APPENDIX D
CHANGE LOG

This page intentionally left blank.

The following DOS functions were copied and pasted to the Windows program as part of the Integrated Exposure Uptake Biokinetic Model for Lead in Children (IEUBK) conversion process. These functions contain the equations used to calculate the exposure, uptake, and biokinetic components. Note that some parameter names were changed in the Windows version (IEUBKwin) [*e.g.*, EXAIR to INAIR, alt_diet to YesNo_AlternativeDiet, alt_water to YesNo_AlternativeWater, uptake to total_uptake, soil_vary to vary_outdoor, dust_vary to vary_indoor, out_air_concentration(t) to air_concentration, user_soil(t) and soil_content(t) to soil_content (t) and alt_soil to m_altsrc without affecting the proper execution of the model source code and are identified in bold text. In addition, some equations were modified as a result of the new functionality in the IEUBKwin model and are identified in italics.

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

1	**FOOD.C
	calc_alternate_diet()**
		meatFraction = 1 - userFishFraction - userGameFraction;
		if (meatFraction < 0.0F) meatFraction = 0.0F;
		
		vegFraction = 1 - userVegFraction;
		fruitFraction = 1 - userFruitFraction;
		
		for(i=0; i < 7; i++)
		{
		InDairy[i] = dairy[i];
		InJuice[i] = juices[i];
		InNuts[i] = nuts[i];
		InBread[i] = bread[i];
		InPasta[i] = pasta[i];
		InBeverage[i] = beverage[i];
		InCandy[i] = candy[i];
		InSauce[i] = sauce[i];
		InFormula[i] = formula[i];
		InInfant[i] = infant[i];
		
		InMeat[i] = meatFraction * meat[i];
		InCanVeg[i] = vegFraction * can_veg[i];
		InFrVeg[i] = vegFraction * f_veg[i];
		InCanFruit[i] = fruitFraction * can_fruit[i];
		InFrFruit[i] = fruitFraction * f_fruit[i];
		InHomeFruit[i] = userFruitFraction *
		home_fruit[i] * UserFruitConc;
		InHomeVeg[i] = userVegFraction * home_veg[i]
		* UserVegConc;
		InFish[i] = userFishFraction * fish[i] *
		UserFishConc;
		InGame[i] = userGameFraction * game[i] *
		UserGameConc;
		
		DietTotal[i] = InDairy[i] + InJuice[i] + InNuts[i]
		+ InBread[i] + InPasta[i] + InBeverage[i]
		+ InCandy[i] + InSauce[i] + InFormula[i]
		+ InInfant[i] + InMeat[i] + InCanVeg[i]
		+ InFrVeg[i] + InCanFruit[i] + InFrFruit[i]
		+ InHomeFruit[i] + InHomeVeg[i]
		+ InFish[i] + InGame[i];
		}
			**DIET.CPP
			Calc_INDIET()**	

2	**INITVALS.C early_biokin_init_vals()**	PBBLD0 = 0.85F * PBBLDMAT	**BASECOMP.CPP Calc_Biokinetic()**	

3	**INITVALS.C
	initial_blood_volumes()**
		for(i=0; i < 85; i++)
		{
		VOLBLOOD[i] = (float) 10.67/(1 + exp(-(i-6.87)/7.09))
		+ 21.86/(1 + exp(-(i-88.15)/26.73));
		VOLPLASM[i] = (float)6.46/(1 + exp(-(i-6.81)/5.74)) +
		8.83/(1 + exp(-(i-65.66)/23.62));
		WTBLOOD[i] = 1.056 * VOLBLOOD[i]/10;
		WTECF[i] = 0.73 * VOLBLOOD[i]/10;
		VOLRBC[i] = (float)4.31/(1 + exp(-(i-6.45)/10.0)) +
		26.47/(1 + exp(-(i-129.61)/25.98));
		VOLECF[i] = VOLBLOOD[i] * 0.73F;
		}
			**BASECOMP.CPP
			Calc_Biokinetic()**	

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

4	**INITVALS.C**
	initial_organ_weights()
		for(i=0; i < 85; i++)
		{
		WTBODY[i] = (float)8.375/(1 + exp(-(i-3.80)/3.60))
		+ 17.261/(1 + exp(-(i-48.76)/20.63));
		WTLIVER[i] = (float)0.261/(1 + exp(-(i-9.82)/3.67))
		+ 0.584/(1 + exp(-(i-55.65)/37.64));
		WTKIDNEY[i] = (float)0.050/(1 + exp(-(i-
		5.24)/4.24)) + 0.106/(1 + exp(-(i-
		65.37)/34.11));
		
		if (i <= 12) WTBONE[i] = 0.111F * WTBODY[i];
		else WTBONE[i] = 0.838F + 0.02F * i;
		
