

Scilab Textbook Companion for
Introduction to Nuclear Engineering
by J. R. Lamarsh and A. J. Baratta¹

Created by
Dhaivat Udayan Mandavia
NUCLEAR SCIENCE AND ENGINEERING
Physics
DELHI TECHNOLOGICAL UNIVERSITY
College Teacher
None
Cross-Checked by
Spandana

June 6, 2015

¹Funded 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: Introduction to Nuclear Engineering

Author: J. R. Lamarsh and A. J. Baratta

Publisher: Prentice Hall, New Jersey

Edition: 3

Year: 2001

ISBN: 0-201-82498-1

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes	4
2 Atomic and Nuclear Physics	9
3 Interaction of Radiation with Matter	24
4 Nuclear Reactors and Nuclear Power	38
5 Neutron Diffusion and Moderation	45
6 Neutron Reactor Theory	50
7 The Time Dependent Reactor	69
8 Heat Removal from Nuclear Reactors	83
9 Radiation Protection	99
10 Radiation Shielding	117
11 Reactor Licensing Safety and the Environment	137

List of Scilab Codes

Exa 2.1	Number of deuterium atoms	9
Exa 2.2	Atomic and Molecular weight	9
Exa 2.3	Mass and Energy	10
Exa 2.4	Mass and Energy	11
Exa 2.5	Excited states and radiation	12
Exa 2.6	Activity of radioactive foil	12
Exa 2.7	Nuclear Reaction	13
Exa 2.8	Q value of nuclear reaction	14
Exa 2.9	Binding energy	15
Exa 2.10	Mass equation	16
Exa 2.11	Energy of Maxwellian distribution	17
Exa 2.12	Atom density	18
Exa 2.13	Atom density	18
Exa 2.14	Atom density	19
Exa 2.15	Atom density	21
Exa 2.16	Atom density	22
Exa 3.1	Neutron collision	24
Exa 3.2	Probability of nuclear reaction	25
Exa 3.3	Neutron collision	25
Exa 3.4	Mean free path	26
Exa 3.5	Absorption cross section	27
Exa 3.6	Scattering cross section	28
Exa 3.7	Reactor power	28
Exa 3.8	Elastic scattering	29
Exa 3.9	Radiative capture reaction	30
Exa 3.10	Energy loss in scattering reactions	30
Exa 3.11	Neutron absorption rate	31
Exa 3.12	Activity estimation	32

Exa 3.13	Fission neutron estimation	33
Exa 3.14	Energy released in fission reaction	34
Exa 3.15	Gamma ray attenuation in materials	35
Exa 3.16	Energy deposition of radioactive samples	36
Exa 3.17	Range of charged particles	37
Exa 4.1	Conversion and Breeding of nuclear fuels	38
Exa 4.2	Production and doubling time of nuclear fuels	39
Exa 4.3	Nuclear fuel performance	40
Exa 4.4	Plant availability factor and capacity factor	41
Exa 4.5	Nuclear fuel utilization	42
Exa 4.6	Separative work	43
Exa 5.2	Neutron diffusion	45
Exa 5.5	Multigroup diffusion theory	46
Exa 5.6	Neutron diffusion	47
Exa 5.7	Neutron diffusion	48
Exa 6.1	Critical neutron parameters	50
Exa 6.2	Maximum and average flux	51
Exa 6.3	Critical radius	52
Exa 6.4	Non leakage probability	53
Exa 6.5	Critical parameter calculations	54
Exa 6.6	Critical mass	56
Exa 6.7	Critical concentration	59
Exa 6.8	Critical radius	60
Exa 6.9	Critical radius and mass	62
Exa 6.10	Critical parameters for heterogenous reactor	63
Exa 6.11	Critical parameter for heterogenous reactor	65
Exa 6.12	Critical parameter for heterogenous reactor	66
Exa 6.13	Critical parameter for heterogenous reactor	67
Exa 7.1	Reactor kinetics	69
Exa 7.2	Reactor period	70
Exa 7.3	Reactivity	70
Exa 7.4	Reactor period	71
Exa 7.5	Reactivity	71
Exa 7.6	Prompt drop	72
Exa 7.7	Control worth of a center rod	73
Exa 7.8	Total control rod worth	74
Exa 7.9	Total control rod worth	75
Exa 7.10	Total control rod worth	77

Exa 7.11	Differential control rod worth	78
Exa 7.12	Chemical Shim	79
Exa 7.13	Temperature coefficient of reactivity	80
Exa 7.14	Fission product poisons	81
Exa 8.1	Coolant temperature	83
Exa 8.2	Steam temperature	84
Exa 8.3	Heat production in fuel rods	85
Exa 8.4	Decay energy	86
Exa 8.5	Fuel rod temperature parameters	89
Exa 8.6	Coolant temperature	90
Exa 8.7	Fuel rod temperature parameters and coolant temperature	91
Exa 8.8	Reynolds number	93
Exa 8.9	Heat transfer coefficient	94
Exa 8.10	Boiling crisis	95
Exa 8.11	Engineering sub factor	97
Exa 8.12	Fuel rod calculation	97
Exa 9.1	Gamma ray interaction	99
Exa 9.2	Absorbed dose rate and dose equivalent rate	100
Exa 9.3	Acute radiation exposure	100
Exa 9.4	Acute radiation exposure	103
Exa 9.5	Chronic radiation exposure	107
Exa 9.6	External exposure due to gamma rays	108
Exa 9.7	External exposure due to X rays	108
Exa 9.8	External exposure due to neutrons	109
Exa 9.9	External exposure due to neutrons	110
Exa 9.10	Internal exposure due to radionuclide	111
Exa 9.11	Maximum Permissible Concentration	113
Exa 9.12	Acute radiation exposure	114
Exa 9.13	Acute radiation exposure	115
Exa 10.1	Gamma ray buildup factor	117
Exa 10.2	Gamma ray buildup factor	118
Exa 10.3	Shielding of infinite planar source	121
Exa 10.4	Multilayered shielding	124
Exa 10.5	Removal cross section	126
Exa 10.6	Removal cross section	128
Exa 10.7	Dose equivalent rate and thickness	129
Exa 10.8	Shielding of prompt fission gamma rays	130

Exa 10.9	Coolant activity	134
Exa 11.1	Diffusion of radioactive effluents	137
Exa 11.2	External dose from gamma rays	138
Exa 11.3	External dose from gamma rays	139
Exa 11.4	External dose from gamma rays	139
Exa 11.5	Dose rate to thyroid	140
Exa 11.6	Ground exposure from gamma rays	142
Exa 11.7	External dose from gamma rays	143
Exa 11.8	Direct dose from gamma rays	145
Exa 11.9	Activity of radionuclide release into atmosphere	147
Exa 11.10	Concentration of radionuclide	148
Exa 11.11	Activity of radioactive gas effluent	150
Exa 11.12	Activity of radioactive liquid effluent	151

List of Figures

10.1 Gamma ray buildup factor	119
10.2 Shielding of infinite planar source	122

Chapter 2

Atomic and Nuclear Physics

Scilab code Exa 2.1 Number of deuterium atoms

```
1 // Example 2.1
2 clear all;
3 clc;
4 // Given data
5 atom_h = 6.6*10^24;           // Number
    of atoms in Hydrogen
6 // Using the data given in Table II.2, Appendix II
    for isotropic abundance of deuterium
7 isoab_H2 = 0.015;           //
    Isotropic abundance of deuterium
8 // Calculation
9 totatom_d=(isoab_H2*atom_h)/100;
10 // Result
11 printf('\n Number of deuterium atoms = %2.1E \n',
    totatom_d);
```

Scilab code Exa 2.2 Atomic and Molecular weight

```

1 // Example 2.2
2 clear all;
3 clc;
4 // Given data
5 // Using the data given in the example 2.2
6 atwt_016 = 15.99492; // Atomic
   weight of O-16 isotope
7 isoab_016 = 99.759; //
   Abundance of O-16 isotope
8 atwt_017 = 16.99913; // Atomic
   weight of O-17 isotope
9 isoab_017 = 0.037; //
   Abundance of O-17 isotope
10 atwt_018 = 17.99916; // Atomic
   weight of O-18 isotope
11 isoab_018 = 0.204; //
   Abundance of O-18 isotope
12 // Calculation
13 atwt_0=(isoab_016*atwt_016 + isoab_017*atwt_017 +
   isoab_018*atwt_018)/100;
14 // Result
15 printf('\n Atomic Weight of Oxygen = %5.5f \n',
   atwt_0);

```

Scilab code Exa 2.3 Mass and Energy

```

1 // Example 2.3
2 clear all;
3 clc;
4 // Given data
5 me = 9.1095*10^(-28); // Mass of
   electron in grams
6 c = 2.9979*10^10; // Speed
   of light in vacuum in cm/sec
7 // Calculation

```

```

8 rest_mass = me*c^2;
9 // Result
10 printf('\n Rest mass energy of electron = %5.4E ergs
    \n',rest_mass);
11 disp('Expressing the result in joules')
12 // 1 Joule = 10^(-7)ergs
13 rest_mass_j = rest_mass*10^(-7);
14 printf('\n Rest mass energy of electron = %5.4E
    joules\n',rest_mass_j);
15 disp('Expressing the result in MeV')
16 // 1 MeV = 1.6022*10^(-13)joules
17 rest_mass_mev = rest_mass_j/(1.6022*10^(-13));
18 printf('\n Rest mass energy of electron = %5.4f MeV\
    n',rest_mass_mev);

```

Scilab code Exa 2.4 Mass and Energy

```

1 // Example 2.4
2 clear all;
3 clc;
4
5 // From the result of Example 2.3
6 // Rest mass energy of electron = 0.5110 MeV
7 rest_mass_mev = 0.5110;
8 me = 9.1095*10^(-28); //
    Mass of electron in grams
9 // From standard data table
10 // 1 amu = 1.6606*10^(-24)g
11 amu = 1.6606*10^(-24);
12 // Calculation
13 en_eq = (amu/me)*rest_mass_mev;
14 // Result
15 printf('\n Energy equivalent of one amu = %3.1f MeV\
    n',en_eq);

```

Scilab code Exa 2.5 Excited states and radiation

```
1 // Example 2.5
2 clear all;
3 clc;
4
5 // From the standard data table
6 h = 6.626*10^(-34); //
   Planck's constant in J-s
7 c = 3*10^8; // Speed
   of light in vacuum in m/sec
8 // Given data
9 disp('The ionization energy of K shell electron in
   Lead atom is 88keV');
10 E = 88*10^3; //
   Ionization energy in keV
11 // Expressing the result in joules by using 1 eV =
   1.6022*10^(-19) J
12 E = E*1.6022*10^(-19);
13 printf("From Planck\'s law of photoelectric effect
   \n Energy = (h*c)/lambda\n");
14 // Calculation
15 lambda = (h*c)/E;
16 // Result
17 printf('\n Wavelength of radiation = %4.3E m\n',
   lambda);
```

Scilab code Exa 2.6 Activity of radioactive foil

```
1 // Example 2.6
2 clear all;
3 clc;
```

```

4
5 // Given data
6 T12 = 64.8;
                                                    // Half
    life = 64.8 hour
7 lambda = 0.693/T12;
                                                    // Decay
    constant in hour(-1)
8 t = 12;
                                                    //
    Analysis time of gold sample in hours
9 alpha = 0.9;
                                                    //
    Activity of gold sample after analysis time
10
11 // 1.
12 // Calculation
13 R = alpha/(1-exp(-lambda*t));
14 // Result
15 printf('\n Theoretical maximum activity = %3.1f
    curie (Ci) \n',R);
16
17 // 2.
18 // Calculation
19 // The expression to calculate 80 percent of maximum
    activity is \n 0.8R = R*(1-exp(-lambda*t))
20 t = -log(0.2)/lambda;
21 // Result
22 printf('\n Time to reach 80 percent of maximum
    activity = %d hours \n',t);

```

Scilab code Exa 2.7 Nuclear Reaction

```

1 // Example 2.7
2 clear all;

```

```

3  clc;
4
5  disp('The reactants are Nitrogen and neutron')
6  // The total atomic number of reactants
7  Z_reactant = 7+0;
8  // The total atomic mass number of reactants
9  A_reactant = 14+1;
10 disp('One of the known product is Hydrogen')
11 Z_H = 1; // The atomic
    number of Hydrogen
12 A_H = 1; // The atomic
    mass number of Hydrogen
13 // The atomic number of unknown element
14 Z_unknown = Z_reactant-Z_H;
15 // The atomic mass number of unknown element
16 A_unknown = A_reactant-A_H;
17 // Result
18 printf("\n For unknown element the atomic number is
    %d and atomic mass number is %d \n",Z_unknown,
    A_unknown);
19 // From periodic table
20 disp('The element corresponds to Carbon-14');

```

Scilab code Exa 2.8 Q value of nuclear reaction

```

1  // Example 2.8
2  clear all;
3  clc;
4
5  disp('The reaction is Tritium(d,n)Helium-4');
6  // Using standard data table of mass in amu
7  M_H3 = 3.016049; //
    Atomic mass of Tritium
8  M_He4 = 4.002604; //
    Atomic mass of Helium

```

```

9 M_d = 2.014102; //
   Atomic mass of Deuterium
10 M_n = 1.008665; //
   Atomic mass of neutron
11 // Calculation of total mass of reactants
12 tot_reac = M_H3+M_d;
13 // Calculation of total mass of products
14 tot_prod = M_He4+M_n;
15 // Calculation
16 Q = tot_reac-tot_prod;
17 // Expressing in MeV by using 1 amu = 931.5 MeV
18 Q_mev = Q*931.5;
19 // Result
20 printf(" \n Q value for the reaction = %5.3f MeV",
   Q_mev);
21 if Q_mev > 0 then
22     printf("\n The reaction is exothermic. \n");
23 else
24     printf("\n The reaction is endothermic. \n");
25 end

```

Scilab code Exa 2.9 Binding energy

```

1 // Example 2.9
2 clear all;
3 clc;
4
5 // Using standard data table of mass in amu
6 M_C12 = 12; // Atomic
   mass of Carbon-12
7 M_n = 1.00866; // Atomic
   mass of neutron
8 M_C13 = 13.00335; // Atomic
   mass of Carbon-13
9 //If one neutron is removed from carbon-13, carbon

```



```

    -12 is obtained
10 tot = M_C12+M_n;
11 dm = tot-M_C13;           // Mass
    defect
12 // Converting to energy equivalent from mass by
    using 1 amu = 931.5 MeV
13 Es = dm*931.5;
14 // Result
15 printf("\n The binding Energy of the last neutron
    in Carbon-13 atom = %4.2 f MeV",Es);

```

Scilab code Exa 2.10 Mass equation

```

1 // Example 2.10
2 clear all;
3 clc;
4
5 // Using standard data table of mass and the
    coefficients of mass equation for Silver-107
6 N = 60;           //
    Number of neutrons
7 Z = 47;           //
    Atomic number
8 A = 107;          //
    Atomic mass number
9 // The coefficients used in mass equation are
10 alpha = 15.56;
11 bet = 17.23;
12 gam = 0.697;
13 zeta = 23.285;
14 mn = 939.573;    //
    Mass of neutron in terms of energy
15 mH = 938.791;    //
    Mass of proton in terms of energy
16 // Calculation

```

```

17 disp('Using mass equation ');
18 M = (N*mn)+(Z*mH)-(alpha*A)+(bet*(A^(2/3)))+(gam*Z
    ^2/A^(1/3))+(zeta*(A-2*Z)^2/A);
19 // Expressing in amu by using 1 amu = 931.5 MeV
20 M_amu = M/931.5;
21 printf(" Mass = %5.5f MeV = %5.5f u \n",M,M_amu);
22 disp('Actual mass = 106.905092 u');
23 // Calculation
24 BE = (alpha*A)-(bet*(A^(2/3)))-(gam*Z^2/A^(1/3))-((
    zeta*(A-(2*Z))^2)/A);
25 // Result
26 printf("\n Binding Energy = %4.2f MeV or %3.1f MeV/
    nucleon \n",BE,BE/107);
27 // The value is different from the answer given in
    the textbook. The textbook answer is wrong.

```

Scilab code Exa 2.11 Energy of Maxwellian distribution

```

1 // Example 2.11
2 clear all;
3 clc;
4
5 // Given data
6 T_C = 38; // Given
    temeperature in celsius
7 //The temperature in Kelvin
8 T_K = T_C+273.15;
9 T_0 = 293.61; // The
    temperature in kelvin equivalent to 0 deg celsius
10 kT = 0.0253; // The term 'kT'
    in eV at temperature T0
11 // Calculation
12 Ep = 0.5*kT*(T_K/T_0);
13 Ebar = 3*Ep;
14 // Result

```

```
15 printf(" Most probable energy of air molecules = %5
    .5f eV \n",Ep);
16 printf(" Average energy of air molecules = %5.5f eV
    \n",Ebar);
```

Scilab code Exa 2.12 Atom density

```
1 // Example 2.12
2 clear all;
3 clc;
4
5 // Given data
6 rho = 0.97; //
    Density of Sodium in gram/cm^3
7 // From standard data table
8 NA = 0.6022*10^24; //
    Avagadro number
9 M = 22.99; //
    Atomic weight of Sodium
10 // Calculation
11 N = rho*NA/M;
12 // Result
13 printf("Atom density of sodium = %5.5E atoms/cm^3 \n
    ",N);
```

Scilab code Exa 2.13 Atom density

```
1 // Example 2.13
2 clear all;
3 clc;
4
5 // Given data
```

```

6 rho_NaCl = 2.17; // Density of
   Sodium Chloride(NaCl) in gram/cm^3
7 // From standard data table
8 NA = 0.6022*10^24; // Avogadro
   number
9 M_Na = 22.99; // Atomic weight
   of Sodium(Na)
10 M_Cl = 35.453; // Atomic weight
   of Chlorine(Cl)
11 M_NaCl = M_Na+M_Cl; // Molecular
   weight of Sodium Chloride(NaCl)
12 // Calculation
13 N = rho_NaCl*NA/M_NaCl;
14 // As in NaCl, there is one atom of Na and Cl
15 N_Na = N;
16 N_Cl = N;
17 // Result
18 printf(" Atom density of Sodium and Chlorine = %5.4E
   molecules/cm^3 \n",N);

```

Scilab code Exa 2.14 Atom density

```

1 // Example 2.14
2 clear all;
3 clc;
4
5 // Given data
6 rho = 1; //
   Density of water in gram/cm^3
7
8 // 1.
9 M_H = 1.00797; //
   Atomic weight of Hydrogen(H)
10 M_O = 15.9994; //
   Atomic weight of Oxygen(O)

```

```

11 // As in water , there is two atoms of Hydrogen(H)
    and one atom of Oxygen(O)
12 M = (2*M_H)+M_O; //
    Molecular weight of water
13 // From standard data table
14 NA = 0.6022*10^24; //
    Avagadro number
15 // Calculation
16 N = rho*NA/M;
17 // Result
18 printf("Atom density of water = %5.5E molecules/cm^3
    \n",N);
19
20 // 2.
21 // As in water , there is two atoms of Hydrogen(H)
    and one atom of Oxygen(O)
22 N_H = 2*N; //
    Atom density of Hydrogen
23 N_O = N; //
    Atom density of Oxygen
24 // Result
25 printf("Atom density of Hydrogen(H) = %5.4E atoms/cm
    ^3 \n",N_H);
26 printf("Atom density of Oxygen(O)= %5.4E atoms/cm^3
    \n",N_O);
27
28 // 3.
29 // Using the data given in Table II.2, Appendix II
    for isotropic abundance of deuterium
30 isoab_H2 = 0.015;
31 // Calculation
32 N_H2 = isoab_H2*N_H/100;
33 // Result
34 printf("Atom density of Deuterium(H-2)= %5.4E atoms/
    cm^3 \n",N_H2);

```

Scilab code Exa 2.15 Atom density

```
1 // Example 2.15
2 clear all;
3 clc;
4
5 // Given data
6 rho = 19.1; //
   Density of Uranium-235 in gram/cm^3
7 wt = 1500; //
   Weight of uranium rods in a reactor in kg
8 nr = 0.2; //
   Enrichment(w/o) of Uranium-235
9
10 // 1.
11 // As Enrichment is 20(w/o)
12 wt_U235 = nr*wt; //
   Amount of Uranium-235
13 // Result
14 printf("Amount of Uranium-235 in the reactor = %d kg
   \n",wt_U235);
15
16 // 2.
17 // From standard data table
18 NA = 0.6022*10^24; //
   Avagadro number
19 M_U235 = 235.0439; //
   Atomic weight of Uranium-235
20 M_U238 = 238.0508; //
   Atomic weight of Uranium-238
21 // Calculation
22 N_U235 = nr*rho*NA/M_U235; // Atom
   density of Uranium-235
23 N_U238 = (1-nr)*rho*NA/M_U238; // Atom
```

```

    density of Uranium-238
24 // Result
25 printf("Atom density of Uranium-235 = %5.2E atoms/cm
    ^3 \n",N_U235);
26 printf("Atom density of Uranium-238 = %5.2E atoms/cm
    ^3 \n",N_U238);

```

Scilab code Exa 2.16 Atom density

```

1 // Example 2.16
2 clear all;
3 clc;
4
5 // Given data
6 rho_UO2 = 10.5; //
    Density of UO2 pellets in gram/cm^3
7 nr = 0.3; //
    Enrichment(w/o) of Uranium-235
8 // From standard data table
9 M_U235 = 235.0439; //
    Atomic weight of Uranium-235
10 M_U238 = 238.0508; //
    Atomic weight of Uranium-238
11 M_O = 15.999; //
    Atomic weight of Oxygen
12 NA = 0.6022*10^24; //
    Avogadro number
13
14 M = 1/((nr/M_U235)+((1-nr)/M_U238));
15 M_UO2 = M+(2*M_O); //
    Molecular weight of UO2
16 nr_U = M/M_UO2*100; // The
    percent(w/o) of Uranium in UO2 pellet
17 rho_U = nr_U*rho_UO2/100 //
    Density of Uranium in g/cm^3

