

```

#include <iostream>
#include <discpp.h>
#include <FileHandle.hpp>

#include <vector>
#include <string>
#include <sstream>
#include <limits>
#include <algorithm>
#include <tuple>

using namespace std;

static Dislin dlin;

#define k_B 1.3806488E-23
#define sigma_B 5.670373E-8
#define m_e 9.10938291E-31 //kg
#define m_p 1.67262178E-27 //kg
#define eV 1.60217657E-19 //joules
#define h 6.62606957E-34
#define hbar 1.054571596E-34
#define pi 3.141592654
#define c_0 299792458.0
#define q_e 1.602176462E-19
#define H_0 70.0*1000.0/3.08567758E22
#define GYR 365.25*24.0*60.0*60.0*1E9
#define AU 149597870700.0 //meters
#define pc 206264.81 * AU
#define R_Sun 695800000.0

const double GYRH_0 = GYR*H_0;
char legendText[255] = { '/0' };
int legendIndex = 1;

void dislinInit(char* xName, char* yName, char* title){
    dlin.disini();
    //dlin.nochek();
    dlin.pagera();
    dlin.name(xName, "X");
    dlin.name(yName, "Y");
    dlin.titlin(title, 1);

    legendIndex = 1;
    dlin.complx();
    dlin.texmod("ON");
};

void q2(){
    //==== PART A =====
    vector<double> B_lambda, lambda;
    tuple<double, double> lambda_max = make_tuple(-INFINITY, 0);

    dislinInit("\\lambda (nm)", "$B_{\\lambda}$ $(1\\times 10^{13})$", "Question 2a");

    auto planck_lambda = [](double lambda, double T){ return 2.0*h*c_0*c_0 / pow(lambda, 5) /