		WTTRAB[i] = 0.20F * WTBONE[i];
		WTCORT[i] = 0.80F * WTBONE[i];
		WTOTHER[i] = WTBODY[i] - WTKIDNEY[i] -
		WTLIVER[i] - WTTRAB[i] - WTCORT[i] -
		WTBLOOD[i] - WTECF[i];
		
		CRKIDBL[i] = (float)(0.777 + 2.35*(1 - exp(-0.0468*i)));
		CRLIVBL[i] = (float)(1.1 + 3.5*(1 - exp(-0.0462*i)) * 1);
		
		CRBONEBL[i] = (float)(6.0 + 215.0*(1 - exp(- 0.000942*i)) * 1);
		CROTHBL[i] = (float)((0.931 + 0.437*(1 - exp(- 0.00749*i))) *
		1);
		}
			BASECOMP.CPP
			
			Calc_Biokinetic()	

5	**INITVALS.C**
	set_residence_times()
		TPLRBC = ResCoef[0];
		RATBLPL = ResCoef[6];
		TRBCPL = (RATBLPL - (0.55/(0.55+0.73))) * TPLRBC;
		
		for(t=0; t < 85; t++)
		{
		TBLUR = ResCoef[1] * pow((double)WTBODY[t]/12.3,
		ALLOMET[1]);
		TBLLIV = ResCoef[2] * pow((double)WTBODY[t]/12.3,
		ALLOMET[2]);
		TBLOTH = ResCoef[3] * pow((double)WTBODY[t]/12.3,
		ALLOMET[3]);
		TBLKID = ResCoef[4] * pow((double)WTBODY[t]/12.3,
		ALLOMET[4]);
		TBLBONE = ResCoef[5] * pow((double)WTBODY[t]/12.3,
		ALLOMET[5]);
		TBONEBL = CRBONEBL[t] * ((WTTRAB[t] + WTCORT[t]) /
		(VOLBLOOD[t]/10)) * TBLBONE;
		
		TPLLIV[t] = TBLLIV / RATBLPL;
		TPLKID[t] = TBLKID / RATBLPL;
		TPLUR[t] = TBLUR / RATBLPL;
		TPLTRAB[t]= TBLBONE / (0.2 * RATBLPL);
		TPLCORT[t]= TBLBONE / (0.8 * RATBLPL);
		TPLOTH[t] = TBLOTH / RATBLPL;
		
		TBLFEC = ResCoef[7] * TBLUR;
		TBLOUT = ResCoef[8] * TBLFEC;
		TKIDPL[t] = TBLKID * CRKIDBL[t] * (WTKIDNEY[t] /
		(VOLBLOOD[t]/10));
		TLIVFEC[t]= CRLIVBL[t] * (WTLIVER[t]/(VOLBLOOD[t]/10)) *
		TBLFEC;
		TOTHOUT[t]= CROTHBL[t] *
		(WTOTHER[t]/(VOLBLOOD[t]/10)) *
		TBLOUT;
		TLIVPL[t] = (TBLLIV * CRLIVBL[t] * WTLIVER[t]) /
		((VOLBLOOD[t]/10) *
		(1 - (TBLLIV/TBLFEC)));
		
		TTRABPL[t] = TBONEBL;
		TCORTPL[t] = TBONEBL;
		
		TOTHPL[t] = (TBLOTH * CROTHBL[t] * WTOTHER[t]) /
		((VOLBLOOD[t]/10) * (1-TBLOTH/TBLOUT));
		}
			BASECOMP.CPP
			Calc_Biokinetic()
				TBLUR = (float)(ResCoef[1] *
				pow(WTBODY[t]/12.300F, ALLOMET[1]));
				TBLLIV = (float)(ResCoef[2] *
				pow(WTBODY[t]/12.300F, ALLOMET[2]));
				TBLOTH = (float)(ResCoef[3] *
				pow(WTBODY[t]/12.300F, ALLOMET[3]));
				TBLKID = (float)(ResCoef[4] *
				pow(WTBODY[t]/12.300F, ALLOMET[4]));
				TBLBONE = (float)(ResCoef[5] *
				pow(WTBODY[t]/12.300F, ALLOMET[5]));