```

```
18 rho_U235 = nr*rho_U //
    Density of Uranium-235 in g/cm^3
19 // Calculation
20 N_U235=rho_U235*NA/M_U235;
21 // Result
22 printf("Atom density of Uranium-235 = %5.2E atoms/cm
    ^3 \n",N_U235);
```

Chapter 3

Interaction of Radiation with Matter

Scilab code Exa 3.1 Neutron collision

```
1 // Example 3.1
2 clear all;
3 clc;
4
5 // Given data
6 // 1 barn = 10(-24) cm2
7 sigma = 2.6*10(-24); // Cross
   section of carbon-12 in cm2
8 I = 5*108; // Intensity
   of neutron beam in neutrons/cm2-sec
9 A = 0.1; // Cross
   sectional area of the beam in cm2;
10 X = 0.05; // Thickness
   of the target in cm
11
12 // 1.
13 // Using the data given in Table I.3, Appendix II
   for carbon-12
14 N = 0.08*10(24); // Atom
```

```

    density in atoms/cm^3
15 // Calculation
16 IR = sigma*I*N*A*X;
17 // Result
18 printf('\n Total interaction rate = %2.1E
    interactions/sec \n',IR);
19
20 // 2.
21 no = I*A; // Neutron
    rate in neutrons/sec
22 // Calculation
23 p = IR/no;
24 printf('\n Probability of collision = %3.2E \n',p);

```

Scilab code Exa 3.2 Probability of nuclear reaction

```

1 // Example 3.2
2 clear all;
3 clc;
4
5 // Given data
6 sigmaf = 582; // Fission cross section of
    U-235 on bombardment of neutron in barn
7 sigmay = 99; // Radiative capture cross
    section of U-235 on bombardment of neutron in
    barn
8 // Calculation
9 pf = sigmaf/(sigmaf+sigmay);
10 // Result
11 printf('\n Probability of fission = %.3f = %3.1f
    percent\n',pf,pf*100);

```

Scilab code Exa 3.3 Neutron collision

```

1 // Example 3.3
2 clear all;
3 clc;
4
5 // Given data
6 // Using the data given in the example 3.1
7 N = 0.08*10^(24); // Atom density
   of Carbon-12 in atoms/cm^3
8 // 1 barn = 10^(-24) cm^2
9 sigma = 2.6*10^(-24); // Cross section
   of carbon-12 in cm^2
10 I = 5*10^8; // Intensity of
   neutron beam in neutrons/cm^2-sec
11
12 // 1.
13 // Calculation
14 SIGMA_t = N*sigma;
15 // Result
16 printf('\n Macroscopic cross section of carbon-12 =
   %3.2f cm^(-1)\n',SIGMA_t);
17
18 //2.
19 // Calculation
20 F= I*SIGMA_t;
21 // Result
22 printf('\n Collision density in the carbon-12 target
   = %3.2E collisions/cm^(3)-sec\n',F);

```

Scilab code Exa 3.4 Mean free path

```

1 // Example 3.4
2 clear all;
3 clc;
4
5 // Given data

```

```

6 E = 100; // Neutron
  energy in keV
7 // Using the data given in Table II.3, for E = 100
  keV
8 atom_density = 0.0254*10^(24); // Atom density
  of sodium in atoms/cm^3
9 // 1 barn = 10^(-24) cm^2
10 sigma = 3.4*10^(-24); // Microscopic
  cross section of sodium in cm^2
11 // Calculation
12 SIGMA = atom_density*sigma;
13 lambda = 1/SIGMA;
14 // Result
15 printf('\n Macroscopic cross section = %5.4f cm^(-1)
  \n',SIGMA);
16 printf('\n Mean Free Path = %3.2f cm\n',lambda);

```

Scilab code Exa 3.5 Absorption cross section

```

1 // Example 3.5
2 clear all;
3 clc;
4
5 // Given data
6 atom_density_U235 = 3.48*10^(-4)*10^(24); // Atom
  density of Uranium-235 in atoms/cm^3
7 atom_density_U238 = 0.0483*10^(24); // Atom
  density of Uranium-238 in atoms/cm^3
8 // 1 barn = 10^(-24) cm^2
9 sigmaa_U235 = 680.8*10^(-24); //
  Absorption cross section of Uranium-235 incm^2
10 sigmaa_U238 = 2.7*10^(-24); //
  Absorption cross section of Uranium-238 incm^2
11 // Calculation
12 SIGMAA=(atom_density_U235*sigmaa_U235)+(

```

```

        atom_density_U238*sigmaa_U238);
13 // Result
14 printf('\n Macroscopic absorption cross section = %4
        .3f cm(-1)\n',SIGMAA);

```

Scilab code Exa 3.6 Scattering cross section

```

1 // Example 3.6
2 clear all;
3 clc;
4
5 // Given data
6 sigmas_H_1 = 3; // Scattering cross
    section of Hydrogen in barn at 1 MeV
7 sigmas_O_1 = 8; // Scattering cross
    section of Oxygen in barn at 1 MeV
8 sigmas_H_th = 21; // Scattering cross
    section of Hydrogen in barn at 0.0253 eV
9 sigmas_O_th = 4; // Scattering cross
    section of Oxygen in barn at 0.0253 eV
10 // Calculation
11 sigmas_H2O_1 = (2*sigmas_H_1)+(1*sigmas_O_1);
12 // Result
13 printf('\n Scattering cross section of Water at 1
    MeV = %d b \n',sigmas_H2O_1);
14 // The equation used to calculate the scattering
    cross section at 1 MeV cannot be used at thermal
    energy.
15 printf(' Experimental value of scattering cross
    section of Water at 0.0253 eV = %d b \n',103);

```

Scilab code Exa 3.7 Reactor power

```

1 // Example 3.7
2 clear all;
3 clc;
4
5 // Given data
6 phi = 1*10^(13); //
   Neutron flux in neutrons/cm^3
7 v = 64000; //
   Volume of research reactor in cm^3
8 sigmaf = 0.1; //
   Macroscopic fission cross section in cm^(-1)
9 // The energy released per fission reaction is 200
   MeV
10 // 1 MeV = 1.6*10^(-13) joule
11 E = 200*1.6*10^(-13);
12 // Calculation
13 fiss_rate = sigmaf*phi; //
   Fission rate in neutrons/cm^2-sec
14 power_cc = E*fiss_rate/10^6; //
   Reactor power/cc
15 power = power_cc*v;
16 printf('\n Reactor power of a research reactor = %d
   MW\n',power);

```

Scilab code Exa 3.8 Elastic scattering

```

1 // Example 3.8
2 clear all;
3 clc;
4
5 // 1 barn = 10^(-24) cm^2
6 // From the Figure 3.4 given in the textbook
7 sigmae = 4.8*10^(-24); //
   Experimental cross section of carbon from 0.02eV
   to 0.01MeV

```

```

8 // Assuming spherical shape and elastic scattering
9 R = sqrt(sigmae/(4*pi));
10 // Result
11 printf('\n Radius of carbon nucleus = %3.1E cm\n',R)
    ;

```

Scilab code Exa 3.9 Radiative capture reaction

```

1 // Example 3.9
2 clear all;
3 clc;
4
5 // Given data
6 E0 = 0.0253; //
    Thermal energy in eV
7 // 1 barn = 10(-24) cm2
8 sigmay_E0 = 0.332*10(-24); //
    Radiative capture cross section at 0.0253 eV in
    cm2
9 E = 1; //
    Energy in eV at which radiative cross section is
    to be found
10 // Calculation
11 sigmay_E = sigmay_E0*sqrt(E0/E);
12 // Result
13 // Expressing the result in barn
14 printf('\n Radiative capture cross section of
    hydrogen at 1 eV = %5.4f b\n',sigmay_E*10(24));

```

Scilab code Exa 3.10 Energy loss in scattering reactions

```

1 // Example 3.10
2 clear all;

```

```

3  clc;
4
5  // Given data
6  E = 1;                               // Energy of
    neutron in MeV
7  A = 2;                               // Atomic
    mass number of deuterium
8  v = 45;                              //
    Scattering angle in degree
9
10 // 1.
11 // Calculation
12 E_dash = E/(A+1)^2 * ((cosd (v)+sqrt(A^2-(sind(v))^2)
    )^2);
13 // Result
14 printf('\n Energy of scattered neutron = %4.3f MeV \
    n',E_dash);
15
16 // 2.
17 // Calculation
18 E_A = E-E_dash;
19 // Result
20 printf('\n Energy of recoil nucleus = %4.3f MeV \n',
    E_A);
21
22 // 3.
23 // Calculation
24 deltau = log(E/E_dash);
25 // Result
26 printf('\n Change in lethargy of neutron on
    collision = %4.3f \n',deltai);

```

Scilab code Exa 3.11 Neutron absorption rate

```
1 // Example 3.11
```



```

2 clear all;
3 clc;
4
5 // Given data
6 phi = 5*10^(12); //
   Neutron flux in neutrons/cm^2-sec
7 T = 600; //
   Temperature of neutron in degree
8 // Using the data given in Table II.3, Appendix II
   for indium
9 N = 0.0383*10^(24); //
   Atom density in atoms/cm^3
10 // 1 barn = 10^(-24) cm^2
11 sigmaa_E0 = 194*10^(-24); //
   Microscopic absorption cross section in cm^2
12 SIGMA_E0 = N*sigmaa_E0; //
   Macroscopic absorption cross section in cm^(-1)
13 // From Table 3.2
14 ga_600 = 1.15; //
   Non 1/v factor at 600 degree celsius
15 // Calculation
16 F_a = ga_600*SIGMA_E0*phi;
17 // Result
18 printf('\n Absorption rate of neutrons per cc in
   indium foil = %4.2E neutrons/cm^3-sec \n',F_a);

```

Scilab code Exa 3.12 Activity estimation

```

1 // Example 3.12
2 clear all;
3 clc;
4
5 // Given data
6 N = 120; // Number of
   fuel rods

```

```

7 P = 100;                // Reactor
   power in MW
8 t = 1;                  //
   Estimation time of fuel rod after removal in days
9 T = 365;                // Time of
   reactor operation
10 // Estimation
11 Activity_total = 1.4*10^6*P*[t^(-0.2)-(t+T)^(-0.2)];
12 Activity_one = Activity_total/N;        // For one
   fuel rod
13 // Result
14 printf('\n The activity of a fuel rod = %2.1E Ci \n'
   ,Activity_one);

```

Scilab code Exa 3.13 Fission neutron estimation

```

1 // Example 3.13
2 clear all;
3 clc;
4
5 // Using the data given in Table 3.4 and Table II.2
   for uranium
6 v_235 = 2.418;          // Average number of
   neutrons released per fission
7 y_235 = 0.72;          // Isotropic abundance of
   Uranium-235 on the earth
8 sigmaf_235 = 582.2;    // Fission cross section
   of Uranium-235
9 sigmaa_235 = 680.8;    // Absorption cross
   section of Uranium-235
10 N_235 = y_235;
11 y_238 = 99.26;        // Isotropic abundance of
   Uranium-238 on the earth
12 sigmaa_238 = 2.7;     // Absorption cross
   section of Uranium-238

```

```

13 // Calculation
14 n = (v_235*y_235*sigmaf_235)/((y_235*sigmaa_235)+(
    y_238*sigmaa_238));
15 // Result
16 printf('\n Eta for natural uranium = %3.2f \n',n);

```

Scilab code Exa 3.14 Energy released in fission reaction

```

1 // Example 3.14
2 clear all;
3 clc;
4
5 // Fission of 1 g of Uranium-235 releases
    approximately 1 MW/day of energy.
6 // 1 MW/day = 8.64*10^(10) J
7 energy_uranium = 8.64*10^10;
8
9 // 1. Coal
10 h_coal = 3*10^7; // Heat content of coal in J/
    kg
11 // Calculation
12 amt_coal = energy_uranium/h_coal;
13 // Result
14 printf('\n Amount of coal required for energy
    equivalent of fission = %3.2E kg \n or %3.2f
    metric tons or %3.2f short tons\n',amt_coal,
    amt_coal/10^3,amt_coal*1.10231/10^3);
15 // The result is expressed in all units of
    commercial importance.
16
17 // 2. Oil
18 h_oil = 4.3*10^7; // Heat content of oil in J/
    kg
19 // Calculation
20 amt_oil = energy_uranium/h_oil;

```

```

21 // Result
22 printf('\n Amount of oil required for energy
    equivalent of fission = %3.2E kg \n or %3.2f tons
    or %3.1f barrels\n',amt_oil,amt_oil/10^3,amt_oil
    *6.3/10^3);
23 // The result is expressed in all units of
    commercial importance.

```

Scilab code Exa 3.15 Gamma ray attenuation in materials

```

1 // Example 3.15
2 clear all;
3 clc;
4
5 // Given data
6 rho = 10; // Density
    of UO2 in g/cm^3
7 mol_wt_UO2 = 238+(16*2); // Molecular
    weight of UO2
8 per_U = (238/mol_wt_UO2)*100; // Percent
    by weight of Uranium
9 per_O = 100-per_U; // Percent
    by weight of Oxygen
10
11 // Calculation
12 //Using the data given in Table II.4 for uranium and
    oxygen
13 mup_U = 0.0757; // Ratio of
    mass attenuation coefficient to density of
    uranium in cm^2/g
14 mup_O = 0.0636; // Ratio of
    mass attenuation coefficient to density of oxygen
    in cm^2/g
15 mup = (per_U/100*mup_U)+(per_O/100*mup_O); // The
    total ratio of mass attenuation coefficient in

```

```

    cm2/g
16 mu = mup*rho;
17 // Calculation
18 lambda = 1/mu;
19 // Result
20 printf('\n Mass attenuation coefficient of Uranium
    dioxide (UO2) = %5.3f cm(-1) \n',mu);
21 printf('\n Mean free path = %3.2f cm \n',lambda);
22 // The answer is marked wrongly in the textbook. But
    the solution is correctly evaluated.

```

Scilab code Exa 3.16 Energy deposition of radioactive samples

```

1 // Example 3.16
2 clear all;
3 clc;
4
5 // Given data
6 E = 0.8; // Average gamma ray
    energy in MeV
7 I = 3*10(11); // Intensity of gamma
    rays incident on the container in gamma rays/cm
    2-sec
8 // Using the data given in Table II.5 for iron at
    0.8 MeV
9 mup_iron = 0.0274; // Ratio of mass
    attenuation coefficient to density of iron in cm
    2/g
10 // Calculation
11 dep_rate = E*I*mup_iron;
12 // Expressing the result in SI units
13 // 1 MeV = 1.6*10(-13) J
14 // 1 kg = 1000 g
15 dep_rate_SI = dep_rate*(1.6*10(-13)*1000);
16 printf('\n Rate of energy deposited = %3.2E MeV/g-

```

```
sec or %.2f J/kg-sec \n',dep_rate,dep_rate_SI);
```

Scilab code Exa 3.17 Range of charged particles

```
1 // Example 3.17
2 clear all;
3 clc;
4
5 // Given data
6 E_max = 1.39; // Maximum energy of
   beta rays in MeV
7 // Calculation
8 R_max = 0.412*E_max^(1.265-(0.0954*log(E_max)));
9 // Result
10 printf('\n Maximum distance of beta rays traversed =
   %4.3f cm \n',R_max);
```

Chapter 4

Nuclear Reactors and Nuclear Power

Scilab code Exa 4.1 Conversion and Breeding of nuclear fuels

```
1 // Example 4.1
2 clear all;
3 clc;
4
5 // Given data
6 // Number of neutrons absorbed by Uranium-238 in
   resonances for every neutron absorbed in Uranium
   -235
7 n_resonance = 0.254;
8 // Number of neutrons absorbed by Uranium-238 at
   thermal energy for every neutron absorbed in
   Uranium-235
9 n_th = 0.64;
10 m = 1; // Amount of Uranium-235
   consumed in kg
11 A_U = 235; // Atomic mass number of
   Uranium-235
12 A_Pu = 239; // Atomic mass number of
   Plutonium-239
```

```

13
14 // 1.
15 // Calculation
16 C = n_resonance+n_th;
17 // Result
18 printf('\n Conversion ratio of the reactor = %4.3f \
      n',C);
19
20 // 2.
21 // Calculation
22 amt_Pu = m*C*A_Pu/A_U;
23 // Result
24 printf('\n Amount of Plutonium-239 produced in the
      reactor = %4.3f kg \n',amt_Pu);

```

Scilab code Exa 4.2 Production and doubling time of nuclear fuels

```

1 // Example 4.2
2 clear all;
3 clc;
4
5 // Given data
6 wP0 = 1; // Total fuel consumption
      rate in terms of kg/day
7 M = 500; // Amount of Plutonium-239
      in kg at startup of the reactor
8 breeding_gain = 0.15; // Breeding gain of the
      reactor
9
10 // 1.
11 printf(" The Fast breeder reactor produces %.2f kg
      of plutonium-239 more for every kilogram consumed
      \n",breeding_gain);
12 // Calculation
13 // 1 year = 365 days

```



```

14 production_rate = ceil(breeding_gain*365);
15 // Result
16 printf("\n Production rate of plutonium-239 = %3.2f
    kg/day = %d kg/year",breeding_gain,
    production_rate);
17
18 // 2.
19 // Calculation
20 t_Dl = M/production_rate;
21 t_De = log(2)*t_Dl;
22 // Result
23 printf(" \n Linear doubling time of plutonium fuel
    in the reactor = %2.1f years \n",t_Dl);
24 printf(" \n Exponential doubling time of plutonium
    fuel in the reactor = %2.1f years \n",t_De);

```

Scilab code Exa 4.3 Nuclear fuel performance

```

1 // Example 4.3
2 clear all;
3 clc;
4
5 // Given data
6 power = 3300; // Reactor power in MW
7 time = 750; // Reactor operation time in
    days
8 amt_UO2 = 98; // Amount of Uranium dioxide (
    UO2) in metric tons
9 atwt_U = 238; // As the enrichment of
    Uranium-235 is 3 w/o the majority portion is
    Uranium-238
10 molwt_O = 16; // Molecular weight of Oxygen
11
12
13 // 1.

```

```

14 amt_U = amt_UO2*atwt_U/(atwt_U+2*molwt_O); //
    Amount of uranium in tonne
15 total_burnup = power*time; // Total
    burnup in MWd
16 // Calculation
17 specific_burnup = total_burnup/amt_U;
18 // Result
19 printf(" \n Specific burnup = %3.2f MWd/tonne \n",
    specific_burnup);
20
21 // 2.
22 // Due to fission of 1.05 g of Uranium-235, 1 MWd of
    energy is released.
23 m = 1.05;
24 P = 10^6;
25 maxth_burnup = P/m; //
    Theoretical maximum burnup
26 // Calculation of Fractional burnup
27 bet = specific_burnup/maxth_burnup;
28 // Result
29 printf(" \n Fractional burnup = %3.2f percent \n",
    bet*100);
30 // Due to approximation of specific burnup value,
    there is a slight change in fractional burnup
    value as compared to the textbook value.

```

Scilab code Exa 4.4 Plant availability factor and capacity factor

```

1 // Example 4.4
2 clear all;
3 clc;
4
5 // Given data
6 ratpower = 1075; // Output rated
    electrical power in MWe of the reactor

```

```

7 delpower_yr = 255000;           // Net output power
   delivered in one year in terms of MWd
8 time_refuel = 28;              // Number of days
   the plant was shutdown for refuelling
9 time_repairs = 45;            // Number of days
   the plant was shutdown for repairs
10 time_convrepairs = 18;       // Number of days
   the plant was shutdown for conventional repairs
11
12 // 1.
13 // 1 year = 365 days
14 ratpower_yr = ratpower*365;   // Net output rated
   power in one year in terms of MWd
15 // Calculation
16 cap_factor = delpower_yr/ratpower_yr;
17 // Result
18 printf(" \n Plant capacity factor = %d percent\n",
   ceil(cap_factor*100));
19
20 // 2.
21 // Number of days the plant was shutdown in one year
22 total_shutdown = time_refuel+time_repairs+
   time_convrepairs;
23 // Number of days the plant was operable in one year
24 total_operation = 365-total_shutdown;
25 // Calculation
26 ava_factor = total_operation/365;
27 // Result
28 printf(" \n Plant availability factor = %d percent\n
   ",ava_factor*100);

```

Scilab code Exa 4.5 Nuclear fuel utilization

```

1 // Example 4.5
2 clear all;

```

```

3  clc;
4
5  // Given data
6  t = 30;                               // Time of uranium
    sufficiency in years
7  // Assuming once through Light Water Reactor (LWR)
    fuel cycle
8  U_LWR = 0.0055;                       // Uranium
    Utilization factor for LWR
9  // Assuming once through Liquid Metal cooled Fast
    Breeder Reactor (LMFBR) fuel cycle
10 U_LMFBR = 0.67;                       // Uranium
    Utilization factor for LMFBR
11 // Estimation
12 est_time = 30*U_LMFBR/U_LWR;
13 // Result
14 printf("The time for which Uranium would fuel LMFBR
    = %d years \n",ceil(est_time));

```

Scilab code Exa 4.6 Separative work

```

1  // Example 4.6
2  clear all;
3  clc;
4
5  // Given data
6  A_U = 238;                             // Atomic Mass number of
    Uranium
7  A_O = 16;                              // Atomic Mass number of
    Oxygen
8  amt_UO2 = 33000;                       // Amount of Uranium dioxide
    (UO2) present in kilogram(kg)
9  x_P = 0.032;                           // Enrichment of 3.2 w/o
    uranium product
10 x_T = 0.002;                           // Enrichment of 0.2 w/o

```

```

        residual tails
11 // From Figure 4.45
12 x_F = 0.00711;           // Enrichemnt of 0.711 w/o
        feed
13
14 // 1.
15 // Estimation of enriched uranium in kg
16 M_P = A_U*amt_UO2/(A_U+2*A_0);
17 // Estimation of amount of Uranium feed in kg
18 M_F = ((x_P-x_T)/(x_F-x_T))*M_P;
19 // Result
20 printf("\n The amount of uranium feed required per
        reload = %d kg \n",ceil(M_F));
21
22 // 2.
23 V_x_P = (1-2*x_P)*log((1-x_P)/x_P);           // Value
        function of uranium product with enrichemnt of
        3.2 w/o
24 V_x_F = (1-2*x_F)*log((1-x_F)/x_F);           // Value
        function of feed with enrichemnt of 0.711 w/o
25 V_x_T = (1-2*x_T)*log((1-x_T)/x_T);           // Value
        function of tallings with enrichemnt of 0.2 w/o
26 rate_SWU = 130.75;           //
        Enrichment cost in dollars per SWU
27 // Calculation
28 SWU = M_P*(V_x_P-V_x_T)-M_F*(V_x_F-V_x_T); //
        Separative Work (SWU) in kg
29 enrich_cost = ceil(SWU)*rate_SWU;           //
        Enrichment cost in dollars
30 // Result
31 printf("\n The enrichment cost = $ %d \n",ceil(
        enrich_cost));
32 // Due to approximation of Separative Work Unit (SWU)
        , there is a difference in the value of
        enrichment cost on comparison with the textbook
        value.