(exp(h*c_0 / lambda / k_B / T) - 1.0); };

    for (double temp_lam = 100; temp_lam <= 2000; temp_lam += 0.01){
        lambda.push_back(temp_lam);
        double val = planck_lambda(temp_lam * 1.0E-9, 5500.0) / 1.0E13;
        B_lambda.push_back(val);

        if (val > get<1>(lambda_max)){
            lambda_max = make_tuple(temp_lam, val);
        }
    }
}

```

```

dlin.labdig(-1, "X");
dlin.labdig(1, "Y");

dlin.graf(0, 2000, 0, 500, 0, 2.5, 0, 0.5);

dlin.title();

dlin.color("Red");
dlin.curve(lambda.data(), B_lambda.data(), lambda.size());

dlin.color("FORE");
dlin.rlsymb(SYMBOL_CIRCLE, get<0>(lambda_max), get<1>(lambda_max));
{
    stringstream ss;
    ss << "(" << get<0>(lambda_max) << " , " << get<1>(lambda_max) << ")";
    dlin.messag(ss.str().c_str(), 900, 580);
}

dlin.disfin();

//===== PART B =====

dislinInit("\nu $(1\times 10^{14}Hz)$", "$B_{\nu}$ $(1\times 10^{-8})$", "Question 2b");

vector<double> B_nu, nu;
tuple<double, double> nu_max = make_tuple(-INFINITY, 0);

auto planck_nu = [(double nu, double T){ return nu == 0 ? 0 : (2.0*h / c_0 / c_0 * pow(nu, 3) /
(exp(h * nu / k_B / T) - 1.0)); }];

for (double temp_nu = 0.0; temp_nu <= 30.0; temp_nu += 0.001){
    nu.push_back(temp_nu);
    double val = planck_nu(temp_nu * 1.0E14, 5500.0) / 1E-8;
    B_nu.push_back(val);

    if (val > get<1>(nu_max)){