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

6	**NEWLEAD.C**
	biokinetic_model()
		switch (TimeSteps)
		{
		case 0: ns=1.0/6.0; steps=30*6; break; /*..Every 4 Hours..*/
		case 1: ns=1.0/1.0; steps=30*1; break; /*..Daily..*/
		case 2: ns=1.0/24.0; steps=30*24; break; /*..Hourly..*/
		case 3: ns=1.0/96.0; steps=30*96; break; /*..Every 15 minutes..*/
		case 4: ns=1.0/72.0; steps=30*72;break; /*..Every 20 minutes..*/
		case 5: ns=1.0/12.0; steps=30*12; break; /*..Every Two Hours..*/
		case 6: ns=1.0/48.0; steps=30*48; break; /*..Every 30 minutes..*/
		case 7: ns=1.0/8.0; steps=30*8; break; /*..Every 3 Hours..*/
		case 8: ns=1.0/2.0; steps=30*2; break; /*..Every 12 Hours..*/
		case 9: ns=30.0; steps=1; break; /*..Monthly..*/
		default: ns=1.0/6.0; steps=30*6; break;
		}
		MATERNAL:
		/*...determine blood level at birth...*/
		PBBLD0 = 0.85F * PBBLDMAT;
		/*...get the model compartment values at month zero...*/
		MRBC[0] = PBBLD0 * (VOLPLASM[0] + VOLRBC[0]) * (TRBCPL/ns) / ((TRBCPL/ns)+(TPLRBC/ns));
		PBRBC[0] = MRBC[0] / VOLRBC[0];
		EXPR[0] = 1 - (PBRBC[0] / CONRBC);
		RECSUM[0] = (1/(TPLLIV[0]/ns)) + (1/(TPLKID[0]/ns)) + (1/(TPLTRAB[0]/ns)) + (1/(TPLCORT[0]/ns)) + (1/(TPLOTH[0]/ns)) + (1/(TPLUR[0]/ns));
		KPLECF[0] = RECSUM[0] + (EXPR[0]/(TPLRBC/ns));
		MPLECF[0] = (PBBLD0 * (VOLPLASM[0] + VOLRBC[0]) * (TPLRBC/ns) / ((TPLRBC/ns)+(TRBCPL/ns))) * (1.7-HCT0);
		MLIVER[0] = RLIVER0 * PBBLD0 * WTLIVER[0];
		DLIVER[0] = (MPLECF[0] /(TPLLIV[0]/ns)) - MLIVER[0] * (1/(TLIVPL[0]/ns) + 1/(TLIVFEC[0]/ns));
		MKIDNEY[0] = RKIDNEY0 * PBBLD0 * WTKIDNEY[0];
		DKIDNEY[0] = (MPLECF[0]/(TPLKID[0]/ns)) - (MKIDNEY[0] /(TKIDPL[0]/ns));
		MOTHER[0] = ROTHER0 * PBBLD0 * WTOTHER[0];
		DOTHER[0] = (MPLECF[0] /(TPLOTH[0]/ns)) - MOTHER[0] * ((1/(TOTHPL[0]/ns)) + (1/(TOTHOUT[0]/ns)));
		MTRAB[0] = RTRAB0 * PBBLD0 * WTTRAB[0];
		DTRAB[0] = (MPLECF[0]/(TPLTRAB[0]/ns)) - (MTRAB[0] /(TTRABPL[0]/ns));
		MCORT[0] = RCORT0 * PBBLD0 * WTCORT[0];
		DCORT[0] = (MPLECF[0]/(TPLCORT[0]/ns)) - (MCORT[0] /(TCORTPL[0]/ns));
		MPLASM[0] = MPLECF[0] / (1.7 - HCT0);
		PBPLAS[0] = MPLASM[0] / (VOLBLOOD[0] - VOLRBC[0]);
		PBBLOODEND[0] = PBBLD0;
		INFLOW1[0] = (MRBC[0]/(TRBCPL/ns)) + (MLIVER[0] /(TLIVPL[0]/ns)) + (MKIDNEY[0]/(TKIDPL[0]/ns)) + (MOTHER[0]/(TOTHPL[0]/ns)) + (MTRAB[0] /(TTRABPL[0]/ns)) + (MCORT[0]/(TCORTPL[0]/ns));
			BASECOMP.CPP
			Calc_Biokinetic()	

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

6	**NEWLEAD.C**
	biokinetic_model()
	(cont)
		for(i=1; i < 85; i++)
		{
		BLOOD=0.0F;
		for (t=0; t < steps; t++)
		{
		TPLRBC2 = TPLRBC / (1 - (MRBC[0]
		/VOLRBC[i-1])/CONRBC);
		
		TLIVALL = 1 / (1/TLIVPL[i] + 1/TLIVFEC[i]);
		TOTHALL = 1 / (1/TOTHPL[i] + 1/TOTHOUT[i]);
		