```

Chapter 5

Neutron Diffusion and Moderation

Scilab code Exa 5.2 Neutron diffusion

```
1 // Example 5.2
2 clear all;
3 clc;
4
5 // Given data
6 // 1 barn = 10(-24) cm2
7 sigma_s = 4.8*10(-24) // Scattering cross
   section of carbon in cm2
8 A_C = 12; // Atomic Mass
   number for carbon-12
9 E = 1; // Energy of carbon
   -12 atom in eV
10 // Using the data given in Table II.3, for carbon (
   graphite) at energy 1 eV
11 N = 0.08023*10(24); // Atom density in
   terms of atom/cm3
12 mu_bar = 2/(3*A_C); // Average value of
   the cosine of the angle at which neutrons are
   scattered in the medium
```

```

13 SIGMA_s = N*sigma_s           // Macroscopic
    scattering cross section of carbon-12
14 // Calculation
15 D = 1/(3*SIGMA_s*(1-mu_bar));
16 // Result
17 printf('\n Diffusion coefficeint of graphite at 1 eV
    = %4.3f cm \n',D);

```

Scilab code Exa 5.5 Multigroup diffusion theory

```

1 // Example 5.5
2 clear all;
3 clc;
4
5 // Given data
6 phi1 = 6*10^(14);           // Neutron flux of
    Group 1
7 phi2 = 1*10^(15);           // Neutron flux of
    Group 2
8 phi3 = 3*10^(15);           // Neutron flux of
    Group 3
9
10 // 1.
11 // Using the data given in Table II.3, for atom
    density of sodium
12 N = 0.02541*10^(24);       // Atom density in
    terms of atom/cm^3
13 // Using the data given for sigmay (Microscopic
    radiative capture cross section) in Table II.3,
14 // 1 barn = 10^(-24) cm^2
15 sigmay1 = 0.0005*10^(-24); // Microscopic gamma
    cross section of Group 1
16 sigmay2 = 0.001*10^(-24); // Microscopic gamma
    cross section of Group 2
17 sigmay3 = 0.001*10^(-24); // Microscopic gamma

```

```

        cross section of Group 3
18 // Calculation
19 F_a = N*((sigmay1*phi1)+(sigmay2*phi2)+(sigmay3*phi3
    ));
20 // Result
21 printf('\n Total absorption rate for three groups =
    %3.2E neutrons/cm^3-sec \n',F_a);
22
23 // 2.
24 // Calculation
25 sigmag_12 = 0.24*10^(-24);          // Microscopic
    scattering cross section of neutrons from Group
    1 to Group 2
26 F_12 = N*sigmag_12*phi1;
27 // Result
28 printf('\n Neutron scattering rate from the first to
    second group = %3.2E neutrons/cm^3-sec \n',F_12)
    ;

```

Scilab code Exa 5.6 Neutron diffusion

```

1 // Example 5.6
2 clear all;
3 clc;
4
5 // Given data
6 S = 10^7;          // Strength of neutron source
    in neutrons/sec
7 r = 15;           // Distance over which neutron
    flux is to be calculated in cm
8 // Using the data given in Table 5.2,
9 L_T = 2.85;       // Thermal diffusion length in
    cm
10 D_bar = 0.16;    // Diffusion coefficient in cm
11 // Calculation

```



```

12 phi_T = S*exp(-r/L_T)/(4*%pi*D_bar*r);
13 // Result
14 printf('\n Neutron flux = %3.2E neutrons/cm^2-sec \n
        ',phi_T);

```

Scilab code Exa 5.7 Neutron diffusion

```

1 // Example 5.7
2 clear all;
3 clc;
4
5 // Given data
6 T_F = 500; // Temperature in
    Fahrenheit
7 P = 2000; // Pressure in psi
8 rho = 49.6; // Density in terms of
    lb/ft^3
9 // Converting the given temperature from Fahrenheit
    to Celsius
10 T_C = (5/9)*(T_F-32);
11 // Converting the temperature from Celsius to Kelvin
    scale
12 T_K = 273+T_C;
13
14 // Using the data given in Table 5.2,
15 D_bar_0 = 0.16; // Diffusion coefficient
    at 293 K
16 rho_0 = 62.4; // Density at 293 K in
    terms of lb/ft^3
17 L_T2_0 = 8.1; // Diffusion area at 293
    K in cm^2
18 T_0 = 293; // Standard Temperature
    in kelvin
19 m = 0.47; // Material specific
    constant

```

```
20 // Calculation
21 D_bar = D_bar_0*(rho_0/rho)*(T_K/T_0)^m;
22 L_T2 = L_T2_0*(rho_0/rho)^2*(T_K/T_0)^(m+1/2);
23 // Result
24 printf('\n Diffusion coefficient of ordinary water =
    %4.3f cm \n',D_bar);
25 printf('\n Diffusion area of ordinary water = %3.1f
    cm^2 \n',L_T2);
```

Chapter 6

Neutron Reactor Theory

Scilab code Exa 6.1 Critical neutron parameters

```
1 // Example 6.1
2 clear all;
3 clc;
4
5 // Given data
6 M_F = 235; // Atomic mass of
   Uranium-235
7 M_S = 23; // Atomic mass of Sodium
   -23
8 rho_F_S = 1; // Ratio of densities of
   Uranium fuel to Sodium
9 // Using the data given in Table 5.2,
10 sigmaa_S=0.0008; // Absorption cross
   section of Sodium
11 sigmaa_F=1.65; // Absorption cross
   section of Uranium
12
13 rho_S_F = 100-rho_F_S;
14 N_S_F = rho_S_F*(M_F/M_S); // Ratio of atomic
   densities of Uranium and Sodium
15 // Using the data in Table 6.1 for Uranium-235
```

```

16 // The value of average number of neutrons produced
    for a neutron absorbed n(eta) for Uranium-235 is
    2.2
17 eta = 2.2;
18
19 // Calculation
20 f = 1/(1+(N_S_F*(sigmaa_S/sigmaa_F)));
21 k_inf = eta*f;
22 // Result
23 printf('\n Thermal Utilization factor = %.3f \n',f);
24 printf('\n Infinite Multiplication factor = %3.2f \n
    ',k_inf);

```

Scilab code Exa 6.2 Maximum and average flux

```

1 // Example 6.2
2 clear all;
3 clc;
4
5 // Given data
6 R = 50; // Radius of reactor core
    in cm
7 P = 100*10^6; // Power level of the
    reactor in watt
8 SIGMA_f = 0.0047; // Macroscopic fission
    cross section in cm(-1)
9 E_R = 3.2*10(-11); // Energy released per
    fission in joules/second
10 // Using the data in Table 6.2 for spherical
    geometry
11 OMEGA = 3.29; // Measure of the
    variation of flux in the reactor
12 // Calculation
13 phi_max = (%pi*P)/(4*E_R*SIGMA_f*R^3);
14 phi_av = phi_max/OMEGA;

```

```

15 // Result
16 printf('\n Maximum flux in the spherical reactor =
      %3.2E neutrons/cm^2-sec \n',phi_max);
17 printf('\n Average flux in the spherical reactor =
      %3.2E neutrons/cm^2-sec \n',phi_av);

```

Scilab code Exa 6.3 Critical radius

```

1 // Example 6.3
2 clear all;
3 clc;
4
5 // Given data
6 N_F = 0.00395*10^(24); // Atom
      density of Plutonium-239 fuel in atom/cm^3
7 N_S = 0.0234*10^(24); // Atom
      density of Sodium-23 in atom/cm^3
8 // Using the data given in Table 6.1,
9 // 1 barn = 10^(-24) cm^2
10 sigmaa_S = 0.0008*10^(-24); // Microscopic
      absorption cross section of Sodium in cm^2
11 sigmaa_F = 2.11*10^(-24); // Microscopic
      absorption cross section of Plutonium in cm^2
12 sigmatr_F = 6.8*10^(-24); // Microscopic
      transport cross section of Plutonium
13 sigmatr_S = 3.3*10^(-24); // Microscopic
      transport cross section of Sodium
14 // The value of average number of neutrons produced
      for a neutron absorbed n(eta) for Plutonium-239
      is 2.61
15 eta = 2.61;
16
17 SIGMAA_S = sigmaa_S*N_S; // Macroscopic
      absorption cross section of Sodium in cm^(-1)
18 SIGMAA_F = sigmaa_F*N_F; // Macroscopic

```

```

    absorption cross section of Plutonium in cm(-1)
19 SIGMAA = SIGMAA_S+SIGMAA_F;           // Total
    macroscopic absorption cross section in cm(-1)
20 SIGMA_tr = (sigmatr_F*N_F)+(sigmatr_S*N_S); //
    Macroscopic transport cross section
21 f = SIGMAA_F/SIGMAA;                 // Calculation
    of Thermal Utilization factor(f)
22 f = ceil(f);
23 k_inf = eta*f;                       // Calculation
    of Infinite Multiplication factor(k_inf)
24
25 D = 1/(3*SIGMA_tr);                 // Calculation
    of Diffusion coefficient
26 L2 = D/SIGMAA;                      // Diffusion
    area
27 d = 2.13*D;                         // Extrapolated
    distance
28 R_ctil = %pi*sqrt(L2/(k_inf-1));     // Critical
    Radius for an extrapolated boundary
29 // Calculation
30 R_c = R_ctil-d;
31 // Result
32 printf('\n Critical Radius = %2.1f cm \n',R_c);
33 // The answer given in the textbook is wrong.

```

Scilab code Exa 6.4 Non leakage probability

```

1 // Example 6.4
2 clear all;
3 clc;
4
5 // Using the result from Example 6.3
6 R_c = 48.5; // Critical Radius
    for an extrapolated boundary in cm
7 L2 = 384; // Diffusion area in

```

```

    cm^2
8 // Calculation
9 P_L = 1/(1+((%pi/R_c)^2*L2));
10 // Result
11 printf('\n Nonleakage probability of a fission
    neutron = %3.2f \n',P_L);

```

Scilab code Exa 6.5 Critical parameter calculations

```

1 // Example 6.5
2 clear all;
3 clc;
4
5 // Given data
6 R = 100; // Radius of a
    spherical reactor in cm
7 P = 10^5; // Power of the
    reactor in watt
8
9 // 1.
10 // Calculation
11 B = sqrt((%pi/R)^2);
12 // Result
13 printf("\n Buckling = %3.2E \n",B);
14
15 // 2.
16 // Using the data from Tables 3.2, 5.2, 5.3 and 6.3
17 L_TM2 = 3500; // Diffusion area of
    moderator (Sodium) in cm^2
18 n_T = 2.065; // Average number of
    fission neutrons emitted per neutron absorbed
19 t_TM = 368; // Diffusion time of
    moderator (Sodium) in cm^2
20 // 1 barn = 10^(-24) cm^2
21 sigma_aM = 0.0034*10^(-24); // Microscopic

```

```

    absorption cross section of Sodium in cm2
22 sigma_aF = 681*10(-24); // Microscopic
    absorption cross section of Uranium-235 in cm2
23 g_a = 0.978; // Non 1/v factor
24 M_F = 235; // Molecular weight of
    Uranium-235
25 M_M = 12; // Molecular weight of
    Carbon-12
26 Z = (1+B2*(L_TM2+t_TM))/(n_T-1-(B2*t_TM)); // An
    intermediate factor
27 // Calculation
28 rho_M = 1.6; // Density of Graphite
    in g/cm3
29 m_M = (4/3*pi*R3)*rho_M; // Mass of moderator
30 // Calculation
31 m_F = ((Z*sigma_aM*M_F)/(g_a*sigma_aF*M_M))*m_M
    /1000;
32 // Result
33 printf("\n Critical mass = %2.1f kg \n",m_F);
34
35 // 3.
36 f = Z/(Z+1); // Thermal
    utilization factor
37 // Calculation
38 k_inf = n_T*f;
39 // Result
40 printf('\n Infinite Multiplication factor (k_inf) =
    %.2f \n',k_inf);
41
42 // 4.
43 // Calculation
44 L_T2 = (1-f)*L_TM2
45 // Result
46 printf("\n Thermal Diffusion area = %d cm2 \n",L_T2
    );
47
48 // 5.
49 E_R = 3.2*10(-11); // Energy per fission

```



```

    reaction in joules/second
50 N_A = 6.02*10^(23);           // Avogadro number (
    constant)
51 V = (4/3*pi*R^3);           // Volume of the
    spherical reactor in cm^3
52 // Using the data from Tables 3.2
53 g_fF = 0.976;               // Non 1/v factor
    Uranium-235 fuel
54 // Using the data from Tables II.2 for Uranium-235
55 sigma_f = 582*10^(-24);     // Microscopic fission
    cross section for Uranium-235 in cm^2
56 // Macroscopic fission cross section is calculated
    as follows
57 SIGMA_f = m_F*N_A*0.886*g_fF*sigma_f*1000/(V*M_F);
58
59 // From Table 6.2, the constant A can be calculated
    as
60 A = P/(4*(R^2)*E_R*SIGMA_f);
61
62 // The expression for thermal flux is
63 printf("\n The expression for thermal flux = %4.3E
    sin (Br)/r \n",A);
64 // The maximum value of thermal flux is given at
    distance equal to zero
65 phi_T0 = A*B;
66 // Result
67 printf(" The maximum thermal flux = %4.3E neutrons/
    cm^2-sec \n",phi_T0);
68 // There is a slight variation in the values of
    diffusion area and constant A as compared from
    the textbook. This is due to approximation of
    values in textbook.

```

Scilab code Exa 6.6 Critical mass

```

1 // Example 6.6
2 clear all;
3 clc;
4
5 // Given data
6 rho_F = 0.0145; // Density of Uranium-235
   in the mixture in g/cm^3
7 rho_M = 1; // Density of Water in the
   mixture in g/cm^3
8 M_M = 18; // Molecular weight of
   water
9 M_F = 235; // Molecular weight of
   Uranium-235
10
11 // 1.
12 // The ratio of number of atoms of Uranium-235 to
   water per cc is
13 NF_NM = (rho_F*M_M)/(rho_M*M_F);
14 // Using the data from Tables 3.2
15 g_aF = 0.978; // Non 1/v factor of
   Uranium-235 fuel
16 g_aM = 2; // Non 1/v factor of Water
17 // Using the data from Table II.2 for Uranium-235
18 // 1 barn = 10^(-24) cm^2
19 sigma_aF = 681*10^(-24); // Microscopic absorption
   cross section of Uranium-235 in cm^2
20 sigma_aM=0.333*10^(-24); // Microscopic absorption
   cross section of Hydrogen in cm^2
21 // Using the data form Table 6.3 at temperature = 20
   deg
22 n_T = 2.065; // Average number of
   neutrons produced per neutron absorbed in fission
23 phisig_aF = 0.886*g_aF*sigma_aF; // Average
   thermal absorption cross-section of fuel
24 phisig_aM = 0.886*g_aM*sigma_aM; // Average
   thermal absorption cross-sections of moderator
25 Z = (NF_NM)*(phisig_aF/phisig_aM); // Parameter Z
26 f = Z/(Z+1); // Thermal

```

```

    utilization factor of the fuel
27 k_inf = n_T*f; // Infinite
    multiplication factor
28
29 // From Table 5.2 and 5.3
30 L_TM2 = 8.1; // Diffusion area in
    cm^2
31 t_T = 27; // Neutron age in cm
    ^2
32 L_T2 = (1-f)*L_TM2; // Diffusion area of
    fuel moderator mixture
33 M_T2 = L_T2+t_T; // Migration area of
    fuel moderator mixture
34 // Buckling can be found as
35 B2 = (k_inf-1)/M_T2;
36 printf(" \n Using the buckling formula from Table
    6.2 \n B^2 = (2.405/R)^2+(pi/H)^2 \n For minimum
    critical mass H = 1.82R \n");
37 // On solving for R in B^2 = 8.763/R^2
38 R = sqrt(8.763/B2);
39 H = 1.82*R;
40 // Result
41 printf(" \n The dimensions of the cylinder are");
42 printf(" \n Radius of cylinder = %2.1f cm \t Height
    of cylinder = %3.1f cm \n",R,H);
43
44 // 2.
45 V = %pi*R^2*H; // Reactor
    volume (in cc) assuming cylindrical geometry
46 // Calculation
47 m_F = rho_F*V;
48 printf(" \n The critical fuel mass = %2.1f kg \n",
    m_F/1000);
49 // There is a slight variation in the values of
    dimensions of cylinder and critical fuel mass as
    compared from the textbook. This is due to
    approximation of values in textbook.

```

Scilab code Exa 6.7 Critical concentration

```
1 // Example 6.7
2 clear all;
3 clc;
4
5 // Given data
6 R = 300; // Radius of the sphere
   in cm
7 M_M = 20; // Molecular weight of
   heavy water
8 M_F = 235; // Molecular weight of
   Uranium-235
9
10 // 1.
11 // Using the data from Table 5.2
12 Dbar_r = 0.84; // Diffusion coefficient
   of graphite in cm
13 Dbar_c = 0.87; // Diffusion coefficient
   of heavy water in cm
14 L_TM2 = 9400; // Diffusion area of
   heavy water in cm^2
15 L_r = 59; // Diffusion length of
   graphite in cm
16 // Using the data from Table 3.2
17 g_aF = 0.978; // Non 1/v factor
   Uranium-235 fuel
18 // Using the data from Table II.2 for Uranium-235
19 // 1 barn = 10^(-24) cm^2
20 sigma_aF = 681*10^(-24); // Microscopic
   absorption cross section of Uranium-235 in cm^2
21 SIGMA_aM = 9.3*10^(-5)*10^(-24); // Macroscopic
   absorption coefficient of Heavy water in cm^(-1)
22 N = 0.03323; // Atomic
```

```

    density of heavy water
23 // Let BRcot(BR)= y
24 y = 1-((Dbar_r/Dbar_c)*((R/L_r)+1));
25 // Considering only the first solution , B*R=2.64
26 B = 2.64/R;
27 // Using the data form Table 6.3 at temperature = 20
    deg
28 n_T = 2.065; // Average
    number of neutrons produced per neutron absorbed
    in fission
29 Z = (1+(B^2*L_TM2))/(n_T-1); // A parameter
30 sigma_aM = sqrt(4/%pi)*SIGMA_aM/N; // Microscopic
    absorption cross section of Heavy water in cm^2
31 // The ratio of densities of fuel to moderator
32 rho_FM = Z*(M_F*sigma_aM)/(M_M*g_aF*sigma_aF)
33 rho_M = 1.1; // Density of
    Heavy water in g/cm^3
34 // Calculation
35 rho_F = rho_FM*rho_M;
36 // Result
37 printf(" \n The critical concentration = %.4f g/
    litre \n",rho_F*1000);
38
39 // 2.
40 V = (4/3)*%pi*R^3; // Reactor
    volume (in cc) assuming spherical geometry
41 // Calculation
42 m_F = rho_F*V;
43 // Result
44 printf(" \n The critical fuel mass = %3.2f kg \n",
    m_F/1000);

```

Scilab code Exa 6.8 Critical radius

```

1 // Example 6.8

```

```

2 clear all;
3 clc;
4
5 // Given data
6 rho_F = 2*10^(-4);           // Concentration of
    Uranium-235 fuel in g/cm^3
7 rho_M = 1.6;                // Concentration of
    graphite moderator in g/cm^3
8 M_F = 235;                  // Molecular mass of
    Uranium-235 fuel
9 M_M = 12;                   // Molecular mass of
    Graphite(Carbon) moderator
10
11 // 1.
12 // Using the data from Tables 3.2
13 g_aF = 0.978;              // Non 1/v factor
    Uranium-235 fuel
14 // Using the data from Table II.2 for Uranium-235
    and Carbon
15 // 1 barn = 10^(-24) cm^2
16 sigma_aF = 681*10^(-24);   // Microscopic
    absorption cross section of Uranium-235 in cm^2
17 sigma_aM = 3.4*10^(-3)*10^(-24); // Microscopic
    absorption cross section of Graphite in cm^2
18 Z = (rho_F*M_M*g_aF*sigma_aF)/(rho_M*M_F*sigma_aM);
    // Parameter Z
19 f = Z/(Z+1);                // Thermal
    utilization factor of the fuel
20 // Using the data form Table 6.3 at temperature = 20
    deg
21 n_T = 2.065;               // Average number of
    neutrons produced per neutron absorbed in
    fission
22 k_inf = n_T*f;             // The infinite
    multiplication factor
23 // From Table 5.2
24 L_TM2 = 3500;              // Diffusion area of
    Graphite in cm^2

```

```

25 L_r = 59; // Diffusion length
    of graphite in cm
26 L_T2 = (1-f)*L_TM2; // Diffusion area of
    fuel moderator mixture
27 // Buckling can be found as
28 B = sqrt((k_inf-1)/L_T2);
29 // Calculation
30 R = acot(-1/(B*L_r))/B;
31 // Result
32 printf(" \n The critical radius of fuel loaded
    thermal reactor = %3.2f cm \n",R);
33
34 // 2.
35 // Reactor is bare or reflector is not present
36 // Calculation
37 R0 = %pi/B;
38 // Result
39 printf(" \n The critical radius of bare thermal
    reactor = %d cm \n",R0);
40 // There is a slight variation in the value of
    critical radius as compared from the textbook.
    This is due to approximation of the thermal
    utilization factor value in textbook.