        nu_max = make_tuple(temp_nu, val);
    }
}

dlin.graf(0, 30, 0, 5, 0, 3.5, 0, 0.5);

dlin.title();

dlin.color("Red");
dlin.curve(nu.data(), B_nu.data(), nu.size());

dlin.color("FORE");
dlin.rlsymb(SYMBOL_CIRCLE, get<0>(nu_max), get<1>(nu_max));
{
    stringstream ss;
    ss << "(" << get<0>(nu_max) << " , " << get<1>(nu_max) << ")";
    dlin.messag(ss.str().c_str(), 600, 450);
}

dlin.disfin();

//===== PART E =====
auto part_e_helper = [&](double T){
    vector<double> tx, ty;
    double x, y;
    for (double temp_x = 1; temp_x <= 1E6; temp_x += 2){
        //tx.push_back(log10(temp_lam));
        x = temp_x;
        y = temp_x*1E-9*planck_lambda(temp_x*1E-9, T);

        if (x <= 0 || y <= 0)
            continue;
    }
}

```

```

        tx.push_back(x);
        ty.push_back(y);
    }

    dlin.curve(tx.data(), ty.data(), tx.size());
};

dislinInit("$\\log{\\lambda} (nm)$", "$\\log{\\lambda B_{\\lambda}} = \\log{\\nu B_{\\nu}}$", "Question 2e - Full");

dlin.nochek();

dlin.axsscl("LOG", "XY");
dlin.labels("ELOG", "XY");

dlin.graf(1, 6, 1, 1, 0, 11, 0, 1);

dlin.legini(legendText, 3, 10);
dlin.title();
dlin.color("Blue");
dlin.lintyp(LINE_SOLID);
dlin.leglin(legendText, "3000K", legendIndex++);
part_e_helper(3000);

dlin.lintyp(LINE_CHNDSH);
dlin.color("Orange");
dlin.leglin(legendText, "5500K", legendIndex++);
part_e_helper(5500);

