0.3f  *10**14 Hz      \n',(V/10**14))
printf('Relative Population= %0.3f    *10**30    \n',(R/10**30))
```

Scilab code Exa 7.3 Power density

```

//variable declaration
C=3*10**8                       //Velocity of light
W=632.8*10**-9                  //wavelength
P=2.3
t=1
h=6.626*10**-34                //Planck constant
S=1*10**-6

//Calculations
V=C/W                          //Frequency
n=((P*10**-3)*t)/(h*V)         //no.of photons emitted
PD=P*10**-3/S                  //Power density

```

```
//Result
printf('Frequency= %0.3f    *10**14 Hz    \n',(V/10**14))
printf('no.of photons emitted= %0.3f    *10**15 photons/sec    \n',(n/10**15))
printf('Power density = %0.3f   kWm**-2     \n',(PD/1000))
```

Scilab code Exa 7.4 `Wavelenght`

```

//variable declaration
h=6.626*10**-34                    //Planck constant
C=3*10**8                          //Velocity of light
E_g=1.44                           //bandgap

//calculations
lamda=(h*C)*10**10/(E_g*1.6*10**-19)        //Wavelenght

//Result
printf('Wavelenght = %0.3f   Angstrom     \n',(lamda))
```

Scilab code Exa 7.5 `Band gap`

```

//variable declaration
W=1.55                             //wavelength

//Calculations
E_g=(1.24)/W                       //Bandgap in eV

```

```
//Result
printf('Band gap = %0.3f   eV \n',E_g)
```

Chapter 8

Fiber Optics

Scilab code Exa 1.1 Critical angle

```
//variable declaration
n1=1.50             //Core refractive index
n2=1.47             //Cladding refractive index

//Calculations
C_a=asin(n2/n1)          //Critical angle
N_a=(n1**2-n2**2)**(1/2)
A_a=asin(N_a)

//Results
printf('The Critical angle =%0.3f degrees\n',(C_a
   *180/%pi))
printf('The numerical aperture =%0.3f \n',(N_a))
printf('The acceptance angle =%0.3f degrees\n',(A_a
   *180/%pi))
```

Scilab code Exa 8.2 No of modes propogated inside the fiber

```
//variable declaration
d=50                //diameter
N_a=0.2             //Numerical aperture
lamda=1             //wavelength

//Calculations
N=4.9*(((d*10**-6*N_a)/(lamda*10**-6))**2)

//Result
printf('N =%0.3f \n',N)
printf('Fiber can support%0.3f guided modes \n',N)
printf('In graded index fiber, No.of modes propogated inside the fiber =%0.3f only',N/2)
```

Scilab code Exa 8.3 Numerical aperture

```
//variable declaration
d=50                //diameter
n1=1.450
n2=1.447
lamda=1             //wavelength

//Calculations
N_a=(n1**2-n2**2)   //Numerical aperture
N=4.9*(((d*10**-6*N_a)/(lamda*10**-6))**2)

//Results
printf('Numerical aperture =%0.3f ',N_a)
printf('No. of modes that can be propogated =%0.3f \n',(N))
```

Scilab code Exa 8.4 Numerical aperture

```
//variable declaration
delta=0.05
n1=1.46

//Calculation
N_a=n1*(2*delta)**(1/2)      //Numerical aperture

//Result
printf('Numerical aperture =%0.3f \n',(N_a))
```

Scilab code Exa 8.5 `maximum no of modes propogating through fiber`

```
//variable declaration
a=5
n1=1.450
n2=1.447
lamda=1              //wavelength

//Calculations
N_a=(n1**2-n2**2)    //Numerical aperture

N=4.9*((a*10**-6*sqrt(N_a)/(lamda*10**-6))**2)

//Result

printf('maximum no.of modes propogating through fiber =%0.3f \n',(N))
printf('Correction needed')
```

Scilab code Exa 8.6 `Number of modes`

```
//variable declaration
a=100
```

```
N_a=0.3                      //Numerical aperture
lamda=850                    //wavelength