```

Scilab code Exa 6.9 Critical radius and mass

```

1 // Example 6.9
2 clear all;
3 clc;
4
5 // Given data
6 rho_F = 0.0145; // Concentration of
    Uranium-235 fuel in g/cm^3
7 // Using the result of Example 6.6
8 M_T2 = 30.8; // Migration area in

```

```

          cm^2
9  B = 0.0529;           // Buckling factor
10 delta = 7.2+0.1*(M_T2-40); // Empirical formula
    for reflector savings
11 R0 = %pi/B;          // The radius of the
    bare reactor
12 // Calculation
13 R = R0-delta;
14 m_F=rho_F*4/3*%pi*R^3;
15 // Result
16 printf(" \n The critical radius of reflected reactor
    = %3.2f cm \n",R);
17 printf(" \n The critical mass of reflected reactor =
    %3.2f kg \n",m_F/1000);

```

Scilab code Exa 6.10 Critical parameters for heterogenous reactor

```

1 // Example 6.10
2 clear all;
3 clc;
4
5 // Given data
6 N = 150;           // Number of zirconium
    atoms for every uranium atom
7
8 // 1.
9 // Using the data of atom density of zirconium from
    Table II.3
10 N_Z = 0.0429;     // Atom density of
    zirconium in terms of 10^(24)
11 sigma_tZ = 6.6;   // Total cross section
    of zirconium in barns
12 // Using the data of cross section of uranium-235
    from Table II.3
13 sigma_tU = 690;   // Total cross section

```



```

    of uranium in barns
14 N_25 = N_Z/N;           // Atom concentration
    of uranium-235
15 // Calculation
16 lambda = 1/((sigma_tZ*N_Z)+(sigma_tU*N_25));
17 // Result
18 printf("\n The mean free path of thermal neutrons =
    %3.1f cm \n",lambda);
19
20 // 2.
21 // Using the data of atom density of water from
    Table II.3
22 N_W = 0.0334;           // Atom density of
    water in terms of 10^(24)
23 // As the water and zirconium occupy half of the
    volume
24 N_W = 0.5*0.0334;
25 N_Z = 0.5*0.0429;
26 // From the Figure 6.6
27 // Uranium is present in one third of the sandwich
    or \n one sixth of the entire area
28 N_25 = 2.86*10^(-4)/6;
29 // Using the data from Table 3.2
30 g_aF = 0.978;           // Non 1/v factor Uranium
    -235 fuel
31 // Using the data from Table II.3 for microscopic
    absorption cross section
32 sigma_aU = 681;         // Microscopic absorption
    cross section of Uranium-235 in barns
33 sigma_aZ = 0.185;       // Microscopic absorption
    cross section of Zirconium in barns
34 sigma_aW = 0.664;       // Microscopic absorption
    cross section of Water in barns
35 f = (N_25*g_aF*sigma_aU)/((N_25*g_aF*sigma_aU)+(N_Z*
    sigma_aZ)+(N_W*sigma_aW)); // Thermal
    utilization factor
36 // Using the data form Table 6.3 at temperature = 20
    deg

```

```

37 n_T = 2.065;           // Average number of
    neutrons produced per neutron absorbed in fission
38 // Calculation
39 k_inf = n_T*f;
40 // Result
41 printf("\n Infinite multiplication factor = %4.3f \n
    ",k_inf);

```

Scilab code Exa 6.11 Critical parameter for heterogenous reactor

```

1 // Example 6.11
2 clear all;
3 clc;
4
5 // Using the data from Table 3.2
6 g_a25=0.978;           // Non 1/v factor Uranium
    -235 fuel for absorption
7 g_f25=0.976;           // Non 1/v factor Uranium
    -235 fuel for fission
8 g_a28=1.0017;          // Non 1/v factor Uranium
    -238 fuel for absorption
9
10 v_25=2.42;            // Average number of
    neutrons in one fission of Uranium-235
11 // Using the data from Table II.3 for microscopic
    absorption and fission cross section
12 sigma_a25=681;        // Microscopic absorption
    cross section of Uranium-235 in barns
13 sigma_a28=2.7;        // Microscopic absorption
    cross section of Uranium-238 in barns
14 sigma_f25=582;        // Microscopic fission
    cross section of Uranium-235 in barns
15
16 // Using the data of atom density of uranium and let
    N_28/N_25= N

```

```

17 N = 138;
18 // Calculation
19 n_T = (v_25*sigma_f25*g_f25)/((sigma_a25*g_a25)+(N*
    sigma_a28*g_a28));
20 // Result
21 printf("\n Average number of neutrons produced per
    neutron absorbed in fission = %3.2f \n",n_T);

```

Scilab code Exa 6.12 Critical parameter for heterogenous reactor

```

1 // Example 6.12
2 clear all;
3 clc;
4
5 // Given data
6 rdist = 25.4; // Distance
    between the rods in cm
7 a = 1.02; // Radius of a rod
    in cm
8 // From the Figure 6.9
9 b = rdist/sqrt(%pi); // Radius of
    equivalent cell in cm
10 // Using the data from Table 5.2
11 L_F = 1.55; // Diffusion length
    of uranium fuel in cm
12 L_M = 59; // Diffusion length
    of graphite moderator in cm
13 // Using the data from Table II.3 at thermal energy
14 SIGMA_aM = 0.0002728; // Macroscopic
    absorption cross section of graphite moderator in
    barns
15 SIGMA_aF = 0.3668; // Macroscopic
    absorption cross section of uranium fuel in barns
16 // Let
17 x = a/L_F;

```

```

18 y = a/L_M;
19 z = b/L_M;
20 // The series expansion relations are
21 F = 1+(0.5*(x/2)^2)-((1/12)*(x/2)^4)+((1/48)*(x/2)
    ^6);
22 E = 1+(z^2/2)*(((z^2*log(z/y))/(z^2-y^2))-(3/4)+(y
    ^2/(4*z^2)));
23 // Let the ratio of volumes of moderator to fuel is
    denoted by V
24 V = (b^2-a^2)/a^2;
25 // Calculation
26 f = 1/((SIGMA_aM*V*F/SIGMA_aF)+E);
27 // Result
28 printf("\n The thermal utilization factor = %.4f \n"
    ,f);
29 // There is a slight variation in the value as
    compared from the textbook. This is due to
    approximation of the parameters value in textbook
    .

```

Scilab code Exa 6.13 Critical parameter for heterogenous reactor

```

1 // Example 6.13
2 clear all;
3 clc;
4
5 // Using the data given in the problem 6.12
6 rdist = 25.4; // Distance
    between the rods in cm
7 a = 1.02; // Radius of
    the rod in cm
8 b = rdist/sqrt(%pi); // Radius of
    equivalent cell
9 V = (b^2-a^2)/a^2; // Ratio of
    volumes of moderator to fuel

```

```

10 // Using the data from Table II.3 for Uranium-238
    density and atom density
11 rho = 19.1; // Uranium-238
    density in g/cm^3
12 N_F = 0.0483; // Atom
    density in terms of 10^(24)
13 // Using Table 6.5 for Uranium-238
14 A = 2.8;
15 C = 38.3;
16 // Using Table 6.6 for graphite
17 // Let zeta_M*SIGMA_sM = s
18 s = 0.0608;
19 I = A+C/sqrt(a*rho); // Empirical
    expression of resonance integral parameter
20 // Calculation
21 p = exp(-(N_F*I)/(s*V));
22 // Result
23 printf("\n Resonance escape probability = %.4f \n",p
    );

```

Chapter 7

The Time Dependent Reactor

Scilab code Exa 7.1 Reactor kinetics

```
1 // Example 7.1
2 clear all;
3 clc;
4
5 // Using the data form Table 6.3 at temperature = 20
  deg
6 n_T = 2.065; // Average number of
  neutrons produced per neutron absorbed in fission
7 // Using the data from Table 7.1
8 t_dM = 2.1e-4; // The mean diffusion time
  of the moderator in seconds
9 k_inf = 1; // The reactor is critical
10 f = k_inf/n_T; // Thermal utilization
  factor
11 // Calculation
12 t_d = t_dM*(1-f);
13 l_p = t_d;
14 // Result
15 printf("\n The prompt neutron lifetime = %3.2E
  seconds \n",l_p);
```

Scilab code Exa 7.2 Reactor period

```
1 // Example 7.2
2 clear all;
3 clc;
4
5 // Given data
6 k_inf = 1.001; // Infinite
   multiplication factor
7 // From the Example 7.1
8 l_p = 1e-4; // Prompt
   neutron lifetime
9 // Calculation
10 T = l_p/(k_inf-1);
11 // Result
12 printf("\n The response time of the reactor = %2.1f
   sec \n",T);
13 printf("\n The reactor power will increase as exp(t
   /%2.1f), where 't' denotes the time in seconds
   \n",T);
```

Scilab code Exa 7.3 Reactivity

```
1 // Example 7.3
2 clear all;
3 clc;
4
5 // Given data
6 k_inf = 1.001; // Infinite
   multiplication factor
7 // Calculation
8 rho = (k_inf-1)/k_inf;
```

```

9 // Result
10 printf(" \n The reactivity = %.1E or %2.1f percent \
    n",rho,rho*100);

```

Scilab code Exa 7.4 Reactor period

```

1 // Example 7.4
2 clear all;
3 clc;
4
5 // Using the result of Example 7.1
6 lp = 1e-4; // Prompt neutron
    lifetime in seconds
7 // Using the result of Example 7.3
8 rho = 1e-3; // Reactivity
9 // By referring to Figure 7.2
10 printf(" \n Reactor period = 57 seconds \n");

```

Scilab code Exa 7.5 Reactivity

```

1 // Example 7.5
2 clear all;
3 clc;
4
5 // Using the result of Example 7.3
6 reactivity = 0.001;
7 // As the reactor is fueled with Uranium-235
8 bet = 0.0065; // Total delayed
    neutron fraction of all groups denoted by 'beta'
9 printf(" \n A dollar is worth 0.0065 in reactivity
    for Uranium-235 reactor. \n");
10 // Calculation
11 rho = reactivity/bet;

```



```

12 // Result
13 printf("\n Reactivity = %4.3f dollars or %2.1f
    cents \n",rho,rho*100);

```

Scilab code Exa 7.6 Prompt drop

```

1 // Example 7.6
2 clear all;
3 clc;
4
5 // Given data
6 P0 = 500; // Reactor power
    in MW
7 rho = -0.1; // 10% in
    reactivity (Insertion of control rods correspond
    to negative reactivity)
8 // As the reactor is fueled with Uranium-235
9 bet = 0.0065; // Total delayed
    neutron fraction of all groups denoted by 'beta'
10
11 P1 = (bet*(1-rho)*P0)/(bet-rho); // The drop in
    power level in terms of MW
12 // Assuming that negative reactivity is greater than
    4%
13 T = 80; // Reactor
    period obtained from Figure 7.2 in seconds
14 t = 600; // Analysis time
    in seconds
15 // Calculation
16 P = P1*exp(-t/T); // Power level
    drop in MW
17 // Result
18 printf("\n The power level drop after 10 minutes =
    %5.4f MW \n",P);

```

Scilab code Exa 7.7 Control worth of a center rod

```
1 // Example 7.7
2 clear all;
3 clc;
4
5 // Given data
6 H = 70; //
   Height of the cylinder in cm
7 R = H/2; //
   Diameter of the cylinder in cm
8 a = 1.9; //
   Radius of black control rod in cm
9 // From Table 6.2, Buckling can be found by
10 B0 = sqrt((2.405/R)^2+(%pi/H)^2);
11 // Using the data from Table 5.2 and 5.3
12 L_TM2 = 8.1; //
   Diffusion area of water moderator in cm^2
13 t_TM = 27; //
   Neutron age of water moderator in cm^2
14 // Using the data form Table 6.3 at temperature = 20
   deg
15 n_T = 2.065; //
   Average number of neutrons produced per neutron
   absorbed in fission
16 // Using the data from Table 5.2 and Table II.3
17 D_bar = 0.16; //
   Thermal neutron diffusion coefficient in cm
18 SIGMA_t = 3.443; //
   Total macroscopic cross section in cm^(-1)
19 f = (1+B0^2*(L_TM2+t_TM))/(n_T+B0^2*L_TM2); //
   Thermal utilization factor
20 M_T2 = (1-f)*L_TM2+t_TM; //
   Thermal migration area in cm^2
```

```

21 d = 2.131*D_bar*(a*SIGMA_t+0.9354)/(a*SIGMA_t
    +0.5098); // Extrapolation distance
22 // Calculation
23 rho_w = (7.43*M_T2*(0.116+log(R/(2.405*a))+(d/a))
    ^(-1))/((1+B0^2*M_T2)*R^2);
24 // Result
25 printf(" \n The worth of a black control rod = %.3f
    or %2.1f percent \n",rho_w,rho_w*100);

```

Scilab code Exa 7.8 Total control rod worth

```

1 // Example 7.8
2 clear all;
3 clc;
4
5 // Using the data and result from Example 7.7
6 f = 0.583; //
    Thermal Utilization factor
7 L_TM2 = 8.1; //
    Diffusion area of water moderator in cm2
8 R = 35; // Radius
    of the cylinder of the core in cm
9 a = 0.508; // Radius
    of control rod in cm
10 Rc = sqrt(R^2/100); //
    Critical radius in cm
11 L_T = sqrt((1-f)*L_TM2); //
    Thermal diffusion length in cm
12 // The points of estimation are chosen as follows
13 y = a/L_T;
14 z = Rc/L_T;
15 // Using the data given in Table V.I for modified
    Bessel functions
16 I0_275 = 1.019; // I0 at
    0.275

```

```

17 I1_275 = 0.1389; // I1 at
    0.275
18 I1_189 = 1.435; // I1 at
    1.89
19 K0_275 = 1.453; // K0 at
    0.275
20 K1_275 = 3.371; // K1 at
    0.275
21 K1_189 = 0.1618; // K1 at
    1.89
22 E = ((z^2-y^2)/(2*y))*(((I0_275*K1_189)+(K0_275*
    I1_189))/((I1_189*K1_275)-(K1_189*I1_275)));
    // The lattice
    function
23 // Using the data from Table 5.2 and Table II.3
24 D_bar = 0.16; // Thermal
    neutron diffusion coefficient in cm
25 SIGMA_t = 3.443; // Total
    macroscopic cross section in cm(-1)
26 d = 2.131*D_bar*(a*SIGMA_t+0.9354)/(a*SIGMA_t
    +0.5098); // Extrapolation distance
27 f_R = 1/(((z^2-y^2)*d)/(2*a))+E); // Rod
    utilization parameter
28 // Calculation
29 rho_w = f_R/(1-f_R);
30 // Result
31 printf("\n The total worth of the control rods = %
    .3f or %2.1f percent \n",rho_w,rho_w*100);
32 // There is a deviation in the value computed on
    comparison with the value given in the textbook.
    This is due to approximation of thermal diffusion
    area in the textbook.

```

Scilab code Exa 7.9 Total control rod worth

```

1 // Example 7.9
2 clear all;
3 clc;
4
5 // Given data
6 SIGMAa_bar = 0.2; //
   Average macroscopic absorption cross section in
   cm(-1)
7 L_T = 1.2; //
   Thermal diffusion length in cm
8 // Converting the given dimensions from inches to
   centimeters
9 // 1 inch = 2.54 cm
10 // From Figure 7.9
11 l = 9.75*(2.54/2); //
   Length of the half rod
12 a = 0.312*(2.54/2); //
   Thickness of the half rod
13 m = 44.5/sqrt(2); //
   Closest distance between two rods
14
15 D_bar = SIGMAa_bar*L_T^2; //
   Thermal neutron diffusion coefficient in cm
16 d = 2.131*D_bar; //
   Extrapolation distance in cm which is obtained
   for bare planar surface
17 f_R = ((4*(1-a)*L_T)/(m-(2*a))^2*(1/((d/L_T)+coth((m
   -(2*a))/(2*L_T))))); // Rod utilization
   parameter
18 // Calculation
19 rho_w = f_R/(1-f_R);
20 // Result
21 printf("\n The total worth of the control rods = %
   .3f or %.1f percent \n",rho_w,rho_w*100);
22 // There is a slight deviation in the value computed
   on comparison with the value given in the
   textbook. This is due to approximation of rod
   utilization parameter in the textbook.

```

Scilab code Exa 7.10 Total control rod worth

```
1 // Example 7.10
2 clear all;
3 clc;
4
5 // Given data
6 d = 5; // Inner
   diameter of the tube in cm
7 a = d/2; // Inner
   radius of the tube in cm
8 l = 76; // Length
   of the tube in cm
9 rho = 2; // Density
   of B4C in g/cm^3
10 n = 5; // Number
   of rods in the reactor
11 m_B4C = 2*(n*pi*(a^2)*l); // Mass of
   B4C in all the rods
12 // Using the data from standard periodic table
13 molwt_B = 10.8; //
   Molecular weight of Boron(B)
14 molwt_C = 12; //
   Molecular weight of Carbon(C)
15 molwt_B4C = (4*molwt_B)+molwt_C; //
   Molecular weight of B4C
16 N_A = 0.6*10^(24); //
   Avogadro number
17 // From Table II.3
18 sigma_a = 0.27*10^(-24); //
   Microscopic absorption cross section of boron in
   cm^2
19 n_B = (4*m_B4C*N_A)/molwt_B4C; // Number
   of boron atoms
```

```

20 // Using the result of Example 6.3
21 SIGMA_aF = 0.00833; //
    Macroscopic absorption cross section of plutonium
    fuel in cm(-1)
22 SIGMA_aC = 0.000019; //
    Macroscopic absorption cross section of sodium
    coolant in cm(-1)
23 R_c = 41.7; //
    Critical radius in cm
24 N_B = n_B/((4/3)*%pi*R_c^3); // Atom
    density of boron over an entire reactor assuming
    spherical shape
25 SIGMA_aB = sigma_a*N_B; //
    Macroscopic absorption cross section of boron
26 // Calculation
27 rho_w = SIGMA_aB/(SIGMA_aF+SIGMA_aC);
28 // Result
29 printf(" \n The worth of the control rods using one
    group theory = %.4f or %.2f percent \n",rho_w,
    rho_w*100);
30 // In textbook, the final answer of total worth of
    control rods in percentage is wrong.