		SUM1 = 1/TPLUR[i] + 1/TPLRBC2 + 1/TPLLIV[i]
		+ 1/TPLKID[i] + 1/TPLOTH[i] + 1/TPLTRAB[i]
		+ 1/TPLCORT[i] ;
		
		SUM2 = 1/(TPLRBC2 *(TRBCPL /ns + 1))
		+ 1/(TPLLIV[i] *(TLIVPL[i] /ns + TLIVPL[i]/TLIVALL))
		+ 1/(TPLKID[i] *(TKIDPL[i] /ns + 1))
		+ 1/(TPLOTH[i] *(TOTHPL[i] /ns
		+ TOTHPL[i]/TOTHALL))
		+ 1/(TPLTRAB[i]*(TTRABPL[i]/ns + 1))
		+ 1/(TPLCORT[i]*(TCORTPL[i]/ns + 1));
		
		SUM3 = MRBC[0] / (TRBCPL /ns + 1)
		+ MLIVER[0] / (TLIVPL[i] /ns + TLIVPL[i]/TLIVALL)
		+ MKIDNEY[0] / (TKIDPL[i] /ns + 1)
		+ MOTHER[0] / (TOTHPL[i] /ns + TOTHPL[i]/TOTHALL)
		+ MTRAB[0] / (TTRABPL[i]/ns + 1)
		+ MCORT[0] / (TCORTPL[i]/ns + 1);
		
		MPLECF[1] = (MPLECF[0] + UPTAKE[i]/steps + SUM3)/
		(1 + ns*SUM1 - ns*SUM2);
		
		MRBC[1] = (MRBC[0] + MPLECF[1]*ns/TPLRBC2) /
		(1 + ns/TRBCPL);
		
		MKIDNEY[1] = (MKIDNEY[0] + MPLECF[1]*ns/TPLKID[i])/
		(1 + ns/TKIDPL[i]);
		
		MTRAB[1] = (MTRAB[0] + MPLECF[1]*ns/TPLTRAB[i])/
		(1 + ns/TTRABPL[i]);
		MCORT[1] = (MCORT[0] + MPLECF[1]*ns/TPLCORT[i])/
		(1 + ns/TCORTPL[i]);
		MLIVER[1] = (MLIVER[0] + MPLECF[1]*ns/TPLLIV[i])/
		(1 + ns/TLIVALL);
		MOTHER[1] = (MOTHER[0] + MPLECF[1]*ns/TPLOTH[i])/
		(1 + ns/TOTHALL);
		
		MPLASM[1] = MPLECF[1]*VOLPLASM[i] /(VOLECF[i] +
		VOLPLASM[i]);
		PBPLAS[1] = MPLASM[1] / (VOLBLOOD[i] - VOLRBC[i]);
		BLOOD = BLOOD + (MRBC[1] + MPLASM[1]) /
		VOLBLOOD[i];
		
		if (only == 1)
		{
		urine = (float)MPLECF[1]/(TPLUR[i]/ns);
		feces = (float)MLIVER[0]*(1/(TLIVFEC[i]/ns));
		hair = (float)MOTHER[0]*(1/(TOTHOUT[i]/ns));
		cumU = cumU + UPTAKE[i]/steps;
		excrete = urine + feces + hair;
		cumE = cumE + urine + feces + hair;
		diff = cumU - cumE;
		cumM = (float)(MPLECF[1] + MRBC[1] + MLIVER[1] +
		MKIDNEY[1] + MOTHER[1] + MTRAB[1] +
		MCORT[1]);
		}
			BASECOMP.CPP
			Calc_Biokinetic()
			(cont)	

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

6	**NEWLEAD.C**
	biokinetic_model()
	(cont)
		MPLECF[0] = MPLECF[1];
		MRBC[0] = MRBC[1];
		MLIVER[0] = MLIVER[1];
		MKIDNEY[0] = MKIDNEY[1];
		MOTHER[0] = MOTHER[1];
		MTRAB[0] = MTRAB[1];
		MCORT[0] = MCORT[1];
		INFLOW1[0] = INFLOW1[1];
		DLIVER[0] = DLIVER[1];
		DKIDNEY[0] = DKIDNEY[1];
		DOTHER[0] = DOTHER[1];
		DTRAB[0] = DTRAB[1];
		DCORT[0] = DCORT[1];
		PBBLOODEND[i] = BLOOD/STEPS;
			BASECOMP.CPP
			Calc_Biokinetic()
			(cont)	