dlin.lintyp(LINE_DOT);
dlin.color("Red");
dlin.leglin(legendText, "30000K", legendIndex++);
part_e_helper(30000);

dlin.color("FORE");
dlin.legtit("Legend");
dlin.legpos(2020, 420);
dlin.legend(legendText, legendIndex - 1);

dlin.disfin();

dislinInit("$\\lambda (nm)$", "$\\log{\\lambda B_{\\lambda}} = \\log{\\nu B_{\\nu}}$", "Question 2e - Visible");

dlin.nochek();

dlin.axsscl("LOG", "Y");
dlin.labels("ELOG", "Y");

dlin.graf(400, 800, 400, 100, 4, 10, 4, 1);

dlin.legini(legendText, 3, 10);
dlin.title();
dlin.color("Blue");
dlin.lintyp(LINE_SOLID);
dlin.leglin(legendText, "3000K", legendIndex++);
part_e_helper(3000);

dlin.lintyp(LINE_CHNDSH);
dlin.color("Orange");
dlin.leglin(legendText, "5500K", legendIndex++);
part_e_helper(5500);

dlin.lintyp(LINE_DOT);
dlin.color("Red");
dlin.leglin(legendText, "30000K", legendIndex++);
part_e_helper(30000);

```

```

dlin.color("FORE");
dlin.legtit("Legend");
dlin.legpos(2020, 420);
dlin.legend(legendText, legendIndex - 1);

dlin.color("FORE");
dlin.legtit("Legend");
dlin.legpos(2020, 420);
dlin.legend(legendText, legendIndex - 1);

dlin.disfin();
}

void q4(){
    stringstream ss;
    //===== PART A =====
    dislinInit("Depth ($km$)", "Temperature (K)", "Question 4a");
    dlin.labdig(-1, "XY");
    dlin.graf(0, 1000, 0, 100, 8408.964, 13000, 9000, 500);