```

Scilab code Exa 7.11 Differential control rod worth

```

1 // Example 7.11
2 clear all;
3 clc;
4
5 // Given data
6 H = 70; //
    Height of square cylindrical reactor in cm
7 rho_wH = 0.065; //
    Total worth of a control rod at full height
8 rho_wX = 0.01; //

```

```

    Total worth of a control rod to be achieved
9 // Let  $y - \sin(y) = t$ 
10 t = 2*%pi*(rho_wx/rho_wH);
11 // Using Newton Raphson method for solving the
    transcendental equation  $y - \sin(y) - 0.966 = 0$ 
12 deff('y=f(y)', 'y = y-sin(y)-0.966')
13 deff('y=f1(y)', 'y = 1-cos(y)')
14 y0=0.5; // Initial value
15 e = 0.00001; // Relative error tolerance
16 for i=1:4
17     y1 = y0-f(y0)/f1(y0)
18     e1 = abs(y0-y1)
19     y0 = y1;
20     if abs(y0)<e then
21         break;
22     end
23 end
24 y = y1; //
    The solution of transcendental equation
25 // Calculation
26 x = (y*H)/(2*%pi);
27 // Result
28 printf('\n The length of control rod to be inserted
    = %2.1 f cm \n', x);

```

Scilab code Exa 7.12 Chemical Shim

```

1 // Example 7.12
2 clear all;
3 clc;
4
5 // Given data
6 f0 = 0.93; // Thermal
    utilization factor
7 rho = 0.205; // Total

```



```

    excess reactivity
8 rho_w = 0.085; // Total
    worth of control rods
9 rho_sh = rho-rho_w; // Total
    worth of shim control
10 C = (rho_sh*10^3)/(1.92*(1-f0)); //
    Concentration of boric acid in ppm
11 printf('\n The minimum concentration of boric acid =
    %d ppm \n',ceil(C));
12 // Expressing in gram/litre
13 // Using the data from standard periodic table
14 molwt_B = 10.8; //
    Molecular weight of Boron(B)
15 molwt_O = 16; //
    Molecular weight of Oxygen(O)
16 molwt_H = 1; //
    Molecular weight of Hydrogen(H)
17 molwt_H3B03 = (3*molwt_H)+molwt_B+(3*molwt_O);
    // Molecular weight of Boric acid
18 // Calculation
19 amt_H3B03 = (molwt_H3B03/molwt_B)*C/1000;
20 // Result
21 printf("\n The shim system must contain %3.2f g/
    litre of boric acid to hold down the reactor. \n"
    ,amt_H3B03);

```

Scilab code Exa 7.13 Temperature coefficient of reactivity

```

1 // Example 7.13
2 clear all;
3 clc;
4
5 // Given data
6 p = 0.878; //
    Resonance escape probability

```

```

7 T = 273+350; // Given
   temperature converted in Kelvin
8 d = 2.8; //
   Diameter of rod in cm
9 a = d/2; // Radius
   of rod in cm
10 rho = 19.1; // Density
   of uranium in g/cm^3
11 // Using data from Table 7.4 for Uranium-238
12 A = 48*10^(-4); //
   Constant value
13 C = 1.28*10^(-2); //
   Constant value
14 beta_I = A+C/(a*rho); // A
   parameter
15
16 // Calculation
17 alpha_prompt = -(beta_I/(2*sqrt(T)))*log(1/p);
18 // Result
19 printf('\n The prompt temperature coefficient = %.2E
   per K \n',alpha_prompt);

```

Scilab code Exa 7.14 Fission product poisons

```

1 // Example 7.14
2 clear all;
3 clc;
4
5 // Assuming that the fission product poisoning
   results in 12 barns per original Uranium-235 atom
   in a time frame of one year
6 sigma_p = 12; //
   Microscopic poison cross section in barns
7 v = 2.42; // Average
   number of neutrons produced in fission

```

```
8 // Using Table II.2 for fission cross section of
   Uranium-235 at thermal energy
9 sigma_f = 587; //
   Microscopic fission cross section in barns
10 // Calculation
11 rho = -sigma_p/(v*sigma_f);
12 // Result
13 printf(" \n The reactivity due to poisons = %.5f or
   %.3f percent \n",rho,rho*100);
```

Chapter 8

Heat Removal from Nuclear Reactors

Scilab code Exa 8.1 Coolant temperature

```
1 // Example 8.1
2 clear all;
3 clc;
4
5 // Given data
6 P = 3025; // Reactor
   thermal power in MW
7 w = 136.3*10^6; // Coolant
   flow rate in lb/hr
8 // According to Table 1.9
9 // 1 kW = 3412 Btu/hr
10 q = P*1000*3412; //
   Converting into Btu/hr
11 delh = q/w; // Rise in
   enthalpy
12 // Using the data from Table IV.1 for temperature
   542.6 F
13 hin = 539.7; //
   Enthalpy of input water in Btu/lb
```

```

14 // Calculation
15 hout = hin+delh; //
    Enthalpy of released water in Btu/lb
16 // From Table IV.1
17 // Result
18 printf(" \n %3.1f Btu/lb corresponds to 599 F
    coolant water temperature. \n",hout);

```

Scilab code Exa 8.2 Steam temperature

```

1 // Example 8.2
2 clear all;
3 clc;
4
5 // Given data
6 P = 6.895; //
    Pressure of steam in MPa
7 w = 2.93*10^6; // Steam
    flow rate in kg/hr
8 Tin = 190.6+273; // Inlet
    temperature in Kelvin
9
10 // 1.
11 // Using the data from Table IV.2
12 // Result
13 printf(" \n At a pressure of 6.895 MPa the steam
    temeperature is 284.86 C \n");
14
15 // 2.
16 // Using the data from Table IV.2
17 hout = 2773.2; //
    Enthalpy of spent steam in kJ/kg
18 // Using the data from Table IV.1
19 hin = 807.8; //
    Enthalpy of inlet steam at Tin in kJ/kg

```

```

20 // Calculation
21 q = w*(hout-hin);
22 // Result
23 printf(" \n Reactor power is %.3E J/hr or %d MW \n",
        q,q/(3600*1000));

```

Scilab code Exa 8.3 Heat production in fuel rods

```

1 // Example 8.3
2 clear all;
3 clc;
4
5 // Given data
6 n = 193*204; // Total
    number of fuel rods in the reactor
7 // 1 feet = 12 inches
8 R = 67/12; // Outer
    radius of the cylinder in feet(ft)
9 H = 144/12; // Outer
    radius of the cylinder in ft
10 d = 0.42/12; //
    Diameter of the fuel rod in ft
11 a = d/2; // Radius
    of the fuel rod in ft
12 P = 1893; // Reactor
    thermal power in MW
13 Ed = 180; // Energy
    deposited locally in the fuel per fission in MW(
    Assumption)
14 ER = 200; //
    Recoverable energy per fission in MW(Assumption)
15
16 // 1.
17 // Calculation
18 // According to Table 1.9

```

```

19 // 1 kW=3412 Btu/hr
20 q_r = (2.32*P*Ed)/(n*ER);
21 q_max=(q_r*3412*1000)/(2*H*a^2);
22 // Result
23 printf(" \n Total energy production at the axis = %
      .2E Btu/hr", (q_r*3412*1000));
24 printf(" \n Maximum energy production at the axis =
      %.2E Btu/hr-ft^3 \n", q_max);
25
26 // 2.
27 r = 20/12; //
      Distance from the axis in ft
28 j0 = besselj(0, ((2.405*r)/R)); //
      Bessel function
29 // Calculation
30 // According to Table 1.9
31 // 1 kW=3412 Btu/hr
32 q_r20 = q_r*j0;
33 q_max20 = (q_r20*3412*1000)/(2*H*a^2);
34 // Result
35 printf(" \n Total energy production at a distance of
      20 inches = %.2E Btu/hr", (q_r20*3412*1000));
36 printf(" \n Maximum energy production at a distance
      of 20 inches = %.2E Btu/hr-ft^3 \n", q_max20);

```

Scilab code Exa 8.4 Decay energy

```

1 // Example 8.4
2 clear all;
3 clc;
4
5 // Given data
6 P0 = 825; //
      Reactor thermal power in MW
7 t0 = 1.5*3.16*10^7; //

```

```

      Reactor operation time in seconds
8  ts = 0.1;                                     //
      Reactor shutdown time in seconds
9
10 // 1.
11 // Let P/P0 = q
12 // From Figure 8.3
13 q_ts = 0.07;                                  //
      Fission product to decay power during shutdown
      time
14 q_t0 = 0.0007;                                //
      Fission product to decay power after operating
      time
15 q = q_ts - q_t0;                              // Net
      fission product to decay power
16 // Calculation
17 P = q * P0;
18 // Result
19 printf(" \n Decay energy at shutdown = %2.1f MW", P);
20
21 // One hour after shutdown
22 ts1 = 3.6 * 10^3;                              //
      Reactor shutdown time in seconds
23 // Let P/P0=q
24 // From Figure 8.3
25 q_ts1 = 0.014;                                //
      Fission product to decay power at shutdown time
26 q_t0 = 0.0007;                                //
      Fission product to decay power after operating
      time
27 q1 = q_ts1 - q_t0;
28 // Calculation
29 P1 = q1 * P0;
30 // Result
31 printf(" \n Decay energy one hour after shutdown =
      %2.1f MW", P1);
32
33 // One year after shutdown

```



```

34 ts2 = 3.16*10^7; //
    Reactor shutdown time in seconds
35 // Let P/P0=q
36 // From Figure 8.3
37 q_ts2 = 0.00079; //
    Fission product to decay power at shutdown time
38 // Now the operating time is t0+ts2 which can be
    denoted by t01
39 q_t01 = 0.00063; //
    Fission product to decay power after operating
    time
40 q2 = q_ts2-q_t01;
41 // Calculation
42 P2 = q2*P0;
43 // Result
44 printf(" \n Decay energy one year after shutdown = %
    .3f MW \n",P2);
45
46 // 2.
47 C = 0.88; //
    Conversion factor
48 // Using data from Table II.2
49 sigma_a25 = 681; //
    Microscopic absorption cross section in barns
50 sigma_f25 = 582; //
    Microscopic fission cross section in barns
51 // At shutdown time
52 P_29 = (2.28*10^(-3)*C*(sigma_a25/sigma_f25))*P0;
53 P_39 = (2.17*10^(-3)*C*(sigma_a25/sigma_f25))*P0;
54 printf(" \n Decay energy at shutdown with effect of
    Uranium-239 and Neptunium-239 decay = %2.2f MW
    and %2.2f MW respectively",P_29,P_39);
55
56 // One hour after shutdown
57 ts1 = 3600; //
    Time in seconds
58 P_291 = P_29*exp(-4.9*10^(-4)*ts1);
59 P_391 = P_39*(exp(-3.41*10^(-6)*ts1)-(7*10^(-3)*exp

```

```

        (-4.9*10(-4)*ts1))));
60 printf(" \n Decay energy one hour after shutdown
    with effect of Uranium-239 and Neptunium-239
    decay = %2.2f MW and %2.2f MW respectively",P_291
    ,P_391);
61
62 // One year after shutdown
63 P_292 = 0; //
    Half life of Uranium-239 is 23.5 minutes
64 P_392 = 0; //
    Half life of Neptunium-239 is 2.35 days
65 printf(" \n Decay energy one year after shutdown
    with effect of Uranium-239 and Neptunium-239
    decay = %d MW and %d MW respectively",P_292,P_392
    );
66 // There is a slight deviation in the values as
    compared with the texbook. This is because of
    approximation of difference values in the
    textbook.

```

Scilab code Exa 8.5 Fuel rod temperature parameters

```

1 // Example 8.5
2 clear all;
3 clc;
4
5 // Given data
6 // 1 feet = 12 inches
7 d = 0.42/12; //
    Diameter of the fuel rod in feet(ft)
8 a = d/2; // Radius
    of the fuel rod in ft
9 b = 0.024/12; //
    Thickness of Zircaloy-4 clad in ft
10 H = 12; // Length

```

```

    of fuel rod in ft
11 T_m = 3970; // Center
    temperature of fuel in F
12
13 // 1.
14 // Using the result of Example 8.3
15 q_max = 4.66*10^7; // Maximum
    heat flux at the center of the rod in Btu/hr-ft
    ^3
16 // Calculation
17 q_bar = (a^2*q_max)/(2*(a+b));
18 // According to Table 1.9
19 // 1 kW=3412 Btu/hr
20 // Result
21 printf("\n Heat flux of the fuel rod = %.2E Btu/hr-
    ft^2 or %3.1f W/cm^2 \n",q_bar,(q_bar*1000)
    /(3412*30.48^2));
22
23 // 2.
24 // Using the data from Table IV.6
25 k_f = 1.1; // Thermal
    conductivity of fuel rod in Btu/hr-ft-F
26 k_c = 10; // Thermal
    conductivity of cladding in Btu/hr-ft-F
27 R_f = 1/(4*pi*H*k_f); // Thermal
    resistance of fuel in F-hour/Btu
28 R_c = log(1+(b/a))/(2*pi*H*k_c); // Thermal
    resistance of cladding in F-hour/Btu
29 // Calculation
30 T_c = T_m-(q_bar*2*pi*(a+b)*H*(R_f+R_c));
31 // Result
32 printf("\n Outer temperature of cladding = %d F \n"
    ,ceil(T_c));

```

Scilab code Exa 8.6 Coolant temperature

```

1 // Example 8.6
2 clear all;
3 clc;
4
5 // Given data
6 h = 7500; //
   Heat transfer coefficient in Btu/hr-ft^2-F
7 // Using the result of Example 8.5
8 q_bar = 3.66*10^5; //
   Heat flux of the fuel rod in Btu/hr-ft^2
9 T_c = 650; //
   Outer temperature of cladding in F
10 // Calculation
11 T_b = T_c - (q_bar/h);
12 // Result
13 printf("\n Temperature of water with respect to the
   midpoint of the hottest fuel rod = %d F \n", T_b)
   ;

```

Scilab code Exa 8.7 Fuel rod temperature parameters and coolant temperature

```

1 // Example 8.7
2 clear all;
3 clc;
4
5 // Given data
6 T_b0 = 543; //
   Temperature of inlet coolant in F
7 w = 3148; // Coolant
   rate per channel in lb/hr
8
9 // 1.
10 V_f = 1.15*10^(-2); // Volume
   of fueled portion in ft^2

```

```

11 // Using the result of Example 8.3
12 q_max = 4.66*10^7; // Maximum
    heat flux at the center of the rod in Btu/hr-ft
    ^3
13 // From Table IV.3
14 c_p = 1.3; //
    Specific heat at constant pressure in Btu/lb-F
15 // Calculation
16 T_bmax = T_b0+((2*q_max*V_f)/(%pi*w*c_p));
17 // Result
18 printf("\n Exit temperature of the coolant = %d F \
    n", T_bmax);
19
20 // 2.
21 // Using the data of Example 8.6
22 h = 7500; // Heat
    transfer coefficient in Btu/hr-ft^2-F
23 // Using the data of Example 8.5
24 d = 0.42/12; //
    Diameter of the fuel rod in feet(ft)
25 a = d/2; // Radius
    of the fuel rod in ft
26 b = 0.024/12; //
    Thickness of Zircaloy-4 clad in ft
27 H = 12; // Length
    of fuel rod in ft
28 A = 2*pi*(a+b)*H; // Area of
    the assumed cylinder in ft^2
29 R_h = 1/(h*A); //
    Convective resistance in F-hour/Btu
30 alpha = pi*w*c_p*R_h; // A
    parameter
31
32 // Using the result from Example 8.5
33 R_f = 6.03*10^(-3); // Thermal
    resistance of fuel
34 R_c = 1.43*10^(-4); // Thermal
    resistance of cladding

```

```

35 R = R_f+R_c+R_h;           // Total
    resistance
36 bet = %pi*w*c_p*R;       // A
    parameter denoted by 'beta'
37 // Calculation
38 T_cmax = T_b0+((q_max*V_f*R_h)*((1+sqrt(1+alpha^2))/
    alpha));
39 T_mmax = T_b0+((q_max*V_f*R)*((1+sqrt(1+bet^2))/bet)
    );
40 // Result
41 printf("\n Maximum temperature of cladding and fuel
    = %d F and %d F respectively. \n",ceil(T_cmax),
    T_mmax);

```

Scilab code Exa 8.8 Reynolds number

```

1 // Example 8.8
2 clear all;
3 clc;
4
5 // Given data
6 d = 0.42;           //
    Diameter of the fuel rod in inches
7 b = 0.024;         //
    Thickness of Zircaloy-4 clad in inches
8 v = 15.6*3600;     // Speed
    of fluid in feet/hour
9 a = (d/2)+b;       // Radius
    of fuel rods in inches
10 P = 2000;         //
    Pressure of water in psi
11 T = 600;          // Water
    temperature in F
12 // Using the data from example 8.5
13 s = 0.6;          // Pitch

```

```

    of square array in inches
14 D_e = 2*((s^2-(%pi*a^2))/(%pi*a));           //
    Equivalent diameter in inches
15 // Converting the units in terms of feet
16 D_e = D_e/12;
17 // Using the Table IV.3 at given T and P value
18 rho = 42.9;                                 // Density
    of fluid in ft/hr
19 mu = 0.212;                                 //
    Viscosity of fluid in lb/hr-ft
20 // Calculation
21 Re = (D_e*v*rho)/mu;
22 // Result
23 printf(" \n Reynolds number = %d \n",Re);
24 if Re >= 10000 then
25     printf(" \n As the Reynolds number is greater
        than 10000, the flow is turbulent. \n");
26 end
27 // The value is different as compared to the
    textbook value. This is due to approximation of
    Reynolds number in the textbook.

```

Scilab code Exa 8.9 Heat transfer coefficient

```

1 // Example 8.9
2 clear all;
3 clc;
4
5 // Using the data from Example 8.8
6 s = 0.6;                                     // Pitch
    of square lattice in inches
7 d = 0.42;                                   //
    Diameter of the fuel rod in inches
8 b = 0.024;                                  //
    Thickness of Zircaloy-4 clad in inches

```

```

9 a = (d/2)+b; // Radius
    of fuel rods in inches
10 D_e = 0.0427; //
    Equivalent diameter in feet
11 Re = 484329; //
    Reynolds number
12 PD = s/(2*a); // The
    ratio of pitch to diameter of fuel rod
13 // For a square lattice
14 C = 0.042*PD-0.024; // A
    constant

15
16 // According to Table IV.3
17 c_p = 1.45; //
    Specific heat at constant pressure in Btu/lb-F
18 mu = 0.212; //
    Viscosity of fluid in lb/hr-ft
19 k = 0.296; //
    Conductivity of fluid in Btu/hr-ft F
20 Pr=(c_p*mu)/k; // Prandtl
    number
21 // The constants are assumed as
22 m = 0.8;
23 n = 1/3;
24 // Calculation
25 h = C*(k/D_e)*(Re)^m*Pr^n;
26 // Result
27 printf("\n Heat transfer coefficient = %d Btu/hr-ft
    ^2-F\n",h);
28 // The value is different as compared to the
    textbook value. This is due to approximation of
    Reynolds number in the textbook and in this
    problem actual value is considered.

```

Scilab code Exa 8.10 Boiling crisis


```

1 // Example 8.10
2 clear all;
3 clc;
4
5 // Using the data from Example 8.3 to 8.8
6 P = 2000; //
   Pressure in psi
7 v = 15.6; // Coolant
   velocity in ft/sec
8 D_e = 0.0427; //
   Equivalent diameter in ft
9 d = 0.42; //
   Diameter of the fuel rod in inches
10 b = 0.024; //
   Thickness of Zircaloy-4 clad in inches
11 a = (d/2)+b; // Radius
   of fuel rods in inches
12 T_b = 600; // Bulk
   temeperature in F
13
14 // 1.
15 // Using Bernath correlation
16 // Calculation
17 T_wc = 102.6*log(P)-((97.2*P)/(P+15))-(0.45*v)+32;
18 // Result
19 printf(" \n Cladding temeperature = %d F\n",T_wc);
20
21 // 2.
22 D_i = (2*pi*a)/(pi*12); // Heated
   perimeter is (2*pi*a)/12 in feet
23 // Calculation
24 h_c = 10890*((D_e)/(D_e+D_i))+((48*v)/D_e^0.6);
25 // Result
26 printf(" \n Heat transfer coefficient = %d Btu/hr-ft
   ^2-F\n",h_c);
27
28 // 3.
29 // Calculation

```

```

30 q_c = h_c*(T_wc-T_b);
31 // Result
32 printf(" \n Critical heat flux = %.2E Btu/hr-ft^2\n"
        ,q_c);
33 // In the textbook, the unit of critical heat flux
    is wrong.

```

Scilab code Exa 8.11 Engineering sub factor

```

1 // Example 8.11
2 clear all;
3 clc;
4
5 // Given data
6 m_lbar = 0.457; // Average
    linear density of UO2 in lb/ft
7 sigma = 0.0122; // Standard
    deviation of set of measured linear densities of
    UO2
8 // Calculation
9 F_Eml = 1+(3*sigma)/m_lbar;
10 printf(" \n Engineering subfactor for the amount of
    fissile material = %.2f \n",F_Eml);

```

Scilab code Exa 8.12 Fuel rod calculation

```

1 // Example 8.12
2 clear all;
3 clc;
4
5 q_max = 539000; //
    Maximum heat flux Btu/hr-ft^2

```

```

6 F = 2.8; // Hot
    channel factor
7 P = 3000; //
    Reactor thermal power in MW
8 // Expressing in Btu/hr
9 // According to Table 1.9, 1 kW = 3412 Btu/hr
10 P = P*3.412*10^6; //
    Reactor thermal power in Btu/hr
11 l = 12; //
    Length of fuel rod in ft
12 d = 0.5/12; //
    Diameter of fuel rod in ft
13 r = d/2; //
    Radius of fuel rod in ft
14
15 // 1.
16 q_av = q_max/F;
17 // Calculation
18 A = P/q_av;
19 // Result
20 printf(" \n Total heat transfer area = %d ft^2\n",A)
    ;
21
22 // 2.
23 A_one = 2*pi*r; // The
    total surface area of one fuel rod
24 // Calculation
25 n = A/A_one;
26 // Result
27 printf(" \n Number of fuel rods = %d \n",n);
28 // The value is different as compared to the
    textbook value. This is due to approximation of
    total heat transfer area in the textbook and in
    this problem actual value is considered. As total
    heat transfer area is further used to calculate
    number of fuel rods, therefore the difference in
    value exists.

```

Chapter 9

Radiation Protection

Scilab code Exa 9.1 Gamma ray interaction

```
1 // Example 9.1
2 clear all;
3 clc;
4
5 // Given data
6 e = 1.6*10^(-19); //
   Electronic charge in coulomb(coul)
7 X = 1*10^(-3)/3600; //
   Exposure rate in terms of R/sec
8 // According to the definition of Roentgen, 1 R =
   2.58*10^(-7) coul/g
9 R = 2.58*10^(-7);
10 // From standard table
11 // There is 0.001293 g of air per 1 cm^3 at 1
   atmospheric pressure at 0 C
12 density_air = 0.001293;
13 // Calculation
14 IR = (X*R*density_air)/e;
15 // Result
16 printf("\n Rate of ions produced from gamma ray
   interaction = %d ions/cm^3-sec",IR);
```

Scilab code Exa 9.2 Absorbed dose rate and dose equivalent rate

```
1 // Example 9.2
2 clear all;
3 clc;
4
5 // According to the definition of radiation absorbed
   dose(rad), 1 rad/sec = 100 ergs/g-sec
6 // Given data
7 D = 5*10^(-3)/100; //
   Absorbed dose in terms of rad/sec
8 // Expressing absorbed dose rate in SI units
9 // 1 Gray(Gy) = 100 rad
10 D_dot = D*3600/100;
11 // Using data from Table 9.2
12 Q = 1; //
   Quality factor for gamma rays for tissue
13 // Calculation
14 H_dot = D_dot*Q;
15 printf(' \n Absorbed dose rate in a tissue = %.1f
   mGy/hr \n',D_dot*1000);
16 printf(' \n Dose equivalent rate in a tissue = %.1f
   mSv/hr \n',H_dot*1000);
```

Scilab code Exa 9.3 Acute radiation exposure

```
1 // Example 9.3
2 clear all;
3 clc;
4
5 // Given data
```

```

6 H = 25; //
  Equivalent dose in rem
7 age = 30; // Age
  of worker in years
8 exp_age = 77; //
  Average age upto which a person lives in years
9 // Using data from Table 9.6
10 // Bone cancer
11 rc_bone = 0.2; //
  Risk coefficient per 10^6 rem/year
12 lp_bone = 10; //
  Latent period in years
13 // Probability of dying from bone cancer
14 p_bone=(H*rc_bone*(exp_age-(lp_bone+age)))/10^6;
15
16 // Breast cancer
17 rc_breast = 1.5; //
  Risk coefficient per 10^6 rem/year
18 lp_breast = 15; //
  Latent period in years
19 // Probability of dying from breast cancer
20 p_breast = (H*rc_breast*(exp_age-(lp_breast+age))
  /10^6;
21
22 // Leukemia
23 rc_leukemia = 1; //
  Risk coefficient per 10^6 rem/year
24 lp_leukemia = 2; //
  Latent period in years
25 // Probability of dying from leukemia
26 p_leukemia = (H*rc_leukemia*(exp_age-(lp_leukemia+
  age)))/10^6;
27
28 // Lung cancer
29 rc_lung = 1; //
  Risk coefficient per 10^6 rem/year
30 lp_lung = 15; //
  Latent period in years

```

```

31 // Probability of dying from lung cancer
32 p_lung = (H*rc_lung*(exp_age-(lp_lung+age)))/10^6;
33
34 // Pancreatic cancer
35 rc_pancreas = 0.2; //
    Risk coefficient per 10^6 rem/year
36 lp_pancreas = 15; //
    Latent period in years
37 // Probability of dying from Pancreatic cancer
38 p_pancreas = (H*rc_pancreas*(exp_age-(lp_pancreas+
    age)))/10^6;
39
40 // Stomach cancer
41 rc_stomach = 0.6; //
    Risk coefficient per 10^6 rem/year
42 lp_stomach = 15; //
    Latent period in years
43 // Probability of dying from stomach cancer
44 p_stomach = (H*rc_stomach*(exp_age-(lp_stomach+age))
    )/10^6;
45
46 // Rest of alimentary cancer
47 rc_ali = 0.2; //
    Risk coefficient per 10^6 rem/year
48 lp_ali = 15; //
    Latent period in years
49 // Probability of dying from rest of alimentary
    cancer
50 p_ali = (H*rc_ali*(exp_age-(lp_ali+age)))/10^6;
51
52 // Thyroid cancer
53 rc_thy = 0.43; //
    Risk coefficient per 10^6 rem/year
54 lp_thy = 10; //
    Latent period in years
55 // Probability of dying from thyroid cancer
56 p_thy = (H*rc_thy*(exp_age-(lp_thy+age)))/10^6;
57

```

```

58 // All other type of cancer
59 rc_other = 1; //
    Risk coefficient per 10^6 rem/year
60 lp_other = 15; //
    Latent period in years
61 // Probability of dying from all other type of
    cancer
62 p_other = (H*rc_other*(exp_age-(lp_other+age)))
    /10^6;
63
64 // Calculation
65 p = p_bone+p_breast+p_leukemia+p_lung+p_pancreas+
    p_stomach+p_alo+p_thy+p_other;
66 // Result
67 printf('\n Probability that the worker will die from
    cancer = %.1E \n',p);
68
69 // The value obtained is different from the value
    given in the textbook. This is because of
    approximation of individual probabilities in the
    textbook.