7	**RESULTS.C**
	ready_biokin()
		for(t=0; t < 7; t++)
		{
		IndoorConc = 0.01F * indoorpercent * air_concentration[t];
		TWA = ((time_out[t]*air_concentration[t]) +
		((24-time_out[t])*IndoorConc)) / 24;
		INAIR[t] = TWA * vent_rate[t];
		}
			AIR.CPP
			Calc_INAIR()	

8	**RESULTS.C**
	ready_biokin()
		if (**m_YesNo_AlternativeDiet** == 0)
		{
		for(t=0; t < 7; t++)
		{ INDIET[t] = diet_intake[t]; }
		}
		else
		{
		calc_alternate_diet();
		[Refer to Row 1 of Page 1 for the source code]
		for(t=0; t < 7; t++)
		{ INDIET[t] = DietTotal[t]; }
			DIET.CPP
			Calc_INDIET()	

9	**RESULTS.C**
	ready_biokin()
		Windows:
		if (m_YesNo_AlternativeWater == 0)
		{
		for(i=0; i < 7; i++)
		{ INWATER[i] = water_consumption[i] * constant_water_conc; }
		}
		else
		{
		HomeFlushedFraction = 1 - FirstDrawFraction - FountainFraction;
		if (HomeFlushedFraction < 0.0F) HomeFlushedFraction=0.0F;
		for(i=0; i < 7; i++)
		{ INWATER[i] = water_consumption[i] *
		(HomeFlushedConc * HomeFlushedFraction +
		FirstDrawConc * FirstDrawFraction +
		FountainConc * FountainFraction);
		}
		}
			WATER.CPP
			Calc_INWATER()	

10	**RESULTS.C**
	ready_biokin()
		DOS:
		for(t=0; t < 7; t++)
		{ INPAINT[t] = paint_intake[t]; }
		Windows:
		for (int t +0; t<AGE; t++)
		input >> INPAINT[t]
			OTHER.CPP
			Other_Takedata()	

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

11	**RESULTS.C**
	ready_biokin()
		i=0;
		
		for(t=0; t < 85; t++)
		{
		/*...get passive uptakes...*/
		UPDIET[t] = PAFF * ABSF * AVF * INDIET[i];
		UPWATER[t] = PAFW * ABSW * AVW * INWATER[i];
		UPDUST[t] = PAFD * ABSD * AVD * INDUST[i];
		UPDUSTA[t] = PAFD * ABSD * AVD * INDUSTA[i];
		UPSOIL[t] = PAFS * ABSS * AVS * INSOIL[i];
		UPPAINT[t] = PAFP * ABSP * AVP * INPAINT[i];
		
		/*...add in active uptakes...*/
		AVINTAKE = ABSD*INDUST[i] + ABSD*INDUSTA[i] +
		ABSS*INSOIL[i] + ABSF*INDIET[i] +
		ABSP*INPAINT[i] + ABSW*INWATER[i];
		
		SATINTAKE = SATINTAKE2 * (WTBODY[t]/12.3);
		temp = 1 + (AVINTAKE/SATINTAKE);
		
		UPDIET[t] = UPDIET[t] + ((1-PAFF) * ABSF * AVF *
		INDIET[i])/temp;
		UPWATER[t] = UPWATER[t] + ((1-PAFW) * ABSW * AVW *
		INWATER[i])/temp;
		UPDUST[t] = UPDUST[t] + ((1-PAFD) * ABSD * AVD *
		INDUST[i])/temp;
		UPDUSTA[t] = UPDUSTA[t] + ((1-PAFD) * ABSD * AVD *
		INDUSTA[i])/temp;
		UPSOIL[t] = UPSOIL[t] + ((1-PAFS) * ABSS * AVS *
		INSOIL[i])/temp;
		UPPAINT[t] = UPPAINT[t] + ((1-PAFP) * ABSP * AVP *
		INPAINT[i])/temp;
		
		UPAIR[t] = air_absorp[i] * 0.01 * INAIR[i];
		/*...total uptakes and convert to a monthlt value...*/
		UPTAKE[t] = 30 * (UPDIET[t] + UPWATER[t] + UPDUST[t]
		+ UPSOIL[t] + UPDUSTA[t] + UPPAINT[t] + UPAIR[t]);
		i=t/12;
		}
			BASECOMP.CPP
			Calc_UPTAKE()	

