```
*************************************************************************;
* program:    Tax Calculator for SimTax Model                           ;
* programmer: Rick Peterson                                             ;
* project:    Washington Excise Tax Microsimulation Model               ;
* date:       Nov 21, 2002                                              ;
*                                                                       ;
* purpose:    Calculates excise taxes and income tax alternatives for   ;
*             input into SimTax.  SimTax is spreadsheet model which     ;
*             allows the users to redesign the Washington Tax System    ;
*-----------------------------------------------------------------------;
* libraries:   extaxmdl - location of excise tax model data sets        ;
*              popsur - location of Washington Population Survey data   ;
*-----------------------------------------------------------------------;
* incoming:    Extaxmdl.taxbase&x Tax base for each imputation group    ;
*              popsur.sps00f04 Washington Population Survey             ;
*              extaxmdl.secondwage - data on relationship between first ;
*              and second earner wages.                                 ;
*-----------------------------------------------------------------------;
* formats:     'Formats for microsimulation model 1'                    ;
*              'Formats for microsimulation model 2'                    ;
*-----------------------------------------------------------------------;
* outgoing:   None                                                      ;
*-----------------------------------------------------------------------;
* reports:    Total tax by tax type                                     ;
*             Average tax by income group                               ;
*             Tax as a percent of income by income group                ;
*-----------------------------------------------------------------------;
* changes:                                                              ;
*                                                                       ;
*-----------------------------------------------------------------------;
* notes:
*************************************************************************;

*-----------------------------------------------------------------------;
*Get household data from WAPOP to merge with tax data;
*-----------------------------------------------------------------------;
Data z;
set popsur.sps00f04;
where pnum=1;
keep id fnlwgt hhinc peopl hhearn99
nwageinc chldrn20 adults21 age hhtype q4p4g;
run;
proc sort data=z;
by id;
run;

*Sum income by income class;

proc summary nway data=z;
class hhinc;
format hhinc incfmtd.;
var hhinc;
weight fnlwgt;
output out=incomebyclass sum=income;
run;

*-----------------------------------------------------------------------;
```

```
*Apply tax rates to taxbase for each imputation group;
*-------------------------------------------------------------------------;
%macro loop1;
%do x=1 %to 7;

Data excisetaxes&x;
set Extaxmdl.taxbase&x;


*Sales tax;
salesrate = .084;
            salestax = salesbase*salesrate;
            alt1tax = .01* alt1base;
            alt2tax = .01* alt2base;

*Alcohol taxes;
      *Liquor sales and liter in container;
      Liquorsalesrate_container = .205;
      AltLiquorsalesrate_container = .205;
      Liquorliterrate = 2.4408;
      AltLiquorliterrate = 2.4408;
            liquorsalestax_container =
liquorsalesbase_container*Liquorsalesrate_container;
            liquorvoltax_container = liquorvolbase_container *
Liquorliterrate;
            Altliquorsalestax_container =
liquorsalesbase_container*AltLiquorsalesrate_container;
            Altliquorvoltax_container = liquorvolbase_container *
AltLiquorliterrate;

      *Liquor sales and liter by the drink;
      Liquorsalesrate_drink = .137;
      AltLiquorsalesrate_drink = .137;
            liquorsalestax_drink =
.15*liquorsalesbase_drink*Liquorsalesrate_drink;
            liquorvoltax_drink = liquorvolbase_drink * Liquorliterrate;
            Altliquorsalestax_drink =