```

Scilab code Exa 9.4 Acute radiation exposure

```

1 // Example 9.4
2 clear all;
3 clc;
4 H = 1; // Equivalent
    dos in rem
5 n = 10^6; // Population
6 // Given data
7
8 // Using the data of number of expected deaths of
    leukemia per 10^6 people from Table 9.9
9 // In utero age group

```



```

10 frac_utero = 0.011;           // Fraction of
    population
11 riskyr_utero = 10;           // Risk years
12 riskcoef_utero = 15;        // Risk
    coefficient
13 // Number of deaths in utero is given by
14 deaths_utero = frac_utero*riskyr_utero*
    riskcoef_utero;
15
16 // In 0–0.99 age group
17 frac_0_099 = 0.014;        // Fraction of
    population
18 riskyr_0_099 = 25;          // Risk years
19 riskcoef_0_099 = 2;         // Risk
    coefficient
20 // Number of deaths in 0–0.99 age group is given by
21 deaths_0_099 = frac_0_099*riskyr_0_099*
    riskcoef_0_099;
22
23 // In 1–10 age group
24 frac_1_10 = 0.146;         // Fraction of
    population
25 riskyr_1_10 = 25;          // Risk years
26 riskcoef_1_10 = 2;         // Risk
    coefficient
27 // Number of deaths in 1–10 age group is given by
28 deaths_1_10=frac_1_10*riskyr_1_10*riskcoef_1_10;
29
30 // In 11–20 age group
31 frac_11_20 = 0.196;        // Fraction of
    population
32 riskyr_11_20 = 25;          // Risk years
33 riskcoef_11_20 = 1;        // Risk
    coefficient
34 // Number of deaths in 11–20 age group is given by
35 deaths_11_20=frac_11_20*riskyr_11_20*riskcoef_11_20;
36
37 // In 21–30 age group

```

```

38 frac_21_30 = 0.164;           // Fraction of
    population
39 riskyr_21_30 = 25;           // Risk years
40 riskcoef_21_30 = 1;         // Risk
    coefficient
41 // Number of deaths in 21–30 age group is given by
42 deaths_21_30=frac_21_30*riskyr_21_30*riskcoef_21_30;
43
44 // In 31–40 age group
45 frac_31_40 = 0.118;         // Fraction of
    population
46 riskyr_31_40 = 25;           // Risk years
47 riskcoef_31_40 = 1;         // Risk
    coefficient
48 // Number of deaths in 31–40 age group is given by
49 deaths_31_40=frac_31_40*riskyr_31_40*riskcoef_31_40;
50
51 // In 41–50 age group
52 frac_41_50 = 0.109;         // Fraction of
    population
53 riskyr_41_50 = 25;           // Risk years
54 riskcoef_41_50 = 1;         // Risk
    coefficient
55 // Number of deaths in 41–50 age group is given by
56 deaths_41_50 = frac_41_50*riskyr_41_50*
    riskcoef_41_50;
57
58 // In 51–60 age group
59 frac_51_60 = 0.104;         // Fraction of
    population
60 riskyr_51_60 = 22.5;         // Risk years
61 riskcoef_51_60 = 1;         // Risk
    coefficient
62 // Number of deaths in 51–50 age group is given by
63 deaths_51_60 = frac_51_60*riskyr_51_60*
    riskcoef_51_60;
64
65 // In 61–70 age group

```

```

66 frac_61_70 = 0.08;
67 riskyr_61_70 = 15.1;
68 riskcoef_61_70 = 1; // Risk
   coefficient
69 // Number of deaths in 61–70 age group is given by
70 deaths_61_70=frac_61_70*riskyr_61_70*riskcoef_61_70;
71
72 // In 71–80 age group
73 frac_71_80 = 0.044; // Fraction of
   population
74 riskyr_71_80 = 9.1; // Risk years
75 riskcoef_71_80 = 1; // Risk
   coefficient
76 // Number of deaths in 71–80 age group is given by
77 deaths_71_80 = frac_71_80*riskyr_71_80*
   riskcoef_71_80;
78
79 // Age greater than 80
80 frac_80 = 0.02; // Fraction of
   population
81 riskyr_80 = 4.5; // Risk years
82 riskcoef_80 = 1; // Risk
   coefficient
83 // Number of deaths with age greater than 80 years
   is given by
84 deaths_80=frac_80*riskyr_80*riskcoef_80;
85
86 // Calculation
87 total_deaths = deaths_utero+deaths_0_099+deaths_1_10
   +deaths_11_20+deaths_21_30+deaths_31_40+
   deaths_41_50+deaths_51_60+deaths_61_70+
   deaths_71_80+deaths_80;
88 // Result
89 printf(" \n Number of cases or deaths expected from
   leukemia = %.1f \n",total_deaths);

```

Scilab code Exa 9.5 Chronic radiation exposure

```
1 // Example 9.5
2 clear all;
3 clc;
4
5 // Given data
6 H_year = 5; //
   Equiavelnt dose per year in rem
7 start_age = 18; //
   Initial age of the worker in years
8 ret_age = 68; //
   Retirement age of the worker in years
9 // Using data from Table 9.6 with respect to Bone
   cancer
10 latent_period = 10; //
   Latent period in years
11 plateau_period = 30; //
   Plateau period in years
12 rc_bone = 0.2; //
   Risk coefficient per 10^6 rem/year
13
14 n = ret_age-(start_age+latent_period); //
   Number of years of accumulated dose
15 H = n*H_year; //
   Total equivalent dose in rem
16 // Calculation
17 p_bone = (H*rc_bone*plateau_period)/10^6;
18 // Result
19 printf(" \n The probability of dying from bone
   cancer = %.1E \n",p_bone);
```

Scilab code Exa 9.6 External exposure due to gamma rays

```
1 // Example 9.6
2 clear all;
3 clc;
4
5 // Given data
6 E =2 ; //
   Energy of gamma radiation in MeV
7 X_dot = 1; //
   Exposure rate in mR/hour
8 // Using the data from Table II.5
9 // Let mu_a/rho of air at 2 Mev be denoted as mu_rho
10 mu_rho = 0.0238; //
   Ratio of absorption coefficient to sensity of air
   in cm^2/g
11 // Calculation
12 I = X_dot/(E*mu_rho*0.0659);
13 printf(" \n The beam intensity of gamma radiation
   required = %d gamma rays/cm^2-sec \n",ceil(I));
```

Scilab code Exa 9.7 External exposure due to X rays

```
1 // Example 9.7
2 clear all;
3 clc;
4
5 // Given data
6 phi = 2.4*10^5; //
   Flux in x-rays/cm^2-sec
7 // From Figure 9.9
8 // To receive an exposure rate of 1 mR/hr at 50 keV,
   the flux is 8*10^3 x-rays/cm^2-sec
9 phi_eq = 8*10^3; //
   Equivalent flux in x-rays/cm^2-sec
```

```

10 X_dot_eq = 1; //
    Equivalent Exposure rate in mR/hr
11 X_dot = (phi*X_dot_eq)/phi_eq; //
    Exposure rate of the operator in mR/hr
12 // From Figure 9.10 at 50 kV energy, the energy
    dependent function is
13 f_bone = 3.3;
14 f_muscle = 0.93;
15 f_fat = 0.9;
16 // Using data from Table 9.2
17 Q = 1; //
    Quality factor for x-rays
18 // Calculation
19 D_dot_bone = X_dot*f_bone*Q; //
    Dose equivalent rate in bone
20 D_dot_muscle = X_dot*f_muscle*Q; //
    Dose equivalent rate in muscle
21 D_dot_fat = X_dot*f_fat*Q; //
    Dose equivalent rate in fat
22 // Result
23 printf(" \n Dose equivalent rate in bone = %d mrem/
    hour \n", ceil(D_dot_bone));
24 printf(" \n Dose equivalent rate in muscle = %d mrem
    /hour \n", ceil(D_dot_muscle));
25 printf(" \n Dose equivalent rate in fat = %d mrem/
    hour \n", ceil(D_dot_fat));

```

Scilab code Exa 9.8 External exposure due to neutrons

```

1 // Example 9.8
2 clear all;
3 clc;
4
5 // Given data
6 phi_n = 20; // Given

```

```

    neutron flux in neutrons/cm2-sec
7 // From Figure 9.12
8 // To receive an dose equivalent rate of 1 mrem/hr,
    the fast neutron flux is 7 neutrons/cm2-sec
9 phi_n_eq = 7;
10 D_dot_eq = 1;
11 D_dot_n = (phi_n*D_dot_eq)/phi_n_eq; // Dose
    rate due to fast neutron flux in mrem/hr
12 phi_th = 300; // Given
    thermal flux in neutrons/cm2-sec
13 // From Figure 9.12
14 // To receive an dose equivalent rate of 1 mrem/hr,
    the thermal flux is 260 neutrons/cm2-sec
15 phi_th_eq = 260;
16 D_dot_th = (phi_th*D_dot_eq)/phi_th_eq; // Dose
    rate due to thermal neutron flux in mrem/hr
17 D_dot = D_dot_n+D_dot_th; // Total
    dose rate in mrem/hr
18 printf("\n The permitted weekly dose is 100 mrem \n"
    );
19 D_dot_perm = 100;
20 // Calculation
21 t = D_dot_perm/D_dot;
22 printf(" \n The time of exposure upto a safe level =
    %.1f hour \n",t);
23 // The answer given in the textbook is wrong. This
    is because of wrong computation of total dose
    rate

```

Scilab code Exa 9.9 External exposure due to neutrons

```

1 // Example 9.9
2 clear all;
3 clc;
4

```

```

5 // Given data
6 fluence = 10^8; //
   Given fluence neutrons/cm^2
7 // From Figure 9.12
8 // To receive an dose equivalent rate of 1 mrem/hr,
   the fast neutron flux is 7 neutrons/cm^2-sec
9 phi_eq = 7; //
   Equivalent flux in neutrons/cm^2-sec
10 D_eq = 1; //
   Equivalent dose rate in mrem/hr
11 // 1 hour = 3600 seconds
12 fluence_eq = phi_eq*3600; //
   Equivalent fluence in neutrons/cm^2
13 // Calculation
14 D = (fluence*D_eq)/fluence_eq;
15 // Result
16 printf("\n Dose received due to exposure of
   accelerator source = %d mrem \n",D);
17 // The answer given in textbook is approximated to a
   nearest value.

```

Scilab code Exa 9.10 Internal exposure due to radionuclide

```

1 // Example 9.10
2 clear all;
3 clc;
4
5 // Given data
6 M = 20; // Mass of
   organ in grams
7
8 // a)
9 // Using the data from Table 9.15
10 T_12 = 8.04; //
   Radiological half life of Iodine-131 in days

```



```

11 T_12_b = 138; // Biological
    half life of Iodine-131 in days
12 lambda = 0.693/T_12; //
    Radiological decay constant of Iodine-131 in days
    ^-1
13 lambda_b = 0.693/T_12_b; // Biological
    decay constant of Iodine-131 in days^-1
14 lambda_e = lambda+lambda_b; // Equivalent
    decay constant in days^-1
15 // Using the data from Table 9.15
16 zeta = 0.23; // Effective
    energy equivalent in MeV
17 q = 0.23; // The
    fraction of Iodine-131 that goes by inhalation
18 // Calculation
19 DCF = (51.1*zeta*q)/(M*lambda_e);
20 // Result
21 printf(" \n The dose commitment factor by inhalation
    = %.2f rem/ucurie \n",DCF);
22
23 // b)
24 breathing_rate = 2.32*10^(-4); // Normal
    breathing rate in m^3/sec
25 time = 2*3600; // Time of
    radiation exposure in seconds
26 I_conc = 2*10^(-9); // Iodine-131
    concentration
27 C0 = breathing_rate*time*I_conc; // Total
    intake of Iodine-131 by inhalation
28 // Calculation
29 H = C0*(DCF*10^6); // Using DCF
    in micro-curie
30 // Result
31 printf(" \n The dose commitment to thyroid = %.2E
    rem = %.2f mrem \n",H,H*1000);

```

Scilab code Exa 9.11 Maximum Permissible Concentration

```
1 // Example 9.11
2 clear all;
3 clc;
4
5 // Given data
6 V_W = 2200; // Volume of
   water intake in terms of cm3/day
7 // 1 litre = 1000 gram(g)
8 M = 43*1000; // Mass of
   water present in standard man according to
   standards
9 // Using the data from Table 9.13
10 MPD = 0.1/7; // Maximum
   Permissible Dose (MPD) in rem/day
11 // Using the data from Table 9.15
12 zeta = 0.01; // Effective
   energy equivalent in MeV
13 q = 1; // The
   fraction of Tritium that goes inside by ingestion
14 T_b = 11.9; // Biological
   Half life of Tritium in years
15 lambda_b = 0.693/T_b; // Biological
   decay constant of Tritium in years-1
16
17 // As biological and radiological half lives are
   less than 50 year intake period, the exponential
   term (exp(-lambda_e*50)) is neglected
18 // Maximum Permissible Concentration(MPC) for a 7
   day or 168 hour week tritium dose
19 MPC_w_168 = (lambda_b*M*MPD)/(51.1*V_W*zeta*q);
20 printf("\n Maximum Permissible Concentration(MPC)
   for a 7 day or 168 hour week tritium dose for
```

```

    occupational purpose = %.2f uCi/cm^3 \n",
    MPC_w_168);
21 // The exposure at work is doubled for 40 hour week
    as compared to 168 hour week
22 // For 40 hour week, with work of 5 days out of 7
    day week according to a study
23 MPC_w_40 = MPC_w_168*2*(7/5);
24 printf("\n Maximum Permissible Concentration(MPC)
    for a 40 hour week tritium dose for occupational
    purpose = %.3f uCi/cm^3 \n",MPC_w_40);
25
26 // By analyzing the data of Table 9.13
27 // The whole body dose of general public is one
    tenth of the occupational purpose.
28 MPC_w_168_gp = MPC_w_168*0.1;
29 printf("\n Maximum Permissible Concentration(MPC)
    for a 7 day or 168 hour week tritium dose for
    general public = %.3f uCi/cm^3 \n",MPC_w_168_gp);
30 // The answer of Maximum Permissible Concentration(
    MPC) for a 168 hour week tritium dose for general
    public is given wrong in the textbook.

```

Scilab code Exa 9.12 Acute radiation exposure

```

1 // Example 9.12
2 clear all;
3 clc;
4
5 // Given data
6 no_home = 10^6; //
    Number of houses
7 no_resident = 4; //
    Number of residents in a home
8 total_time = 50; //
    Number of years the analysis is carried out

```

```

9 radon_concn_old = 1; //
    Radon concentration in older uninsulated homes in
    terms of pCi/l
10 radon_concn_new = 6; //
    Radon concentration in modern insulated homes in
    terms of pCi/l
11 time = 3500; //
    Time spent in home by a person per year in hours
12 eq_concn = 0.5; //
    Equilibrium concentration of 50%
13 // 1 year = 24*365 hours
14 X_increased = eq_concn*(radon_concn_new-
    radon_concn_old)*(time/(24*365)); // The
    increased exposure of radon per person
15
16 // Using the data of Radon risk assessment of United
    States of America
17 // There are nearly 100 cases of cancer per 10^6
    persons at 1 pCi-year dose.
18 // Calculation
19 no_cancer = (no_home*no_resident)*total_time
    *(100/10^6)*X_increased;
20 // Result
21 printf("\n Number of additional cases of cancer from
    insulation of home = %d \n",no_cancer);
22 // There is a slight deviation in the value given in
    the textbook. This is because of approximation
    of the number of additional cases of cancer in
    the textbook.

```

Scilab code Exa 9.13 Acute radiation exposure

```

1 // Example 9.13
2 clear all;
3 clc;

```

```
4
5 // Given data
6 H_ext = 3; //
   External dose in rem
7 H_wbL = 5; //
   Annual whole body dose limit in rem
8 // Using the data from Table 9.17
9 // Annual Limit Intake (ALI) for inhalation of
   Iodine-131 is 54uCurie (Ci)
10 ALI = 54;
11 // Calculation
12 I = ALI*(1-(H_ext/H_wbL));
13 // Result
14 printf("\n Amount of Iodine-131 intake within safety
   limits = %d uCi \n", ceil(I));
```

Chapter 10

Radiation Shielding

Scilab code Exa 10.1 Gamma ray buildup factor

```
1 // Example 10.1
2 clear all;
3 clc;
4
5 // Given data
6 E0 = 2; //
   Energy of gamma rays in MeV
7 a = 10; //
   Thickness of lead shield in cm
8 phi0 = 10^6; //
   Intensity of gamma rays in gamma-rays/cm^2-sec
9
10 // 1.
11 // Using the data from Table II.4 for E0 = 2 MeV
12 mu_rho = 0.0457; // The
   ratio of total attenuation coefficient to
   density in cm^2/g
13 // From standard data tables for lead
14 rho = 11.34; //
   Density of lead in g/cm^3
15 // Calculation
```

```

16 phi_u = phi0*exp(-(mu_rho*rho*a));
17 // Result
18 printf("\n Uncollided flux at the rear side of lead
    shield = %.2E gamma-rays/cm^2-sec \n",phi_u)
19
20 // 2.
21 // Using the data from Table 10.1 for 2 MeV of lead
    material
22 mua = mu_rho*rho*a;
23 B_4 = 2.41; //
    Buildup factor if mu*a = 4
24 B_7 = 3.36; //
    Buildup factor if mu*a = 7
25 // Using two point method of straight line for
    calculating buildup factor at mu*a
26 B_m = B_4+((mua-4)*((B_7-B_4)/(7-4)));
27 // Calculation
28 phi_b = phi_u*B_m;
29 // Result
30 printf("\n Buildup flux at the rear side of lead
    shield = %.2E gamma-rays/cm^2-sec \n",phi_b);
31
32 // 3.
33 // Using the data from Table II.5 for 2 MeV
34 mua_rho_air = 0.0238; //
    The ratio of total attenuation coefficient to
    density of air in cm^2/g
35 // Calculation
36 X_dot = 0.0659*E0*mua_rho_air*phi_b;
37 // Result
38 printf("\n Exposure rate at the rear side of lead
    shield = %.1f mR/hour \n",X_dot);