12	**RESULTS.C**
	Calc_Yearly_Averages()
		DOS:
		float sum1, sum2, sum3, sum4, sum5, sum6;
		int index;
		
		sum1 = sum2 = sum3 = sum4 = sum5 = sum6= 0.0F;
		
		for(i=6; i < 13; i++)
		{ sum1 += UPAIR[i];
		sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
		sum3 += PBBLOODEND[i];
		sum4 += UPDIET[i];
		sum5 += UPWATER[i];
		sum6 += UPPAINT[i];
		}
		
		air[0] = sum1 / 7;
		soil[0] = sum2 / 7;
		blood[0] = sum3 / 7;
		diet[0] = sum4 / 7;
		water[0] = sum5 / 7;
		paint[0] = sum6 / 7;
		uptake[0] = soil[0] + diet[0] + water[0] + paint[0] + air[0];
		
		index=13;
		for(t=1; t < 7; t++)
		{
		sum1 = sum2 = sum3 = sum4 = sum5 = sum6 = 0.0F;
		
		for (i=index; i < index+12; i++)
		{ sum1 += UPAIR[i];
		sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
		sum3 += PBBLOODEND[i];
		sum4 += UPDIET[i];
		sum5 += UPWATER[i];
		sum6 += UPPAINT[i];
		}
			BASECOMP.CPP
			Calc_UPTAKE()	

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

12	**RESULTS.C**
	Calc_Yearly_Averages()
	(cont)
		DOS:
		air[t] = sum1 / 12;
		soil[t] = sum2 / 12;
		blood[t] = sum3 / 12;
		diet[t] = sum4 / 12;
		water[t] = sum5 / 12;
		paint[t] = sum6 / 12;
		uptake[t] = soil[t] + diet[t] + water[t] + paint[t] + air[t];
		index += 12;
		}
		
		Windows:
		float sum1, sum2, sum3, sum4, sum5;
		int index;
		
		sum1 = sum2 = sum3 = sum4 = sum5 = 0.0F;
		
		for(i=6; i < 13; i++)
		{ sum1 += UPAIR[i];
		sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
		sum3 += UPDIET[i];
		sum4 += UPWATER[i];
		sum5 += UPPAINT[i];
		}
		
		air[0] = sum1 / 7;
		soil[0] = sum2 / 7;
		diet[0] = sum3 / 7;
		water[0] = sum4 / 7;
		paint[0] = sum5 / 7;
		total_uptake[0] *= soil[0] + diet[0] + water[0] + paint[0] + air[0];*
		
		index=13;
		for(t=1; t < 7; t++)
		{
		sum1 = sum2 = sum3 = sum4 = sum5 = 0.0F;
		
		for (i=index; i < index+12; i++)
		{ sum1 += UPAIR[i];
		sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
		sum3 += UPDIET[i];
		sum4 += UPWATER[i];
		sum5 += UPPAINT[i];
		}
		air[t] = sum1 / 12;
		soil[t] = sum2 / 12;
		diet[t] = sum3 / 12;
		water[t] = sum4 / 12;
		paint[t] = sum5 / 12;
		total_uptake[t] *= soil[t] + diet[t] + water[t] + paint[t] + air[t];*
		index += 12;
		}
			BASECOMP.CPP
			Calc_UPTAKE()
			(cont)	

<table>
<tr><th>Row No.</th><th>DOS Module & Function</th><th>DOS Code
(copied and pasted in the Windows program)</th><th>Windows Module & Function</th><th>Changes made in IEUBKwin</th></tr>
<tr><td>13</td><td><b>RESULTS.C</b>
<b>Calc_Yearly_Averages()</b>
<i>(cont)</i></td><td>

DOS:

```
float sum1, sum2, sum3, sum4, sum5, sum6;
int index;

sum1 = sum2 = sum3 = sum4 = sum5 = sum6= 0.0F;

for(i=6; i < 13; i++)
{ sum1 += UPAIR[i];
   sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
   sum3 += PBBLOODEND[i];
   sum4 += UPDIET[i];
   sum5 += UPWATER[i];
   sum6 += UPPAINT[i];
 }

   air[0] = sum1 / 7;
   soil[0] = sum2 / 7;
   blood[0] = sum3 / 7;
   diet[0] = sum4 / 7;
   water[0] = sum5 / 7;
   paint[0] = sum6 / 7;
   uptake[0] = soil[0] + diet[0] + water[0] + paint[0] + air[0];

   index=13;
   for(t=1; t < 7; t++)
   {
      sum1 = sum2 = sum3 = sum4 = sum5 = sum6 = 0.0F;

      for (i=index; i < index+12; i++)
      {    sum1 += UPAIR[i];
           sum2 += UPSOIL[i] + UPDUST[i] + UPDUSTA[i];
           sum3 += PBBLOODEND[i];
           sum4 += UPDIET[i];
           sum5 += UPWATER[i];
           sum6 += UPPAINT[i];
       }

        air[t] = sum1 / 12;
      soil[t] = sum2 / 12;
      blood[t] = sum3 / 12;
      diet[t] = sum4 / 12;
      water[t] = sum5 / 12;
      paint[t] = sum6 / 12;
      uptake[t] = soil[t] + diet[t] + water[t] + paint[t] + air[t];
      index += 12;
   }
```