    dlin.addlab("8408.96", 8408.96415, 1, "Y");
    dlin.title();

    vector<double> vS, vT;

    tuple<double, double, double> tau_2_3 = make_tuple(0, 0, -INFINITY);
    tuple<double, double> T_T_e = make_tuple(0, -INFINITY);

    for (double s = 0; s < 1000; s += 0.001){
        vS.push_back(s);

        double tau = 1E-6*3.0*s * 1000.0;

        double T = 10000.0 * pow(3.0 / 4.0*(tau + 2.0 / 3.0), 1.0 / 4.0);

        if (tau <= 2.0 / 3.0 && tau > get<2>(tau_2_3)){
            tau_2_3 = make_tuple(s, T, tau);
        }

        if (T <= 10000.0 && T > get<1>(T_T_e)){
            T_T_e = make_tuple(s, T);
        }

        vT.push_back(T);
    }

    dlin.color("Red");
    dlin.curve(vS.data(), vT.data(), vS.size());
    dlin.color("Blue");
    dlin.rlsymb(SYMBOL_CIRCLE, get<0>(T_T_e), get<1>(T_T_e));
    dlin.rlsymb(SYMBOL_PLUS, get<0>(tau_2_3), get<1>(tau_2_3));

    dlin.color("FORE");
    ss << "(" << get<0>(T_T_e) << "$km$ = " << get<0>(T_T_e) / R_Sun*1000.0/1E-4 << "$\\times 10^{-4}R_\\odot" << " , " << get<1>(T_T_e) << ")";
    dlin.messag(ss.str().c_str(), 970, 1300);

    dlin.disfin();

    //===== PART B =====

    auto getTemp = [](double s){return 10000.0 * pow(3.0 / 4.0*(1E-6*3.0*s * 1000.0 + 2.0 / 3.0), 1.0 / 4.0);};
    tuple<double, double> f2_max = make_tuple(0, -INFINITY);

    dislinInit("Depth (km)", "$f_2(s)$ $(\\times 10^{-5})$", "Question 4b");