```

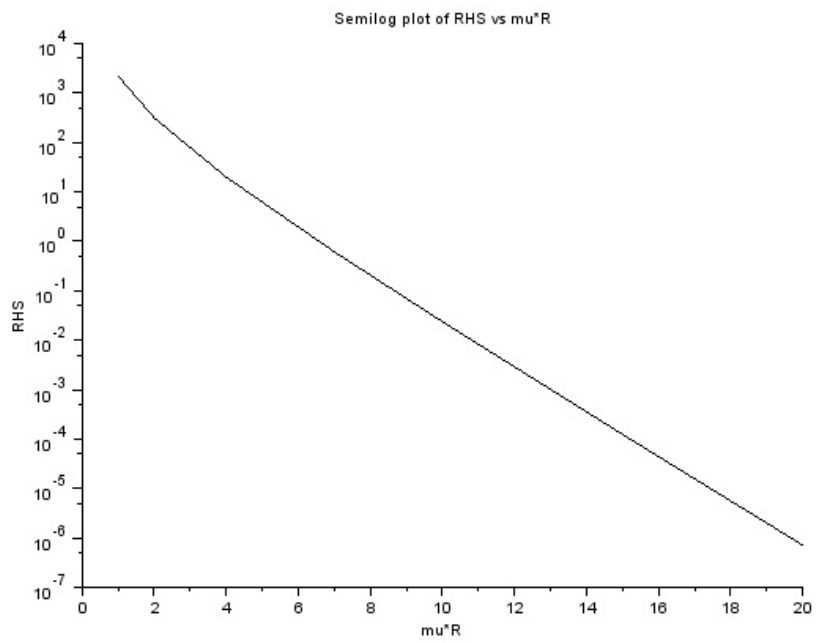


Figure 10.1: Gamma ray buildup factor

Scilab code Exa 10.2 Gamma ray buildup factor

```
1 // Example 10.2
2 clear all;
3 clc;
4
5 // Given data
6 E = 1; //
   Energy of gamma rays in MeV
7 X_dot = 1; //
   Exposure rate in mR/hour
8 phi0 = 10^8; //
   Intensity of gamma rays in gamma-rays/cm^2-sec
   from isotropic point source
9 // Using the data from Table II.5 for 1 MeV
10 mua_rho_air = 0.028; //
   The ratio of total attenuation coefficient to
   density of air in cm^2/g
11 phi_b = X_dot/(0.0659*E*mua_rho_air); //
   Buildup flux in gamma-rays/cm^2-sec
12 // Using Eq 10.14
13 printf("\n The equation to calculate radius is \n %
   .2E = %E * Bp*exp(-mu*R)/(4*pi*R^2) \n", phi_b,
   phi0);
14 // Using the data from Table II.4 for E = 1 MeV for
   Iron
15 mu_rho = 0.0595; // The
   ratio of total attenuation coefficient to
   density in cm^2/g
16 // From standard data tables for iron
17 rho = 7.864; //
   Density of iron in g/cm^3
18 mu = mu_rho*rho;
19 // On solving the right hand side of equation
20 // RHS = 3.22*10^3*Bp*exp(-mu*R)/(mu*R)^2
21 // Let mu*R = x
22 // Using the data from Table 10.2 for isotropic
   point source of 1 MeV incident on iron material
```

```

23 Bp = [1.87 2.89 5.39 10.2 16.2 28.3 42.7];
24 x = [1 2 4 7 10 15 20];
25 for i = 1:7
26     RHS(i) = (3.22*10^3*Bp(i)*exp(-x(i))/x(i)^2);
27 end
28 plot2d("nl",x(:),RHS(:));
29 xlabel("mu*R");
30 ylabel("RHS");
31 title("Semilog plot of RHS vs mu*R")
32 // From the graph
33 muR = 6.55; //
    This is the value when RHS = 1
34 // Calculation
35 R = muR/mu;
36 // Result
37 printf("\n The shield radius required = %d cm \n",
    ceil(R));

```

Scilab code Exa 10.3 Shielding of infinite planar source

```

1 // Example 10.3
2 clear all;
3 clc;
4
5 // Given data
6 E = 2; //
    Energy of gamma rays in MeV
7 X_dot = 2.5; //
    Exposure rate in mR/hour
8 phi0 = 10^9; //
    Intensity of gamma rays in gamma-rays/cm^2-sec
    from isotropic point source
9 // Using the data from Table II.5 for 1 MeV

```

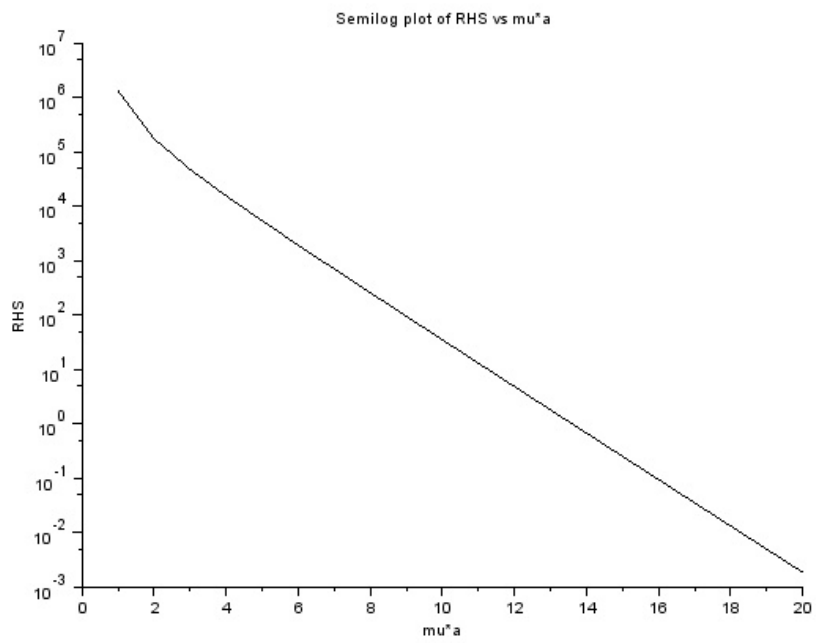


Figure 10.2: Shielding of infinite planar source

```

10 mua_rho_air = 0.0238; //
    The ratio of total attenuation coefficient to
    density of air in cm^2/g
11 phi_b = X_dot/(0.0659*E*mua_rho_air); //
    Buildup flux in gamma-rays/cm^2-sec
12
13 // From standard data tables for concrete
14 rho = 2.35; //
    Density of concrete in g/cm^3
15 // Using the data from Table 10.3 for concrete at 2
    MeV
16 A1 = 18.089;
17 A2 = 1-A1;
18 alpha1 = -0.0425;
19 alpha2 = 0.00849;
20 // Using Eq 10.26
21 printf(" \n The equation to calculate thickness is \
    \n %2E = (%E/2) *(%4.3f*E1(%4.3f*mu*a) %4.3f*E1(
    %4.3f*mu*a)) \n", phi_b, phi0, A1, (1+alpha1), A2, (1+
    alpha2));
22 // Using the data from Table II.4 for E = 1 MeV for
    concrete
23 mu_rho = 0.0445; // The
    ratio of total attenuation coefficient to
    density in cm^2/g
24 mu = mu_rho*rho;
25 // On solving the right hand side of equation
26 // RHS = 1.13*10^7*(E1(0.9575*mu*a) -0.94*E1(1.00849*
    mu*a))
27 // Let mu*a = x
28 x = 1:20
29 for i = 1:20
30     RHS(i) = 1.13*10^7*(exp(-0.9575*x(i))
        *((1/(0.9575*x(i)))+(1/(0.9575*x(i))^3))) -
        exp(-1.00849*x(i))*((1/(1.00849*x(i))
        +(1/(1.00849*x(i))^3))));
31 end
32 plot2d(" nl", x(:), RHS(:));

```

```

33 xlabel("mu*a");
34 ylabel("RHS");
35 title("Semilog plot of RHS vs mu*a")
36 // From the graph
37 mua = 13.6; //
    This is the value when RHS = 1
38 // Calculation
39 a = mua/mu;
40 // Result
41 printf("\n The concrete thickness = %d cm \n",a);

```

Scilab code Exa 10.4 Multilayered shielding

```

1 // Example 10.4
2 clear all;
3 clc;
4
5 // Given data
6 E = 6; //
    Energy of gamma rays in MeV
7 phi0 = 10^2; //
    Intensity of gamma rays in gamma-rays/cm^2-sec
    from mono-directional beam
8 x_w = 100; //
    Thickness of water in cm
9 x_Pb = 8; //
    Thickness of lead in cm
10 // Using data from Table II.4 at 6 MeV
11 mu_w = 0.0275; //
    Total attenuation coefficient of water in cm^(-1)
12 mu_Pb = 0.4944; //
    Total attenuation coefficient of lead in cm^(-1)
13
14 mua_w = x_w*mu_w; //
    Attenuation due to thickness of water

```

```

15 mua_Pb = x_Pb*mu_Pb; //
    Attenuation due to thickness of lead
16
17 // Case (a) – Water is placed before the lead
18 printf(" \n In case (a), Buildup factor is only due
    to lead measured at %.2f",mua_Pb);
19 // Using the data from Table 10.1 at 6 MeV
20 B_Pb = 1.86;
21 phi_b_a = phi0*B_Pb*exp(-(mua_w+mua_Pb));
22
23 // Using the data from Table II.5 for 6 MeV
24 mua_rho_air = 0.0172; //
    The ratio of total attenuation coefficient to
    density of air in cm^2/g
25 // Calculation
26 X_dot_a = 0.0659*E*mua_rho_air*phi_b_a;
27 // Result
28 printf("\n Exposure rate if water is placed before
    lead shield = %.2f uR/hour \n",X_dot_a*1000);
29
30 // Case (b) – Lead is placed before water
31 printf(" \n In case (b), Buildup factor is due to
    water and lead measured at %.2f and %.2f
    respectively",mua_w,mua_Pb);
32 // Using the data from Table 10.1 for water at 3.2
    MeV,, which is the minimum point of mu_Pb curve
33 B_w = 2.72;
34 B_m = B_Pb*B_w;
35 phi_b_b = phi0*B_m*exp(-(mua_w+mua_Pb));
36
37 // Calculation
38 X_dot_b = 0.0659*E*mua_rho_air*phi_b_b;
39 // Result
40 printf("\n Exposure rate if lead is placed before
    water = %.2f uR/hour \n",X_dot_b*1000);
41 // The answer given in the textbook is wrong. This
    is because the intensity of gamma rays is wrongly
    taken for calculation.

```

Scilab code Exa 10.5 Removal cross section

```
1 // Example 10.5
2 clear all;
3 clc;
4
5 // Given data
6 fission_density = 4*10^7; //
   Fission density in fissions/cm^2-sec
7 // 1 inches = 2.54 cm
8 d = 28*2.54; //
   Diameter of plate in cm
9 R = d/2; //
   Radius of plate in cm
10 v = 2.42; //
   Number of fission neutrons emitted per fission
11 x = 75; //
   Distance of point from center of plate in cm
12 // Using the data from Table 10.4 for removal
   macroscopic cross section of water
13 sigma_RW = 0.103; //
   Removal macroscopic cross section of water in cm
   ^-1
14 S = v*fission_density; //
   Strength of neutron source in terms of neutrons/
   cm^2-sec
15 A = 0.12; // A
   constant
16 // From Figure 10.19
17 tan_theta = R/x;
18 theta = atan(R/x);
19 sec_theta = sec(theta);
20
21 // 1.
```

```

22 x11 = sigma_RW*x; //
    Exponential integral function
23 x21 = sigma_RW*x*sec_theta; //
    Exponential integral function
24 // Let the upper limit of integral be 20
25 x_limit = 20;
26 function y=f(x),y=exp(-x)/x, endfunction
27 E1_x11 = intg(x11,x_limit,f);
28 E1_x21 = intg(x21,x_limit,f);
29 // Calculation
30 phi_1 = S*A/2*(E1_x11-E1_x21);
31 // Result
32 printf(" \n The fast flux without iron shield = %d
    neutrons/cm^2-sec \n",phi_1);
33
34 // 2. Iron slab is inserted in front of the fission
    plate
35 // Using the data from Table 10.4 for removal
    macroscopic cross section of iron
36 sigma_R = 0.168; //
    Removal macroscopic cross section of iron in cm
    ^-1
37 t = 3*2.54; //
    Thickness of iron slab in cm
38 // Now the analysis is similar to multi layered
    shielding
39 x12 = sigma_RW*x+sigma_R*t; //
    Exponential integral function
40 x22 = sigma_RW*x*sec_theta+sigma_R*t*sec_theta; //
    Exponential integral function
41 // Let the upper limit of integral be 20
42 x_limit = 20;
43 function y=f(x),y=exp(-x)/x, endfunction
44 E1_x12 = intg(x12,x_limit,f);
45 E1_x22 = intg(x22,x_limit,f);
46 // Calculation
47 phi_2 = S*A/2*(E1_x12-E1_x22);
48 // Result

```



```

49 printf("\n The fast flux with iron shield = %.1f
        neutrons/cm^2-sec \n",phi_2);

```

Scilab code Exa 10.6 Removal cross section

```

1 // Example 10.6
2 clear all;
3 clc;
4
5 // Given data
6 // Assuming average energy produced per fission
  reaction is 200 MeV
7 P = 250*10^3; //
  Power of research reactor in Watts
8 P_fission = 200*10^6*1.6*10^(-19); //
  Energy produced in a fission reaction in terms of
  joule
9 f = 0.75; //
  Metal volume fraction
10 // In this problem, both reflector and shield act as
  a composite shield
11 a = 150+15; //
  Net shield distance in cm
12 // 1 litre = 1000 grams
13 V = 32*1000; //
  Core volume in gram
14
15 fission_density = (P/P_fission)*(1/V);
16 v = 2.42; //
  Number of fission neutrons emitted per fission
17 S = fission_density*v; //
  Neutron source strength in neutrons/cm^3-sec
18 // Assuming spherical shape
19 // Volume of sphere = (4/3)*pi*(radius)^3
20 R = ((3*V)/(4*pi))^(1/3);

```

```

21 // Using the data from Table 10.4 for removal
    macroscopic cross section
22 sigma_R = 0.174; //
    Removal macroscopic cross section of uranium in
    cm-1
23 sigma_RW = 0.103; //
    Removal macroscopic cross section of water in cm
    -1
24 A = 0.12; // A
    constant
25 alpha = ((1-f)*sigma_RW)+(f*sigma_R); // A
    parameter
26 // Calculation
27 theta = (S*A/(4*alpha))*(ceil(R)/(ceil(R)+a))^2*exp
    (-sigma_RW*a)*(1-exp(-2*alpha*ceil(R)));
28 // Converting into equivalent dose rate by referring
    Figure 9.12
29 // Fast neutron flux of 6.8 neutrons/cm2-sec is
    equivalent to 1 mrem/hr
30 H_dot = theta/6.8;
31 // Result
32 printf("\n Fast neutron dose rate near the surface
    of the shield = %.1f mrem/hour \n ",H_dot);

```

Scilab code Exa 10.7 Dose equivalent rate and thickness

```

1 // Example 10.7
2 clear all;
3 clc;
4
5 // Given data
6 E = 14; //
    Energy of neutrons in MeV
7 phi0 = 109; //
    Intensity of neutrons in neutrons/cm2-sec from

```

```

    isotropic point source
8 // 1 feet = 30.48 cm
9 d = 10*30.48; //
    Distance of concrete wall from a point source in
    cm
10 // As Intensity obeys inverse square law
11 I = phi0/(4*pi*d^2); //
    Intensity of neutron beam in terms of neutrons/cm
    ^2-sec
12 H_dot = 1; //
    The required dose equivalent rate in mrem/hour
13 // From Figure 10.23(b)
14 H0_dot = H_dot/I; //
    The dose equivalent rate
15 // Result
16 printf("\n The reduced dose equivalent rate due to
    concrete wall is = %.2E mrem/hr \n",H0_dot);
17 // By looking into Figure 10.23(b) on thickness axis
18 printf("\n The concrete slab thickness is = 70 cm \
    n");

```

Scilab code Exa 10.8 Shielding of prompt fission gamma rays

```

1 // Example 10.8
2 clear all;
3 clc;
4
5 // Given data
6 R = 7*30.48; //
    Distance of core from the center of shield in cm
7 // Assuming average energy produced per fission
    reaction is 200 MeV
8 P = 10; //
    Power of teaching reactor in Watts
9 P_fission = 200*10^6*1.6*10^(-19); //

```

```

    Energy produced in a fission reaction in terms of
    joule
10 fission_rate = P/P_fission;           //
    Number of fission reactions
11
12 // By assuming that the gamma rays are of equal
    energy of 1 MeV (Group 1) and looking into Table
    10.5
13 E1 = 1;                               //
    Energy of gamma rays in MeV (Assumed)
14 chi_pn1 = 5.2;                         //
    Number of prompt gamma rays emitted per fission
    with energy 2 MeV
15 S1 = chi_pn1*fission_rate;            //
    Source strength in gamma rays/sec
16 // Using the data from Table II.4 for E = 1 MeV for
    water
17 mu1 = 0.0996;                          //
    Mass attenuation coefficient at 1 MeV in cm-1
18 printf("\n Buildup factor is due to water measured
    at %.2f",mu1*R);
19 // Using the data from Table 10.2 at 1 MeV
20 B_p1 = 373;
21 phi_b1 = (S1/(4*pi*R^2))*B_p1*exp(-mu1*R); //
    Buildup flux
22 // Using the data from Table II.5 for 1 MeV
23 mua_rho_air1 = 0.028;                  //
    The ratio of total attenuation coefficient to
    density of air in cm2/g
24 // Calculation
25 X_dot1 = 0.0659*E1*mua_rho_air1*phi_b1;
26 printf("\n Exposure rate due to group 1 = %.4f mR/
    hour \n",X_dot1);
27
28 // By assuming that the gamma rays are of equal
    energy of 2 MeV (Group 2) and looking into Table
    10.5
29 E2 = 2;                               //

```

```

    Energy of gamma rays in MeV (Assumed)
30 chi_pn2 = 1.8; //
    Number of prompt gamma rays emitted per fission
    with energy 2 MeV
31 S2 = chi_pn2*fission_rate; //
    Source strength in gamma rays/sec
32 // Using the data from Table II.4 for E = 2 MeV for
    water
33 mu2 = 0.0493; //
    Mass attenuation coefficient at 2 MeV in cm-1
34 printf("\n Buildup factor is due to water measured
    at %.2 f",mu2*R);
35 // Using the data from Table 10.2 at 2 MeV
36 B_p2 = 13.1;
37 phi_b2 = (S2/(4*pi*R^2))*B_p2*exp(-mu2*R); //
    Buildup flux
38 // Using the data from Table II.5 for 2 MeV
39 mua_rho_air2 = 0.0238; //
    The ratio of total attenuation coefficient to
    density of air in cm2/g
40 // Calculation
41 X_dot2 = 0.0659*E2*mua_rho_air2*phi_b2;
42 printf("\n Exposure rate due to group 2 = %.1 f mR/
    hour \n",X_dot2);
43
44 // By assuming that the gamma rays are of equal
    energy of 4 MeV (Group 3) and looking into Table
    10.5
45 E3 = 4; //
    Energy of gamma rays in MeV (Assumed)
46 chi_pn3 = 0.22; //
    Number of prompt gamma rays emitted per fission
    with energy 4 MeV
47 S3 = chi_pn3*fission_rate; //
    Source strength in gamma rays/sec
48 // Using the data from Table II.4 for E = 4 MeV for
    water
49 mu3 = 0.0339; //

```

```

    Mass attenuation coefficient at 4 MeV in  $\text{cm}^{-1}$ 
50 printf(" \n Buildup factor is due to water measured
    at %.1 f", mu3*R);
51 // Using the data from Table 10.2 at 4 MeV
52 B_p3 = 5.27;
53 phi_b3 = (S3/(4*pi*R^2))*B_p3*exp(-mu3*R); //
    Buildup flux
54 // Using the data from Table II.5 for 4 MeV
55 mua_rho_air3=0.0194; //
    The ratio of total attenuation coefficient to
    density of air in  $\text{cm}^2/\text{g}$ 
56 // Calculation
57 X_dot3 = 0.0659*E3*mua_rho_air3*phi_b3;
58 printf("\n Exposure rate due to group 3 = %.1 f mR/
    hour \n", X_dot3);
59
60 // By assuming that the gamma rays are of equal
    energy of 6 MeV (Group 4) and looking into Table
    10.5
61 E4 = 6; //
    Energy of gamma rays in MeV (Assumed)
62 chi_pn4 = 0.025; //
    Number of prompt gamma rays emitted per fission
    with energy 4 MeV
63 S4 = chi_pn4*fission_rate; //
    Source strength in gamma rays/sec
64 // Using the data from Table II.4 for E = 6 MeV for
    water
65 mu4 = 0.0275; //
    Mass attenuation coefficient at 6 MeV in  $\text{cm}^{-1}$ 
66 printf(" \n Buildup factor is due to water measured
    at %.2 f", mu4*R);
67 // Using the data from Table 10.2 at 6 MeV
68 B_p4 = 3.53;
69 phi_b4 = (S4/(4*pi*R^2))*B_p4*exp(-mu4*R); //
    Buildup flux
70 // Using the data from Table II.5 for 4 MeV
71 mua_rho_air4=0.0172; //

```

```

    The ratio of total attenuation coefficient to
    density of air in cm2/g
72 // Calculation
73 X_dot4 = 0.0659*E4*mua_rho_air4*phi_b4;
74 printf("\n Exposure rate due to group 3 = %.2f mR/
    hour \n",X_dot4);
75
76 //Calculation
77 X_dot = X_dot1+X_dot2+X_dot3+X_dot4;
78 // Result
79 printf("\n The total exposure rate due to all groups
    = %.2f mR/hour \n",X_dot);

```

Scilab code Exa 10.9 Coolant activity

```

1 // Example 10.9
2 clear all;
3 clc;
4
5 // Given data
6 // Assuming average energy produced per fission
    reaction is 200 MeV
7 P = 55; //
    Power density of reactor in watts/cm3
8 rho_eff_U = 0.33; //
    Effective density of uranium in g/cm3
9 rho_eff_W = 1-rho_eff_U; //
    Effective density of water in g/cm3
10 t_i = 3; //
    Average time spent by water in the reactor in
    seconds
11 t_0 = 2; //
    Average time spent by water in the external
    coolant circuit in seconds
12 // 1 eV = 1.6*10(-19) J