Windows:

```
register int i;
float sum6;
int index;

sum6=0.0;

for(i=6; i < 13; i++)
{
     sum6 += PBBLOODEND[i];
}

// month 0
blood[0] = sum6 / 7.0f;

index=13;
for(t=1; t < 7; t++)
{
     sum6 = 0.0F;

     for(i=index; i < index+12; i++)
     {
            sum6 += PBBLOODEND[i];
      }

      blood[t] = sum6/ 12.0f;
      index += 12;
}
```

</td><td><b>BASECOMP.CPP</b>
<b>Calc_Yearly_Averages()</b></td><td></td></tr>
</table>

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

14	**RESULTS.C**
	Ready_biokin()
		if ((**m_vary_indoor** == 2) && (**m_altsrc** == 1))
		   calc_alternate_soil();
		   [Refer to Row 15 on page 11 for source code.]
		else calc_default_soil();
		   [Refer to Row 16 on page 11 for source code.]
			Soil2.cpp
			Calc_INSOIL()	

15	**RESULTS.C**
	calc_alternate_soil()
		register int t;
		float DustTotal;
		float OCCUP, SCHOOL, DAYCARE, SECHOME, PAINT;
		
		HouseFraction = 1.0F - OccupFraction - SchoolFraction -
		      DaycareFraction - SecHomeFraction - PaintFraction;
		
		if (HouseFraction < 0.0F) HouseFraction = 0.0F;
		
		   if (**m_vary_outdoor** == 0)
		   {
		         for(t=0; t < 7; t++)
		         {
		            INSOIL[t] = constant_soil_conc * soil_ingested[t] *
		                 (0.01*weight_soil);
		            soil_indoor[t] = (contrib_percent * constant_soil_conc)
		                 + (multiply_factor * air_concentration[t]);
		         }
		   }
		   else /*...if m_vary_outdoor = 1...*/
		   {
		         for (t=0; t < 7; t++)
		         {
		            INSOIL[t] = user_soil[t] * soil_ingested[t] *
		                     (0.01*weight_soil);
		            soil_indoor[t] = (contrib_percent * user_soil[t]) +
		                     (multiply_factor * air_concentration[t]);
		         }
		      }
		
		   for(t=0; t < 7; t++)
		      {
		       DustTotal = soil_ingested[t] * (0.01*(100-weight_soil));
		       INDUST[t] = DustTotal * HouseFraction * soil_indoor[t];
		       OCCUP = DustTotal * OccupFraction * OccupConc;
		       SCHOOL = DustTotal * SchoolFraction * SchoolConc;
		       DAYCARE = DustTotal * DaycareFraction * DaycareConc;
		       SECHOME = DustTotal * SecHomeFraction * SecHomeConc;
		       PAINT = DustTotal * PaintFraction * PaintConc;
		       INDUSTA[t] = OCCUP + SCHOOL + DAYCARE + SECHOME
		            + PAINT;
		      }
			SOIL2.CPP
			Calc_INSOIL()
				Merged the parameters soil_content[t] and user_soil[t] into soil_content[t]
				INSOIL[t] = soil_content[t] * soil_ingested[t]
				                    * (0.010f * weight_soil);
				soil_indoor[t] = (contrib_percent*soil_content[t])
				                         +(multiply_factor * air_concentration[t]

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

16	**RESULTS.C**
	calc_default_soil()
		register int t;
		
		if (**m_vary_indoor** == **2**)
		{
		if (**m_vary_outdoor** == 0)
		{
		for(t=0; t < 7; t++)
		{
		INSOIL[t] = **constant_soil_conc[t]** * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = (contrib_percent *
		constant_soil_conc[t]) +
		(multiply_factor * air_concentration[t]);
		INDUST[t] = soil_indoor[t] * soil_ingested[t] *
		(0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
		}
		else /*...if m_vary_outdoor = 1...*/
		{
		for(t=0; t < 7; t++)
		{
		INSOIL[t] = user_soil[t] * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = (contrib_percent * user_soil[t]) +
		(multiply_factor * air_concentration[t]);
		INDUST[t] = soil_indoor[t] * soil_ingested[t] *
		(0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
		}
		}
		