    dlin.labdig(-1, "X");

```

```

dlin.graf(0, 1000, 0, 100, 0, 2, 0, 0.5);
dlin.title();
//dlin.nochek();

vector<double> vf2;

for each (double s in vS){
double T = getTemp(s);
double b = (m_p / 1E-6*pow(m_e*k_B*T / 2.0 / pi / hbar / hbar, 3.0 / 2.0)*exp(-13.6*eV /
k_B / T));
double NI_Nt = 1.0 - b*0.5*(-1.0 + sqrt(1.0 + 4.0 / b));

double N6I_N1I = 36.0*exp(13.6*eV / k_B / T*(1.0 / 36.0 - 1.0));
double N5I_N1I = 25.0*exp(13.6*eV / k_B / T*(1.0 / 25.0 - 1.0));
double N4I_N1I = 16.0*exp(13.6*eV / k_B / T*(1.0 / 16.0 - 1.0));
double N3I_N1I = 9.0*exp(13.6*eV / k_B / T*(1.0 / 9.0 - 1.0));
double N2I_N1I = 4.0*exp(13.6*eV / k_B / T*(1.0 / 4.0 - 1.0));
double N2I_NI = N2I_N1I / (1 + N2I_N1I + N3I_N1I + N4I_N1I + N5I_N1I + N6I_N1I);

double f2 = N2I_NI * NI_Nt;

if (f2 > get<1>(f2_max)){
f2_max = make_tuple(s, f2);
}

vf2.push_back(f2 / 1E-5);
}

dlin.color("Red");
dlin.curve(vS.data(), vf2.data(), vS.size());

dlin.color("FORE");
dlin.rlsymb(SYMBOL_CIRCLE, get<0>(f2_max), get<1>(f2_max) / 1E-5);
ss.str("");
ss << "(" << get<0>(f2_max) << ", " << get<1>(f2_max) / 1E-5 << ")";
dlin.messag(ss.str().c_str(), 1440, 420);

dlin.disfin();

//===== PART C =====
vector<double> vTau_normal, vTau_balmer;

tuple<double, double> tau_normal_2_3 = make_tuple(0, -INFINITY), tau_balmer_2_3 = make_tuple(0, -
INFINITY);