//Calculations
V_n=(2*(%pi**2*a**2*10**-12*N_a**2)/lamda**2*10**-18)
//Result
printf('Number of modes =%0.3f modes\n',(V_n/10**-36))
printf('No.of modes is doubled to account for the two possible polarisations')
printf('Total No.of modes =%0.3f \n',(V_n/10**-36)*2)
```

Scilab code Exa 8.7 Cutoff Wavellength

```
//variable declaration
a=5;
n1=1.48;
delta=0.01;
V=25;

//Calculation
lamda=(%pi*(a*10**-6)*n1*sqrt(2*delta))/V    //Cutoff Wavelength

//Result
printf('Cutoff Wavellength =%0.3f micro m. \n',(lamda*10**7))
```

Scilab code Exa 8.8 Maximum core radius

```
//variable declaration
```

```
V=2.405
lamda=1.3
N_a=0.05

//Calculations
a_max=(V*lamda)/(2*%pi*N_a)

//Result
printf('Maximum core radius=%0.3f micro m\n',(a_max)
    )
```

Scilab code Exa 8.9 Acceptance angle

```
//variable declaration
N_a=0.3
gamma=45

//Calculations
theta_a=asin(N_a)
theta_as=asin((N_a)/cos(gamma))

//Results
printf('Acceptance angle, theta_a =%0.3f degrees\n'
    ,(theta_a*180/%pi))
printf('For skew rays,theta_as %0.3f degrees\n',(
    theta_as*180/%pi))
printf('//Answer given in the textbook is wrong')
```

Scilab code Exa 8.10 Numerical aperture and Acceptance angle

```
//variable declaration
n1=1.53
delta=0.0196
```

```

//Calculations
N_a=n1*(2*delta)**(1/2)
A_a=asin(N_a)
//Result
printf('Numerical aperture =%0.3f \n',(N_a))
printf('Acceptance angle =%0.3f degrees \n',(A_a*180/%pi))
```

Scilab code Exa 8.11 Core radius

```
//variable declaration
n1=1.480
n2=1.465
V=2.405
lamda=850*10**-9

//Calculations
delta=(n1**2-n2**2)/(2*n1**2)
a=(V*lamda*10**-9)/(2*%pi*n1*sqrt(2*delta))

//Results
printf('delta =%0.3f \n',(delta))
printf('Core radius,a =%0.3f micro m\n',(a*10**15))
```

Scilab code Exa 8.12 Total dist travelled by light over one metre of fiber

```
//variable declaration
n1=1.5
n2=1.49
a=25

//Calculations
```

```
C_a=asin(n2/n1)                //Critical angle
L=2*a*tan(C_a)
N_r=10**6/L

//Result
printf('Critical angle=%0.3f degrees\n',(C_a*180/%pi))
printf('Fiber length covered in one reflection=%0.3f micro m\n',(L))
printf('Total no.of reflections per metre=%0.3f \n',(N_r))
printf('Since L=1m, Total dist. travelled by light over one metre of fiber =%0.3f m \n',(1/sin(C_a)))
```

Scilab code Exa 8.13 No of modes

```
//variable declaration
alpha=1.85
lamda=1.3*10**-6
a=25*10**-6
N_a=0.21

//Calculations
V_n=((2*%pi**2)*a**2*N_a**2)/lamda**2
N_m=(alpha/(alpha+2))*V_n

printf('No.of modes =%0.3f =155(approx)\n',(N_m))
printf('Taking the two possible polarizations , Total No.of nodes =%0.3f \n',(N_m*2))
```

Scilab code Exa 8.14 ignal attention per unit length

```
//variable declaration
P_i=100
P_o=2
L=10

//Calculations
S=(10/L)*log(P_i/P_o)
O=S*L

//Result
printf('a.Signal attention per unit length =%0.3f dB km**-1\n',(S))
printf('b.Overall signal attenuation =%0.3f dB\n',(O))
printf('//Answer given in the textbook is wrong')
```

Scilab code Exa 8.15 Bandwidth length product

```
//variable declaration
L=10
n1=1.55
delta=0.026
C=3*10**5

//Calculations
delta_T=(L*n1*delta)/C
B_W=10/(2*delta_T)

//Result
printf('Total dispersion =%0.3f ns\n',(delta_T/10**-9))
printf('Bandwidth length product =%0.3f Hz-km\n',(B_W/10**5))
printf('//Answer given in the text book is wrong”)
```