.15*liquorsalesbase_drink*AltLiquorsalesrate_drink;
            Altliquorvoltax_drink = liquorvolbase_drink *
AltLiquorliterrate;

      *Wine tax;
      Wineliterrate = .2292;
      AltWineliterrate = .2292;
            Winelitertax = Wineliterbase*wineliterrate;
            AltWinelitertax = Wineliterbase*altwineliterrate;

      *Beer Tax - rate per 31 gallons;
      Beertaxrate = 8.08;
      AltBeertaxrate = 8.08;
            Beertax = beerbase * beertaxrate;
            AltBeertax = beerbase * altbeertaxrate;

*Insurance Tax;
      Insrate = .02;
      Altinsrate = .02;
            Instax = insrate*insbase;
            AltInstax = altinsrate*insbase;
```

```
*Cigarette and other tobacco products taxes;
      Cigrate = 1.425;
      Altcigrate = 1.425;
            cigtax = cigbase*cigrate;
            Altcigtax = cigbase*altcigrate;
      Othertobrate = 1.294;
      Altothertobrate = 1.294;
            othertobtax = othertobbase*othertobrate;
            altothertobtax = othertobbase*altothertobrate;

*Public Utility Tax;
      ElecPUTrate = .0378+.03201;
      AltElecPUTrate = .0378+.03201;
            ElecPUT = ElecPUTrate*elecbase;
            AltElecPUT = AltElecPUTrate*elecbase;
      NatgasPUTrate = .03852+.03046;
      AltNatgasPUTrate = .03852+.03046;
            NatgasPUT = NatgasPUTrate*naturalgasbase;
            AltNatgasPUT = AltNatgasPUTrate*naturalgasbase;
      WaterseweragePUTrate = (.05029+.03852)/2+((.0535+.17192)/2);
      AltWaterseweragePUTrate = (.05029+.03852)/2+((.0535+.17192)/2);
            WaterseweragePUT = WaterseweragePUTrate*waterseweragebase;
            AltWaterseweragePUT = AltWaterseweragePUTrate*waterseweragebase;
      GarbagePUTrate = .036+.06372;
      AltGarbagePUTrate = .036+.06372;
            GarbagePUT = GarbagePUTrate*garbagebase;
            AltGarbagePUT = AltGarbagePUTrate*garbagebase;
      IntercityPUTrate = .01926;
      AltIntercityPUTrate = .01926;
            IntercityPUT = IntercityPUTrate*intercitybase;
            AltIntercityPUT = AltIntercityPUTrate*intercitybase;
      IntracityPUTrate = .01926;
      AltIntracityPUTrate = .01926;
            IntracityPUT = IntracityPUTrate*intracitybase;
            AltIntracityPUT = AltIntracityPUTrate*intracitybase;

*Gas Tax;
      Gasrate = .23;
      Altgastax = .23;
            Gastax = gasrate*gasbase;
            Altgastax = altgastax*gasbase;

Alcoholtaxes = liquorsalestax_container + liquorvoltax_container +
                        liquorsalestax_drink + liquorvoltax_drink +
Winelitertax + Beertax;
AltAlcoholtaxes = Altliquorsalestax_container + Altliquorvoltax_container +
                        Altliquorsalestax_drink + Altliquorvoltax_drink +
AltWinelitertax + AltBeertax;
Tobaccotaxes = cigtax + othertobtax;
AltTobaccotaxes = Altcigtax + altothertobtax;
UtilityTaxes =  ElecPUT + NatgasPUT + WaterseweragePUT + GarbagePUT +
                        IntercityPUT + IntracityPUT;
AltUtilityTaxes = AltElecPUT + AltNatgasPUT + AltWaterseweragePUT +
                        AltGarbagePUT + AltIntercityPUT + AltIntracityPUT;

TotalExciseTaxes = salestax + Alcoholtaxes + Instax + Tobaccotaxes +
UtilityTaxes + Gastax;
AltTotalExciseTaxes = alt1tax + Altalcoholtaxes + AltInstax +
AltTobaccotaxes +
```

```
AltUtilityTaxes + Altgastax;
run;

*-----------------------------------------------------------------------;
*Merge aggregated tax with household characteristics from WAPOP;
*Run pgm called 'Second Earner Wages for Married Households'
to create data extaxmdl.secondwage;
*-----------------------------------------------------------------------;

data extaxmdl.excisetaxmerged&x;
merge excisetaxes&x z extaxmdl.secondwage;
by id;
if earned2 = . then earned2=0;
households=1;
income=hhinc;

*-----------------------------------------------------------------------;
*Income Tax Calculations;
*-----------------------------------------------------------------------;

*Earned Income Credit Calculation;
if nwageinc in (.A, .D, .N, .R, .S) and hhearn99 not in (.A, .D, .N, .R, .S)
 then nwageinc = hhinc - hhearn99;
if hhearn99 in (.A, .D, .N, .R, .S) and nwageinc not in (.A, .D, .N, .R, .S)
 then hhearn99 = hhinc - nwageinc;