double ds = vS[1] - vS[0];

double tau = 0;
double tau_balmer = 0;

for (int i = 0; i < vS.size(); i++){
if (vS[i] > 350)
break;
double tau_normal = 1E-6*3.0*vS[i] * 1000.0;
vTau_normal.push_back(tau_normal);

if (tau_normal <= 2.0 / 3.0 && tau_normal > get<1>(tau_normal_2_3)){
tau_normal_2_3 = make_tuple(vS[i], tau_normal);
}

double T = 10000.0 * pow(3.0 / 4.0*(vTau_normal[i] + 2.0 / 3.0), 1.0 / 4.0);

double b = (m_p / 1E-6*pow(m_e*k_B*T / 2.0 / pi / hbar / hbar, 3.0 / 2.0)*exp(-13.6*eV /
k_B / T));

double NI_Nt = 1 - b*0.5*(-1.0 + sqrt(1.0 + 4.0 / b));

double N6II_N1I = 36.0*exp(13.6*eV / k_B / T*(1.0 / 36.0 - 1.0));
double N5II_N1I = 25.0*exp(13.6*eV / k_B / T*(1.0 / 25.0 - 1.0));

```

```

double N4II_N1I = 16.0*exp(13.6*eV / k_B / T*(1.0 / 16.0 - 1.0));
double N3II_N1I = 9.0*exp(13.6*eV / k_B / T*(1.0 / 9.0 - 1.0));
double N2II_N1I = 4.0*exp(13.6*eV / k_B / T*(1.0 / 4.0 - 1.0));
double N2II_NI = N2II_N1I / (1 + N2II_N1I + N3II_N1I + N4II_N1I + N5II_N1I + N6II_N1I);

double f3 = N2II_NI * NI_Nt;

double dtau_balmer = f3*1E-6*3.5E5*1000.0*ds;

tau_balmer += dtau_balmer;

vTau_balmer.push_back(vTau_normal[i] + tau_balmer);

if ((vTau_normal[i] + tau_balmer) <= 2.0 / 3.0 && (vTau_normal[i] + tau_balmer) >
get<1>(tau_balmer_2_3)){
    tau_balmer_2_3 = make_tuple(vS[i], vTau_normal[i] + tau_balmer);
}
}