		if (**m_vary_indoor** == 1)
		{
		if (**m_vary_outdoor** == 0)
		{
		for(t=0; t < 7; t++)
		{
		soil_content[t] = **constant_soil_conc[t]**;
		INSOIL[t] = **constant_soil_conc[t]** * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = **dust_indoor[t]**;
		INDUST[t] = **dust_indoor[t]** * soil_ingested[t] *
		(0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
		}
		else /*...if m_vary_outdoor = 1...*/
		{
		for(t=0; t < 7; t++)
		{
		INSOIL[t] = user_soil[t] * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = user_dust[t];
		INDUST[t] = user_dust[t] * soil_ingested[t] *
		(0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
		}
		}
		
		if (**m_vary_indoor** == **0**)
		{
		if (**m_vary_outdoor** == 0)
		{
		for(t=0; t < 7; t++)
		{
		INSOIL[t] = **constant_soil_conc[t]** * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = **constant_dust_conc[t]**;
		INDUST[t] = **constant_dust_conc[t]** * soil_ingested[t]
		* (0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
			SOIL2.CPP
			Calc_INSOIL()
				Merged the parameters soil_content[t] and user_soil[t] into soil_content[t]
				INSOIL[t] = soil_content[t] * soil_ingested[t]
				* (0.010f * weight_soil);
				soil_indoor[t] = (contrib_percent*soil_content[t])
				+(multiply_factor * air_concentration[t]
				
				Merged the parameters soil_content[t] and user_soil[t] into soil_content[t]
				INSOIL[t] = soil_content[t] * soil_ingested[t]
				* (0.010f * weight_soil);
				soil_indoor[t] = (contrib_percent*soil_content[t])
				+(multiply_factor * air_concentration[t]
				
				Merged the parameters soil_content[t] and user_soil[t] into soil_content[t]
				INSOIL[t] = soil_content[t] * soil_ingested[t]
				* (0.010f * weight_soil);
				soil_indoor[t] = (contrib_percent*soil_content[t])
				+(multiply_factor * air_concentration[t]

Row No.	DOS Module & Function	DOS Code (copied and pasted in the Windows program)	Windows Module & Function	Changes made in IEUBKwin

16	**RESULTS.C**
	calc_default_soil()
	(cont)
		}
		else /*...if m_vary_outdoor = 1...*/
		{ for(t=0; t < 7; t++)
		{
		INSOIL[t] = user_soil[t] * soil_ingested[t] *
		(0.01*weight_soil);
		soil_indoor[t] = **dust_indoor[t]**;
		INDUST[t] = **constant_dust_conc[t]** * soil_ingested[t] * (0.01*(100-weight_soil));
		INDUSTA[t] = 0.0F;
		}
			SOIL2.CPP
			Calc_INSOIL()
			con't
				Merged the parameters soil_content[t] and user_soil[t] into soil_content[t]
				INSOIL[t] = soil_content[t] * soil_ingested[t]
				* (0.010f * weight_soil);
				soil_indoor[t] = (contrib_percent*soil_content[t])
				+(multiply_factor air_concentration[t]

17	**SOIL.C**
	Calc_Avg_Multi_Source()
		DOS:
		int t;
		
		HouseFraction = 1.0F - OccupFraction - SchoolFraction -
		DaycareFraction - SecHomeFraction - PaintFraction;
		if (HouseFraction < 0.0F) HouseFraction = 0.0F;
		
		calc_Soil_Indoor();
		
		AvgMultiSrc=0.0F;
		for(t=0; t < 7; t++)
		{
		AvgMultiSrc += HouseFraction * soil_indoor[t] +
		OccupFraction * OccupConc +
		SchoolFraction * SchoolConc +
		DaycareFraction * DaycareConc +
		SecHomeFraction * SecHomeConc +
		PaintFraction * PaintConc;
		}
		AvgMultiSrc = AvgMultiSrc / 7;
		
		WINDOWS:
		for(int jj=0; jj<AGE; jj++)
		{
		AvgMultiSrc += m_HouseFracPercent/100.0F *
		soil_indoor[jj] + m_OccupFracPercent/ 100.0F *
		m_OccupConc + m_SchoolFracPercent/100.0F *
		m_SchoolConc + m_DaycareFracPercent/100.0F *
		m_DaycareConc + m_SecHomeFracPercent/100.0F *
		m_SecHomeConc + m_PaintFracPercent/100.0F *
		m_PaintConc;
		}
		
		AvgMultiSrc = AvgMultiSrc/AGE;
			MULTI.CPP
			UpdateData()	