```

```

13 P_fission = 200*10^6*1.6*10^(-19);           //
    Energy produced in a fission reaction in terms of
    joule
14 fission_density = P/P_fission;             //
    Number of fission reactions
15 v = 2.42;                                   //
    Number of fission neutrons emitted per fission
16 S = v*fission_density;                     //
    Strength of neutron source in terms of neutrons/
    cm^2-sec
17 // Atom density of oxygen at normal density of 1 g/
    cm^3 is
18 rho = 1;                                    //
    Density of water in g/cm^3
19 N_A = 6.02*10^(23);                         //
    Avogadro's constant
20 M = 18;                                     //
    Molecular weight of water
21 atom_density = (rho*N_A)/M;
22
23 // Using the data from Table 10.7
24 sigma_r = 1.9*10^(-5)*10^(-24);           //
    Reaction cross section in cm^2
25 T_12 = 7.1;                                 //
    Half life of the given reaction in seconds
26 lambda = 0.693/T_12;                       //
    Decay constant of the given reaction in seconds
    ^(-1)
27 sigma_act = rho_eff_W*atom_density*sigma_r; //
    Average macroscopic activation cross section
28 // Using the data from Table 10.4
29 sigma_RW = 0.103;                           //
    Removal cross section of water in cm^-1
30 sigma_RU = 0.174;                           //
    Removal cross section of Uranium in cm^-1
31 sigma_R = (sigma_RU*rho_eff_U)+(sigma_RW*rho_eff_W);
    // Removal cross section of mixture
32 // Let activation rate given by (sigma_act*phi_av)

```



```
    be denoted as AR
33 AR = (sigma_act*S)/sigma_R;
34 // Calculation
35 alpha = AR*(1-exp(-t_i*lambda))/(1-exp(-(t_i+t_0)*
    lambda));
36 // 1 curie = 3.7*10^(10) disintegrations/sec
37 // Result
38 printf("\n Equilibrium activity of water due to
    neutron capture of oxygen = %.2E disintegrations/
    cm^3-sec or %d uCi/cm^3 \n",alpha,ceil(alpha
    *10^6/(3.7*10^10)));
```

Chapter 11

Reactor Licensing Safety and the Environment

Scilab code Exa 11.1 Diffusion of radioactive effluents

```
1 // Example 11.1
2 clear all;
3 clc;
4
5 // Given data
6 h = 30; //
   Height at which the effluent is released
7 // Calculation of maxima location
8 sigma_z = h/sqrt(2); //
   Vertical dispersion coefficient
9 // Using the plot given in Figure 11.12 for Type F
   condition
10 // The corresponding value with calculated maximum
   location is noted.
11 h_max = 1900;
12 // Result
13 printf("\n The point of maximum concentration of
   non-radioactive effluent = %d m \n",h_max);
```

Scilab code Exa 11.2 External dose from gamma rays

```
1 // Example 11.2
2 clear all;
3 clc;
4
5 // Given data
6 A = 2*10^3; //
   Amount of radioactivity release due to Xenon-133
   in curie
7 t = 365*24*3600; // Time
   in seconds
8 Q_dash = A/t; //
   Average emission rate of Xenon-133
9 h = 100; //
   Location of radioactivity release through vent
10 v_bar = 1; // Wind
   speed in m/sec
11 // Using the plot given in Figure 11.11 and 11.12
   for Type F condition at 100 m
12 sigma_y = 275; //
   Horizontal dispersion coefficient
13 sigma_z = 46; //
   Vertical dispersion coefficient
14 chi = (Q_dash*exp(-h^2/(2*sigma_z^2)))/(%pi*v_bar*
   sigma_y*sigma_z); // Radionuclide
   concentration in terms of Ci/cm^3
15 // Using data from Table 11.3
16 Eg_bar = 0.03; //
   Average gamma decay energy per disintegration in
   MeV
17 // Calculation
18 H_dot = 0.262*chi*Eg_bar*t*10^3; // The
   units are in mrem/year
```

```

19 // Expressing the dose rate in SI units
20 H_dot_SI = 2.62*chi*Eg_bar*t*10^3;
21 // Result
22 printf(" \n The external gamma dose rate due to
        xenon release under type F condition = %.4f mrem/
        year or %.3f mSv/year \n",H_dot,H_dot_SI);

```

Scilab code Exa 11.3 External dose from gamma rays

```

1 // Example 11.3
2 clear all;
3 clc;
4
5 // Using data from Table 11.3
6 Eg_bar = 0.00211; //
        Average gamma decay energy per disintegration in
        MeV
7 // Calculation
8 C_y = 0.262*Eg_bar;
9 // Result
10 printf(" \n The dose rate factor due to krypton
        exposure = %.2E rem*m^3/sec-Ci \n",C_y);

```

Scilab code Exa 11.4 External dose from gamma rays

```

1 // Example 11.4
2 clear all;
3 clc;
4
5 // The results given in Example 11.2 are to be used
        in this problem
6 chi = 1.5*10^(-10); //
        Radionuclide concentration in terms of Ci/cm^3

```

```

7 t = 365*24*3600; //
  Time in seconds
8 // Using data from Table 11.3
9 Ebeta_bar = 0.146; //
  Average gamma decay energy per disintegration in
  MeV
10 f = 1; //
  Experimentally detemined factor
11 // Calculation
12 H_dot = 0.229*Ebeta_bar*chi*f*t;
13 // Expressing the result in milli-rem
14 printf("\n The external beta dose rate due to xenon
  exposure for a year = %.3f mrem/year \n",H_dot
  *10^3);

```

Scilab code Exa 11.5 Dose rate to thyroid

```

1 // Example 11.5
2 clear all;
3 clc;
4
5 // Given data
6 A = 1.23; //
  Amount of radioactivity release due to I-131 in
  curie in one year
7 h = 30; //
  Location of radioactivity release through vent in
  meter
8 v_bar = 1.2; // Wind
  speed in m/sec
9 T_12 = 8.04; // Half
  life of Iodine 131 in days
10 T_12b = 138; //
  Biological half life of Iodine 131 in days
11 zeta = 0.23; //

```

```

    Fraction of core inventory in MeV
12 q = 0.23; //
    Fraction of Iodine-131 in thyroid
13 M = 20; // Mass
    of an adult thyroid in grams
14
15 // 1.
16 t = 365*24*3600; // Time
    in seconds
17 Q_dash = A/t; //
    Average emission rate of Iodine-131
18 // Converting days into seconds by using 1 day =
    86400 seconds
19 lambda = 0.693/(T_12*86400); // Decay
    constant of Iodine-131
20 lambda_b = 0.693/(T_12b*86400); //
    Biological decay constant of Iodine-131
21 // Let the quantity  $\chi \cdot \bar{v} / \bar{Q} = x$ 
22 // Using the plot given in Figure 11.13 for Type E
    condition at 2000 m
23 x = 6*10(-5);
24 // Solving for  $\chi$ 
25  $\chi = (x \cdot Q\_dash) / \bar{v}$ ;
26 // As per the standards of International Commission
    on Radiological Protection (ICRP)
27 B = 2.32*10(-4); //
    Normal breathing rate in m3/sec
28 // Calculation
29  $H\_dot = (592 \cdot B \cdot \zeta \cdot q \cdot \chi) / (M \cdot (\lambda + \lambda\_b))$ ;
30 // Result
31 printf("\n The equilibrium dose rate to an adult
    thyroid = %.2E rem/sec \n", H_dot);
32
33 // 2.
34 // Calculation
35 H = H_dot*t;
36 // Expressing the result in milli-rem
37 // Result

```

```

38 printf("\n The annual dose to an adult thyroid = %
    .2f mrem \n",H*10^3);

```

Scilab code Exa 11.6 Ground exposure from gamma rays

```

1 // Example 11.6
2 clear all;
3 clc;
4
5 // Given data
6 E = 0.66; //
    Energy of gamma ray emitted by caesium in MeV //
7 x = 100; //
    Height of exposure in cm //
8 // Using the data from Table II.5 for air at E =
    0.66 MeV
9 mua_rho = 0.0294; //
    The ratio of absorption coefficient to density of
    air in cm^2/g
10 // Using the data from Table II.4 for air at E =
    0.66 MeV
11 mu_rho = 0.0775; //
    The ratio of attenuation coefficient to density
    of air in cm^2/g
12 // Using standard value for density of air
13 rho = 1.293*10^(-3);
14 mu = mu_rho*rho;
15 mux = mu*x;
16 gamma = 0.57722; //
    Euler's constant
17 E1 = -gamma+log(1/mux)+mux; //
    Conversion factor
18 // Using parameter data from Table 11.16
19 C = 1.41; // A
    constant

```

```

20 beta = 0.0857;                                // A
    constant
21 // Calculation
22 H_dot_S = 3.39*10^(-2)*E*mua_rho*(E1+(C*exp(-(1-beta
    )*mux)/(1-beta)));
23 // Converting time in hours by 1 hour = 3600 seconds
24 // Result
25 printf(" \n The gamma ray dose rate conversion
    factor due to caesium-137 = %.2E rem*m^2/sec-Ci
    or %.2 f rem*m^2/hour-Ci\n", H_dot_S, H_dot_S*3600);

```

Scilab code Exa 11.7 External dose from gamma rays

```

1 // Example 11.7
2 clear all;
3 clc;
4
5 // Given data
6 C0 = 6.25*10^6;                                //
    Amount of initial radioactivity release due to I
    -131 in curie
7 p = 0.1;                                       //
    Leakage rate in percent
8 t0 = 2*3600;                                   //
    Analysis time in seconds
9 v_bar = 1;                                     // Wind
    speed in m/sec
10
11 // 1.
12 lambda1 = 0.01*p/86400;                       // Decay
    constant corresponding to leakage rate in
    seconds (1 day = 86400 seconds)
13 // Using the data from Example 11.5 for the half
    life of Iodine-131
14 T_12 = 8.04;                                  // Half

```



```

    life of Iodine 131 in days
15  lambdac = 0.693/(T_12*86400);           // Decay
    constant of Iodine-131 (I-131) in seconds
16  // Using data from Table 11.3
17  Eg_bar = 0.371;                         //
    Average gamma decay energy per disintegration of
    I-131 in MeV
18  // Using the plot given in Figure 11.11 and 11.12
    for Type F condition at 100 m
19  sigma_y = 21;                           //
    Horizontal dispersion coefficient
20  sigma_z = 70;                           //
    Vertical dispersion coefficient
21  // As lambdac*t << 1,
22  // Calculation
23  H = (0.262*Eg_bar*lambdal*C0*t0)/(%pi*v_bar*sigma_y*
    sigma_z);
24  // Result
25  printf(" \n The total external dose due to gamma ray
    exposure = %.2E rem\n",H)
26
27  // 2.
28  // Using the data given in Example 11.5
29  B = 2.32*10^(-4);                       //
    Normal breathing rate in m^3/sec
30  zeta = 0.23;                            //
    Fraction of core inventory in MeV
31  q = 0.23;                               //
    Fraction of Iodine-131 in thyroid
32  M = 20;                                  // Mass
    of an adult thyroid in grams
33  // Calculation
34  H_dot = (592*B*zeta*q*lambdal*C0*t0)/(%pi*v_bar*
    sigma_y*sigma_z*M);
35  // Converting the units from rem/sec to milli-rem/
    hour by multiplying by (1000*3600)
36  // Result
37  printf(" \n The dose rate to an adult thyroid after

```

```

    2 hours = %.2E rem/sec or %d mrem/hour\n",H_dot ,
    ceil(H_dot*(1000*3600)));
38
39 // 3.
40 // Using the data given in Example 11.5
41 T_12 = 8.04; // Half
    life of Iodine 131 in days
42 T_12b = 138; //
    Biological half life of Iodine 131 in days
43 lambda = 0.693/(T_12*86400); // Decay
    constant of Iodine-131 in sec(-1)
44 lambda_b = 0.693/(T_12b*86400); //
    Biological decay constant of Iodine-131 in sec
    (-1)
45 // Calculation
46 H = H_dot/(lambda+lambda_b);
47 // Result
48 printf("\n The dose commitment to thyroid by Iodine
    -131 exposure after 2 hours = %.2f rem \n",H);

```

Scilab code Exa 11.8 Direct dose from gamma rays

```

1 // Example 11.8
2 clear all;
3 clc;
4
5 // Given
6 E = 2.4; //
    Energy of gamma rays in MeV
7 r = 1000*100; //
    Distance from the building where radiation is
    exposed in cm
8 t0 = 2*3600; // Time
    of exposure in seconds
9 A = 3*107; //

```

```

    Amount of initial radioactivity release due to Kr
    -88 in curie
10 f = 0.4; //
    Fraction of disintegrations which release 2.4 MeV
    gamma rays
11 C0 = A*f; //
    Effective number of curies
12 T_12 = 2.79; // Half
    life of Iodine 131 in hours
13
14 lambda = 0.693/(T_12*3600); // Decay
    constant of Iodine-131 in sec(-1)
15 // Using the result given in Example 11.7
16 lambdal = 1.16*10(-8); // Decay
    constant corresponding to fission product
    release from building
17 lambdac = lambda+lambdal; // Total
    decay constant in sec(-1)
18 // Using the data from Table II.4 for air at E = 2.4
    MeV
19 mu_rho = 0.041; // The
    attenuation coefficient in cm2/g
20 // Using standard value for density of air in g/cm3
21 rho = 1.293*10(-3);
22 mu = mu_rho*rho;
23 // Using the data from Table II.5 for air at E = 2.4
    MeV
24 mua_rho = 0.0227; // The
    ratio of absorption coefficient to density of air
    in cm2/g
25 printf("\n Buildup factor is measured at %.2f",mu*r
);
26 // Using Berger's form in Problem 11.9
27 B_p = 4.7; //
    Buildup factor due to a point source
28 // Calculation
29 H = (54*C0*(1-exp(-lambdac*t0))/lambdac)*(E*mua_rho*
    B_p*exp(-mu*r)/r2);

```

```

30 // Result
31 printf("\n The direct dose due to gamma ray
    exposure = %.4f rem \n",H)
32 // There is a slight deviation in the answer due to
    approximation of value in the textbook.

```

Scilab code Exa 11.9 Activity of radionuclide release into atmosphere

```

1 // Example 11.9
2 clear all;
3 clc;
4
5 // Given data
6 gammai = 0.0277; //
    Fission yield of Iodine-131 in fraction
7 P = 3200; //
    Thermal power of the plant in MW
8 // Calculation
9 alphi = 8.46*10^5*P*gammai;
10 // Result
11 printf("\n The saturation activity of Iodine-131
    during reactor operation = %.2E curie \n",alpha)
12
13 // Using assumption 2 of Nuclear Regulatory
    Commission (NRC) in calculation of radii of
    exclusion zone and Low Population Zone (LPZ)
14 // Due to core meltdown, 25 percent of iodine
    inventory is released and out of which 91 percent
    is in elemental form.
15 Fp = 0.25*0.91; //
    Fraction of Iodine-131 released from the fuel
    into the reactor containment
16 // As entire iodine escapes through air
17 Fb = 1; //
    Fraction of 'Fp' which remains airborne and is

```

```

    capable of escaping from the building
18 // Calculation
19 C0 = alphai*Fp*Fb;
20 // Result
21 printf("\n The activity of Iodine-131 in elemental
    form due to core meltdown = %.2E curie \n",C0);

```

Scilab code Exa 11.10 Concentration of radionuclide

```

1 // Example 11.10
2 clear all;
3 clc;
4
5 // Given data
6 P = 1000; //
    Electrical power of the plant in Mwe
7 eta = 0.38; //
    Plant efficiency
8 P_th = P/eta; //
    Thermal power of the plant in MW
9 h = 100; //
    Height of stack in metre
10 t = 24*365; //
    The number of hours in a year
11 m0 = 13000; //
    Amount of coal in the plant in Btu/lb
12 m0_ash = 0.09; //
    Fraction of ash in the coal
13
14 // 1.
15 E = P_th*t; //
    Energy released in a year in MW-hour
16 // Converting the units in Btu by using 1 MW-hour =
    3.412*10^6 Btu
17 m = (E*3.412*10^6)/m0;

```

```

18 // Converting the units in g/year by using 1 lb/year
    = 453.59237 g/year
19 m = m*453.59237;
20 // Assuming the fly ash equipment has an efficiency
    of 97.5% of fly ash removal
21 eta_flyash = 0.025; //
    Only (1-efficiency) is exhausted
22 m_ash = eta_flyash*m0_ash*m;
23 // A typical power plant contains 3pCi/g of each
    nuclide (Radium-226) in one year
24 A = 3*10^(-12);
25 // Calculation
26 A_total = A*m_ash;
27 // Result
28 printf(" \n Total activity of Radium-226 emitted = %
    .4f curie \n",A_total)
29
30 // 2.
31 v_bar = 1; //
    Wind speed in m/sec
32 t = 24*365*3600; //
    Analysis time of one year equivalently in seconds
33 MPC = 3*10^(-12); //
    Maximum Permissible Concentration in micro-curie/
    cm^3
34 Q_bar = A_total/t; //
    Emission rate for one year in curie/year
35 // Let the quantity chi*v_bar/Q_bar = x
36 // Using the plot for Pasquill F, given in Fig. A.7,
    Pg no 413 from Slade, D. H., Editor, Meteorology
    and Atomic Energy-1968. U. S. Atomic Energy
    Commission Report TID-24190, 1968.
37 x = 2.5*10^(-6); //
    Maximum value of x at 10^4 m
38 // Solving for chi
39 chi = (x*Q_bar)/v_bar;
40 // Converting the units from Ci/m^3 to micro-curie/
    cm^3 by multiplying by 10^6/10^6 = 1

```

```

41 printf(" \n Concentration of Radium-226 present = %
    .2E micro-curie/cm^3 \n",chi)
42 // Let c be the comparison factor
43 // Calculation
44 c = MPC/chi;
45 // Result
46 printf(" \n On comparison, the total concentration
    of Radium-226 is %d times smaller than Maximum
    Permissible Concentration (MPC) \n",c)
47 // The comparison factor is slightly different from
    the value in the textbook. This is because of
    approximation used in the textbook.

```

Scilab code Exa 11.11 Activity of radioactive gas effluent

```

1 // Example 11.11
2 clear all;
3 clc;
4
5 // Given data
6 Qy_bar = 1.04*10^(-2); //
    Emission rate for one year in curie/year
7 // Let (chi/Q_bar) = d which is called 'Dilution
    factor '
8 d = 4*10^(-8); //
    Dilution factor in year/cm^3
9 vd = 0.01; //
    Experimentally determined constant
10
11 // 1.
12 T_12 = 8.04; // Half
    life of Iodine 131 in days
13 T_12f = 14; // First
    order half life of Iodine 131 in days
14 // Converting days into years by using 1 year = 365

```

```

    days
15 lambda = 0.693/(T_12/365);           // Decay
    constant of Iodine-131
16 lambdaf = 0.693/(T_12f/365);       // First
    order decay constant of Iodine-131
17 // Calculation
18 Cf = (Qy_bar*d*vd)/(lambda+lambdaf);
19 // Expressing the result in micro-curie
20 Cf = Cf*10^6;
21 // Result
22 printf(" \n The activity of I-131 on the vegetation
    = %.2E micro-curie/m^2 \n", Cf);
23
24 // 2.
25 // The proportionality factor has a value  $9 \times 10^{-5}$ 
    Ci/cm^3 of milk per Ci/m^2 of grass
26 // Calculation
27 Cm = 9*10^(-5)*Cf;
28 // Result
29 printf(" \n The concentration of I-131 in the milk =
    %.2E micro-curie/m^2 \n", Cm);
30
31 // 3.
32 MPC = 3*10^(-7);                     //
    Maximum Permissible Concentration in micro-curie/
    cm^3
33 // Calculation
34 H_dot = (2270*Cm)/MPC;
35 // Result
36 printf(" \n The annual dose rate to an infant
    thyroid by consuming radiated milk = %.2E mrem/
    year \n", H_dot);

```

Scilab code Exa 11.12 Activity of radioactive liquid effluent


```

1 // Example 11.12
2 clear all;
3 clc;
4
5 // Given data
6 Qy_bar = 0.197; //
   Emission rate for one year in micro-curie/year
7 // Let (chi/Q_bar) = d which is called 'Dilution
   factor'
8 d = 6.29*10^(-16); //
   Dilution factor in year/cm^3
9 MPC_w = 6*10^(-5); //
   Maximum Permissible Concentration (MPC) of iron
   in micro-curie/cm^3
10
11 Cw = Qy_bar*d; // The
   concentration of Fe-59
12 // For fish
13 Rs_fish = 110; //
   Consumption rate in g/day
14 // Using the data from Table 11.15 for saltwater
   concentration of fish for iron
15 CF_fish = 1800; //
   Concentration Factor of fish
16 Cs_fish = CF_fish*Cw; //
   Activity of fish
17 H_dot_fish = (Cs_fish*Rs_fish*500)/(MPC_w*2200);
   // Dose rate for fish
18
19 // For mollusks
20 Rs_mollusk = 10; //
   Consumption rate in g/day
21 // Using the data from Table 11.15 for saltwater
   concentration of mollusk for iron
22 CF_mollusk = 7600; //
   Concentration Factor of mollusk
23 Cs_mollusk = CF_mollusk*Cw; //
   Activity of mollusk

```

```

24 H_dot_mollusk = (Cs_mollusk*Rs_mollusk*500)/(MPC_w
    *2200); // Dose rate for mollusk
25
26 // For crustaceans
27 Rs_crustacean = 10; //
    Consumption rate in g/day
28 // Using the data from Table 11.15 for saltwater
    concentration of crustacean for iron
29 CF_crustacean = 2000; //
    Concentration Factor of crustacean
30 Cs_crustacean = CF_crustacean*Cw; //
    Activity of crustacean
31 H_dot_crustacean = (Cs_crustacean*Rs_crustacean*500)
    /(MPC_w*2200); // Dose rate for crustacean
32
33 // Calculation
34 H_dot = H_dot_fish+H_dot_mollusk+H_dot_crustacean;
35 // Result
36 printf(" \n The annual dose rate to GI tract by
    consuming fish = %.2E mrem/year",H_dot_fish);
37 printf(" \n The annual dose rate to GI tract by
    consuming mollusk = %.2E mrem/year",H_dot_mollusk
    );
38 printf(" \n The annual dose rate to GI tract by
    consuming crustaceans = %.2E mrem/year",
    H_dot_crustacean);
39 printf(" \n The annual dose rate to GI tract by
    consuming seafood = %.2E mrem/year \n",H_dot);
40 // The answer for annual dose rate to GI tract by
    consuming fish is wrong in the textbook. This is
    because the value of fish consumption rate is
    wrongly considered.

```