if hhearn99 in (.A, .D, .N, .R, .S) and nwageinc in (.A, .D, .N, .R, .S)
 then hhearn99 = hhinc;

If nwageinc<2450 then do;
      if chldrn20 = 0 then do;
            if 25<= age <65 and 1< hhearn99 < 10711 then
            EarnIncCredit = Min(hhearn99*.0765, 4758*.0765) -
                              Max(0,(hhearn99-6000)*.0765);
            Else EarnIncCredit = 0;
            End;
      if chldrn20 > 0 then do;
            If chldrn20=1 and 1< hhearn99 < 28281 then
                  EarnIncCredit = Min(hhearn99*.34, 7141*.34) -
                              Max(0,(hhearn99-13100)*.1598);
            If chldrn20>1 and 1< hhearn99 < 32121 then
                  EarnIncCredit = Min(hhearn99*.4, 10020*.4) -
                              Max(0,(hhearn99-13100)*.2106);
            Else EarnIncCredit = 0;
            End;
      Else EarnIncCredit = 0;
      End;
Else EarnIncCredit = 0;


*Elderly calculations;

elderly=0;
if age>64 then elderly = 1;
if age>64 and hhtype=1 then elderly=2;
if q4p4g =1 then elderly =1;

*Establish income tax filer type;

If hhtype in (1) then filertype=1;
```

```
If hhtype=2 then do;
      if ranuni(567) < .72 then filertype=2;
      else filertype = 3;
      end;
if hhtype not in (1,2) then filertype=3;

*Itemized deduction calculations;
*Households are selected to be itemizers based on share of
itemizers within each income category;
*Data on itemizers and amount from DOR income tax simulation model;

If filertype=1 then do;
      if hhinc<20000 then do;
            if ranuni(123)<.13 then itemized = 1*hhinc;
            else itemized = 0;
            end;
      if 20000<=hhinc<30000 then do;
            if ranuni(123)<.209 then itemized = .554*hhinc;
            else itemized = 0;
            end;
      if 30000<=hhinc<40000 then do;
            if ranuni(123)<.284 then itemized = .41*hhinc;
            else itemized = 0;
            end;
      if 40000<=hhinc<50000 then do;
            if ranuni(123)<.411 then itemized = .296*hhinc;
            else itemized = 0;
            end;
      if 50000<=hhinc<60000 then do;
            if ranuni(123)<.549 then itemized = .247*hhinc;
            else itemized = 0;
            end;
      if 60000<=hhinc<70000 then do;
            if ranuni(123)<.612 then itemized = .221*hhinc;
            else itemized = 0;
            end;
      if 70000<=hhinc<80000 then do;
            if ranuni(123)<.688 then itemized = .202*hhinc;
            else itemized = 0;
            end;
      if 80000<=hhinc<100000 then do;
            if ranuni(123)<.761 then itemized = .19*hhinc;
            else itemized = 0;
            end;
      if 100000<=hhinc<130000 then do;
            if ranuni(123)<.824 then itemized = .177*hhinc;
            else itemized = 0;
            end;
      if hhinc>=130000 then do;
            if ranuni(123)<.801 then itemized = .072*hhinc;
            else itemized = 0;
            end;
end;

If filertype=2 then do;
      if hhinc<20000 then do;
            if ranuni(123)<.025 then itemized = .735*hhinc;
            else itemized = 0;
            end;
```

```
      if 20000<=hhinc<30000 then do;
            if ranuni(123)<.106 then itemized = .501*hhinc;
            else itemized = 0;
            end;
      if 30000<=hhinc<40000 then do;
            if ranuni(123)<.322 then itemized = .334*hhinc;
            else itemized = 0;
            end;
      if 40000<=hhinc<50000 then do;
            if ranuni(123)<.432 then itemized = .252*hhinc;
            else itemized = 0;
            end;
      if 50000<=hhinc<60000 then do;
            if ranuni(123)<.477 then itemized = .232*hhinc;
            else itemized = 0;
            end;
      if 60000<=hhinc<70000 then do;
            if ranuni(123)<.658 then itemized = .204*hhinc;
            else itemized = 0;
            end;
      if 70000<=hhinc<80000 then do;
            if ranuni(123)<.70 then itemized = .17*hhinc;
            else itemized = 0;
            end;
      if 80000<=hhinc<100000 then do;
            if ranuni(123)<.765 then itemized = .151*hhinc;
            else itemized = 0;
            end;
      if 100000<=hhinc<130000 then do;
            if ranuni(123)<.836 then itemized = .161*hhinc;