dislinInit("Depth (km)", "$\\tau$", "Question 4c");
dlin.labdig(-1, "x");
dlin.graf(0, 350, 0, 50, 0, 1, 0, 0.1);
dlin.title();

dlin.nocek();

dlin.color("Red");
dlin.curve(vS.data(), vTau_balmer.data(), vTau_balmer.size());
dlin.color("FORE");
dlin.curve(vS.data(), vTau_normal.data(), vTau_normal.size());
dlin.lintyp(LINE_DOT);
dlin.rline(0, 2.0 / 3.0, 350, 2.0 / 3.0);

dlin.rlsymb(SYMBOL_CIRCLE, get<0>(tau_normal_2_3), get<1>(tau_normal_2_3));
ss.str("");
ss << "(" << get<0>(tau_normal_2_3) << ", $\\frac{2}{3}$";
dlin.messag(ss.str().c_str(), 1780, 900);

dlin.rlsymb(SYMBOL_CIRCLE, get<0>(tau_balmer_2_3), get<1>(tau_balmer_2_3));
ss.str("");
ss << "(" << get<0>(tau_balmer_2_3) << ", $\\frac{2}{3}$";
dlin.messag(ss.str().c_str(), 920, 790);

dlin.disfin();
}

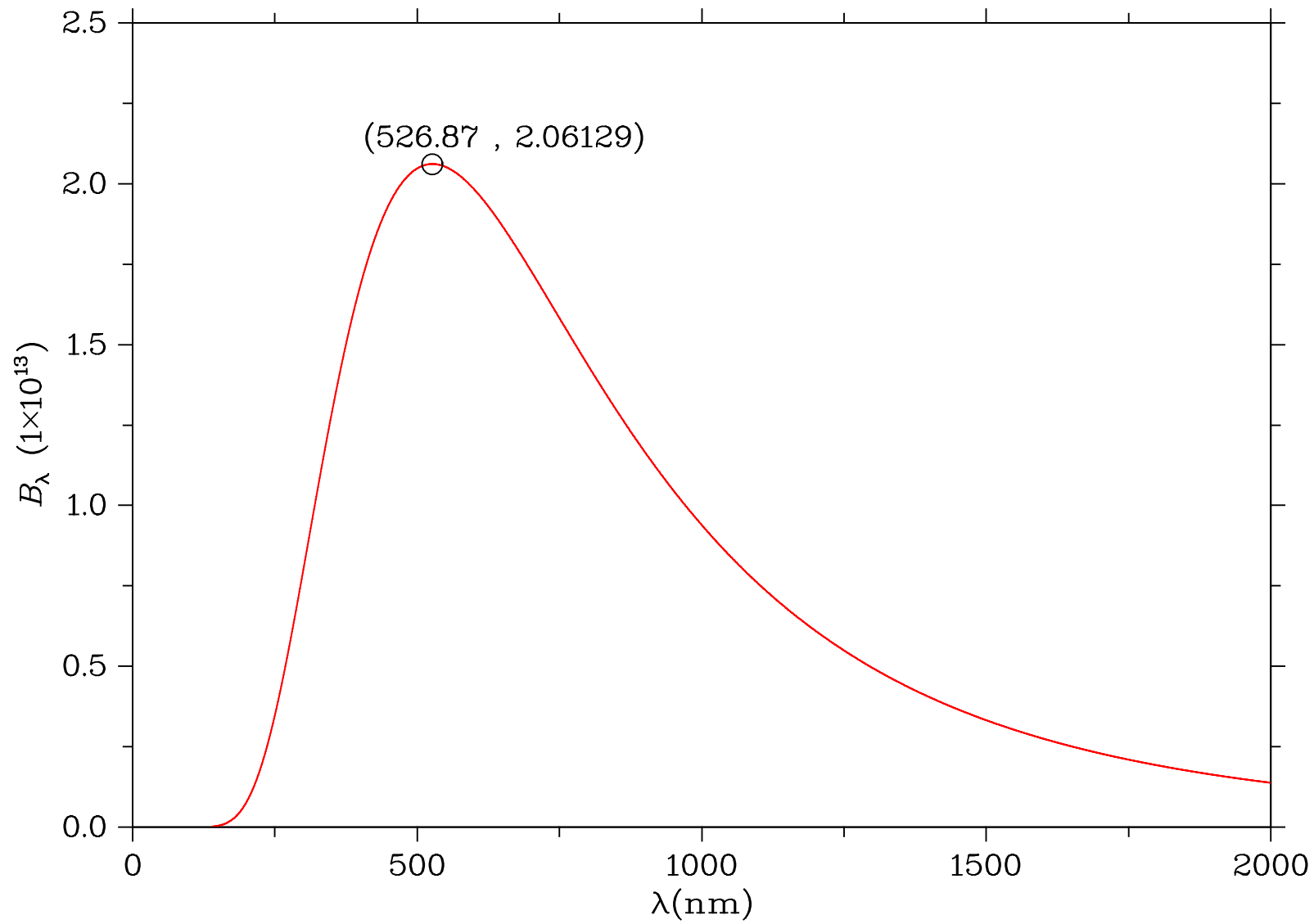
int main() {
    dlin.metafl("PDF");
    dlin.imgfmt("RGB");
    dlin.filmod("COUNT");
    dlin.scrmod("REVERS");
#ifdef _DEBUG
    dlin.metafl("XWIN");
#endif

    q2();
    q4();

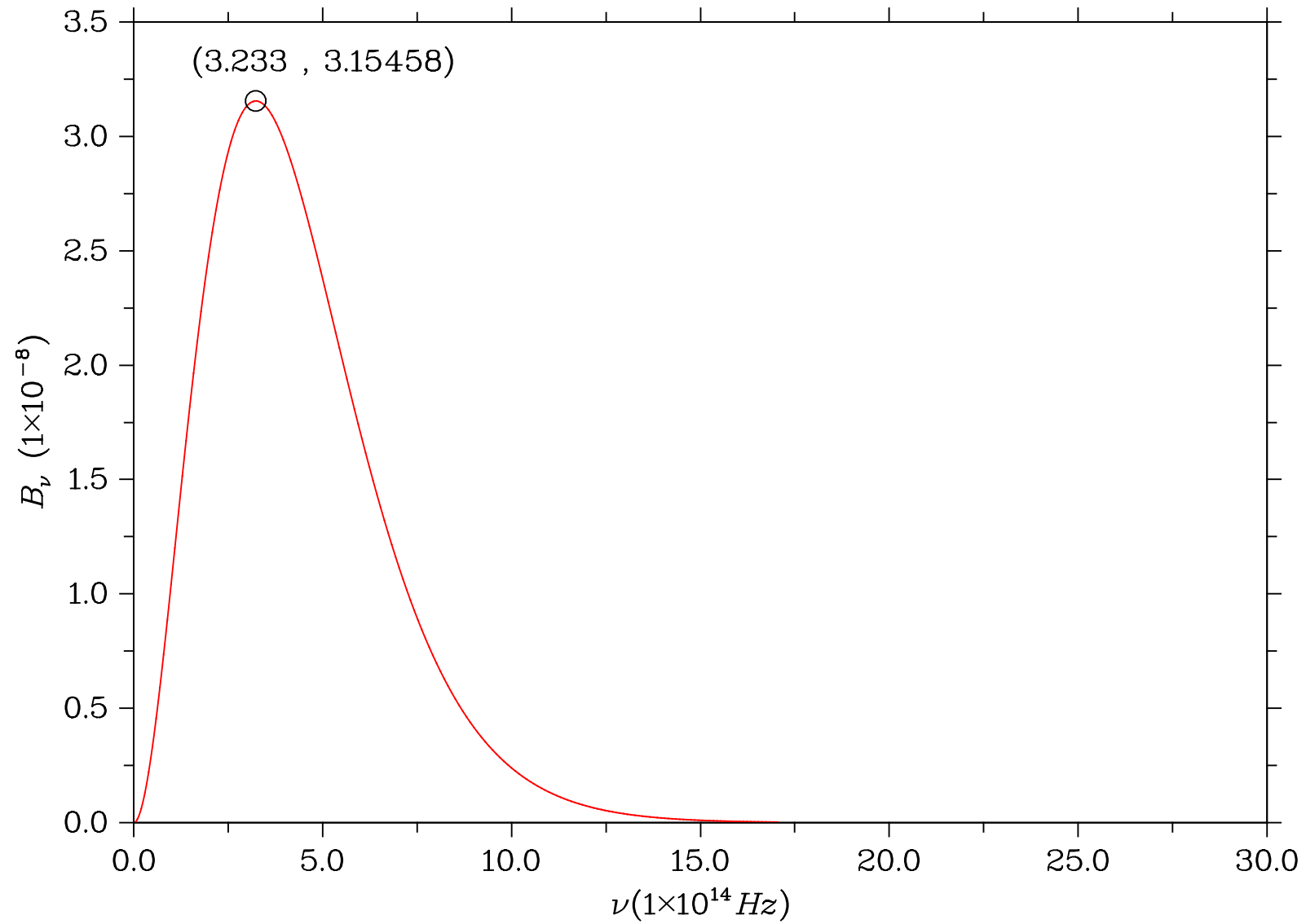
    system("PAUSE");
    return(0);
}

```

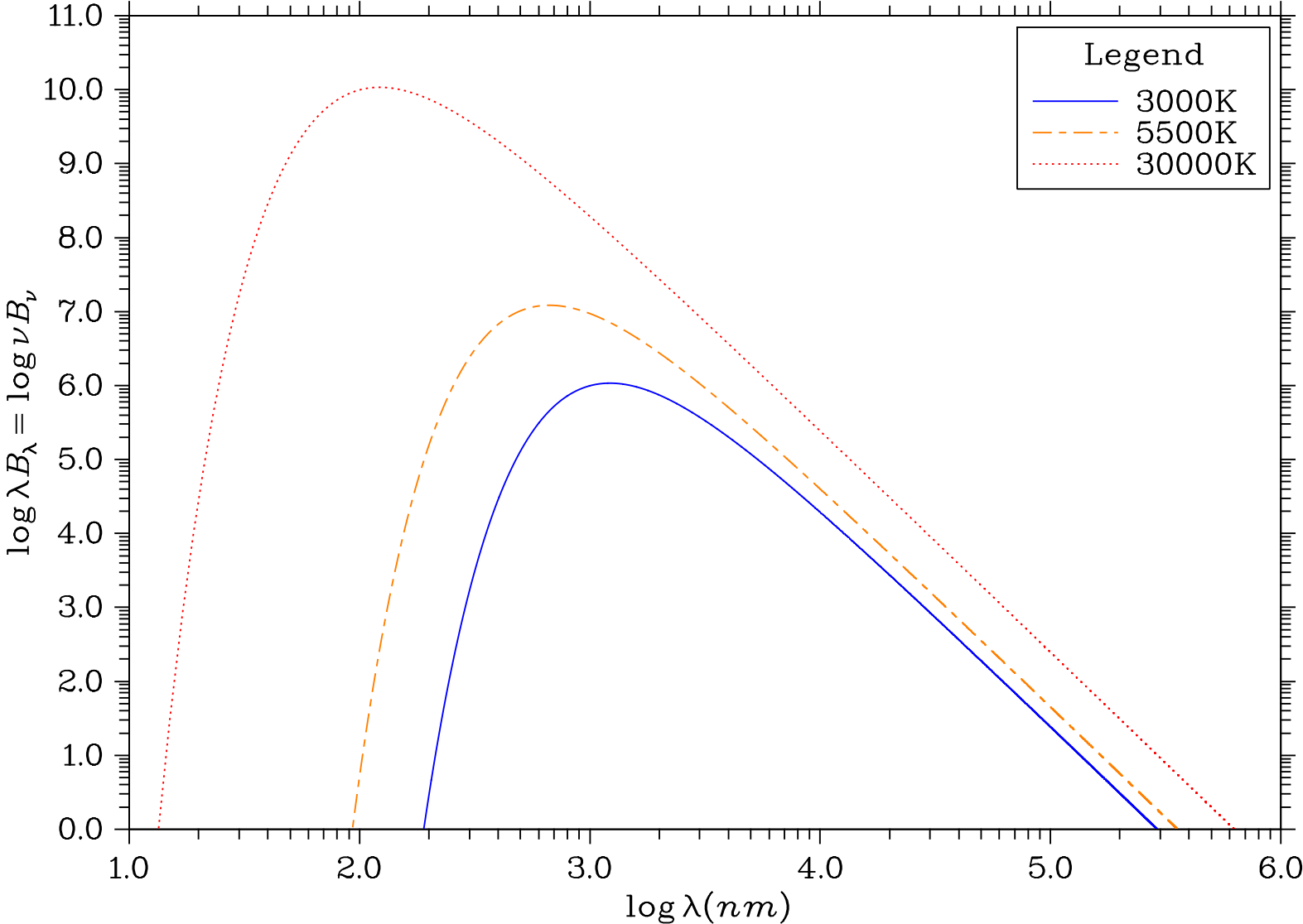
Question 2a



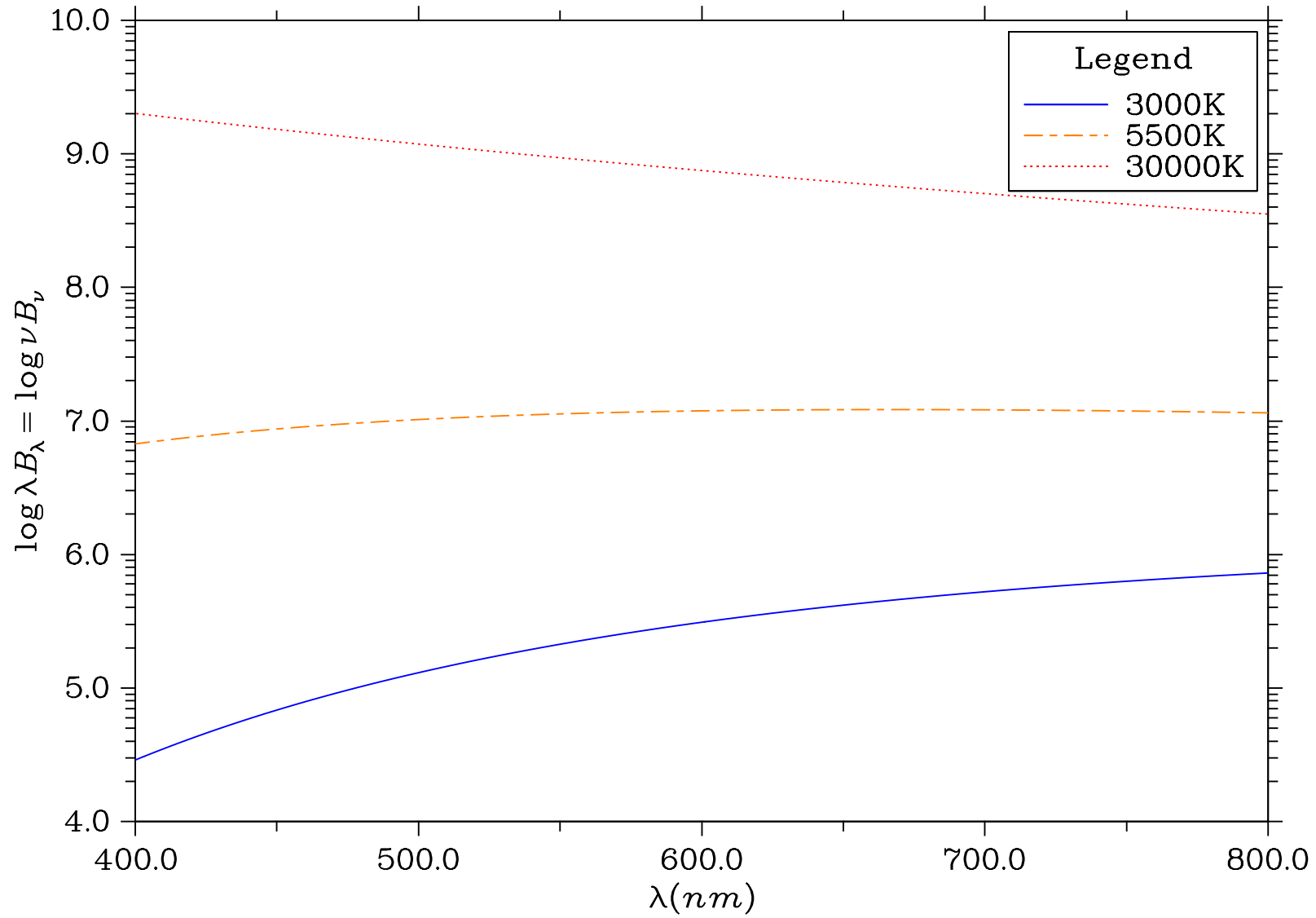
Question 2b



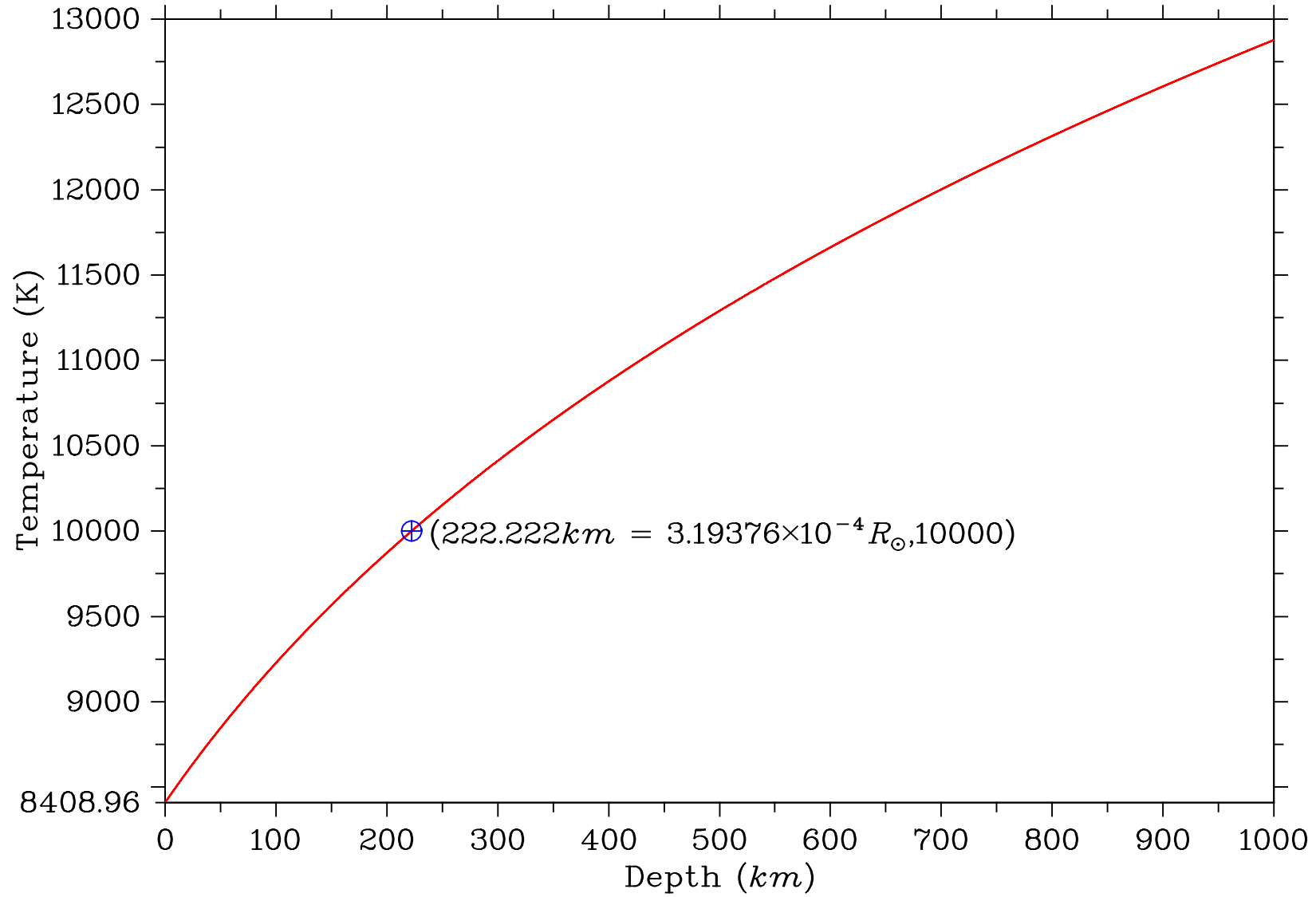
Question 2e – Full



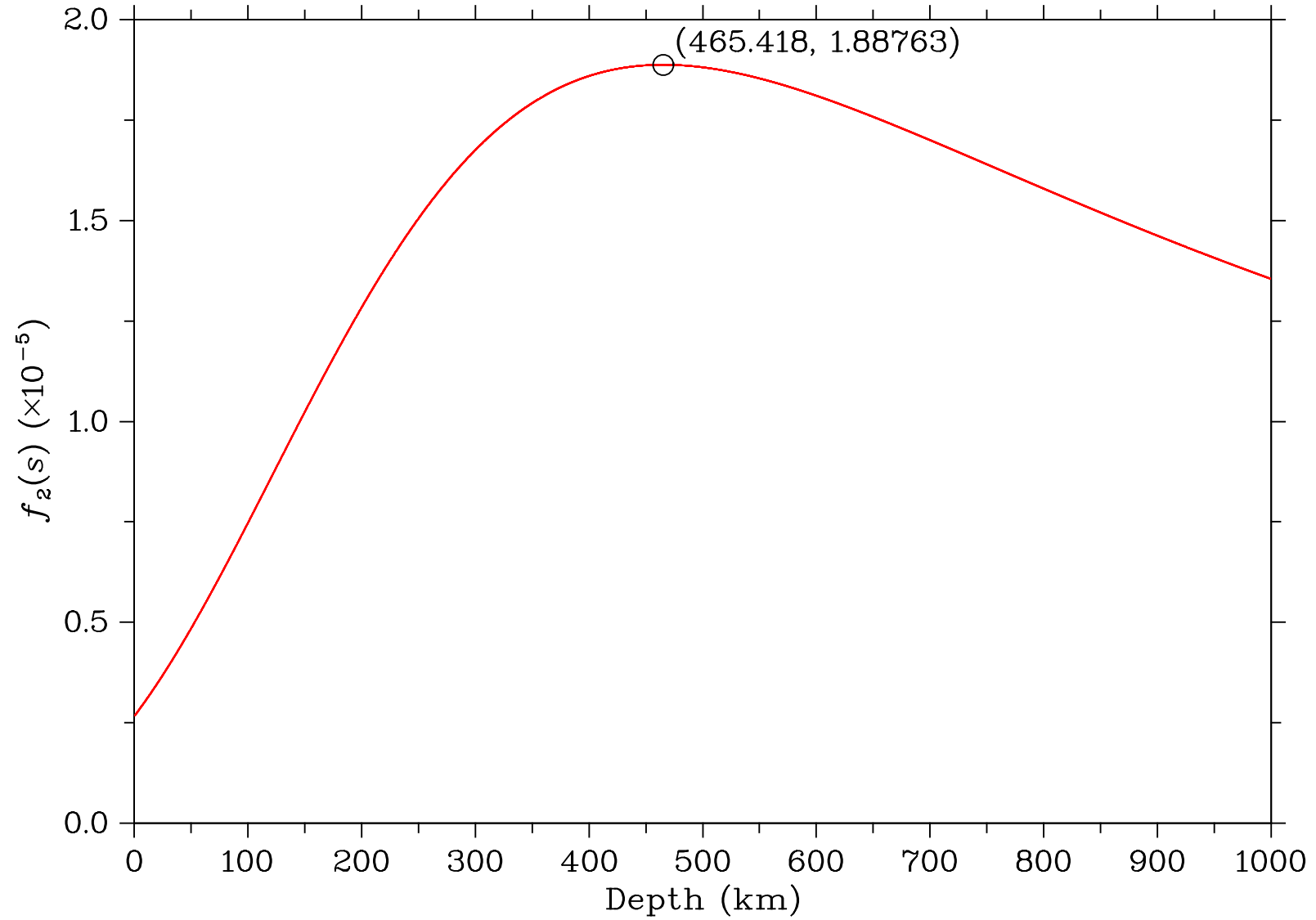
Question 2e – Visible



Question 4a



Question 4b



Question 4c