            else itemized = 0;
            end;
      if hhinc>=130000 then do;
            if ranuni(123)<.737 then itemized = .102*hhinc;
            else itemized = 0;
            end;
end;

If filertype=3 then do;
      if hhinc<20000 then do;
            if ranuni(123)<.049 then itemized = 1*hhinc;
            else itemized = 0;
            end;
      if 20000<=hhinc<30000 then do;
            if ranuni(123)<.141 then itemized = .40*hhinc;
            else itemized = 0;
            end;
      if 30000<=hhinc<40000 then do;
            if ranuni(123)<.239 then itemized = .297*hhinc;
            else itemized = 0;
            end;
      if 40000<=hhinc<50000 then do;
            if ranuni(123)<.386 then itemized = .271*hhinc;
            else itemized = 0;
            end;
      if 50000<=hhinc<60000 then do;
            if ranuni(123)<.506 then itemized = .219*hhinc;
            else itemized = 0;
            end;
```

```
      if 60000<=hhinc<70000 then do;
            if ranuni(123)<.568 then itemized = .203*hhinc;
            else itemized = 0;
            end;
      if 70000<=hhinc<80000 then do;
            if ranuni(123)<.592 then itemized = .198*hhinc;
            else itemized = 0;
            end;
      if 80000<=hhinc<100000 then do;
            if ranuni(123)<.638 then itemized = .19*hhinc;
            else itemized = 0;
            end;
      if 100000<=hhinc<130000 then do;
            if ranuni(123)<.666 then itemized = .164*hhinc;
            else itemized = 0;
            end;
      if hhinc>=130000 then do;
            if ranuni(123)<.664 then itemized = .085*hhinc;
            else itemized = 0;
            end;
end;

*Alternative Definitions of taxed income;

*Taxedincome = max(0,hhinc);
*Taxedincome = max(0,(hhinc - 5000*peopl));
*Taxedincome = max(0,(hhinc - 20000));

*Calculation allowing variable std ded for two earners;

If filertype in (1) then
      Taxedincome = max(0,hhinc-(7000+min(earned2,3000))
                        -1000*elderly-2900*peopl);
If filertype in (2) then
      Taxedincome = max(0,hhinc-(7000)
                        -1000*elderly-2900*peopl);
If filertype in (3) then
      Taxedincome = max(0,hhinc-(5000)
                        -1000*elderly-2900*peopl);

*Calculation allowing fed std ded or itemization plus
fed personal exemption estimated for 2005;

*If filertype in (1) then
      Taxedincome = max(0,hhinc-max(8350,itemized)
                        -3175*peopl);
*If filertype in (2) then
      Taxedincome = max(0,hhinc-max(7350,itemized)
                        -3175*peopl);
*If filertype in (3) then
      Taxedincome = max(0,hhinc-max(4975,itemized)
                        -3175*peopl);

*Flat Rate Income Tax Calculation;

FlatrateIncTax=1*Taxedincome;

*FlatrateIncTax=max(0,.05*Taxedincome-EarnIncCredit*.1);
```

```
*Graduated Rate Income Tax;

Rate1=.02;
Rate2=.03;
Rate3=.05;
If filertype in (1) then do;
      Bracket1 = min(Taxedincome,49900);
      Bracket2 = min(120650-49900,max(0,Taxedincome-49900));
      Bracket3 = max(0,Taxedincome-120650);
End;
If filertype in (2) then do;
      Bracket1 = min(Taxedincome,37425);
      Bracket2 = min(90487.5-37425,max(0,Taxedincome-37425));
      Bracket3 = max(0,Taxedincome-90487.5);
End;
If filertype not in (1,2) then do;
      Bracket1 = min(Taxedincome,24950);
      Bracket2 = min(60325-24950,max(0,Taxedincome-24950));
      Bracket3 = max(0,Taxedincome-60325);
End;

GradRateIncTax = Rate1*Bracket1+Rate2*Bracket2+Rate3*Bracket3;
*GradRateIncTax = Max(0,Rate1*Bracket1+Rate2*Bracket2
                        +Rate3*Bracket3-EarnIncCredit*.1);

drop _freq_ _type_;
run;
%end;
%mend loop1;
%loop1;

OPTIONS ls=123 ps=52 pageno=1;

*-----------------------------------------------------------------------;
*Calculate total tax by tax type;
*-----------------------------------------------------------------------;

%macro loop2;
%do z=1 %to 7;
proc summary  data=extaxmdl.excisetaxmerged&z;
weight fnlwgt;
var salestax alt1tax alt2tax liquorsalestax_container liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax households EarnIncCredit GradRateIncTax Bracket1 Bracket2
Bracket3
FlatrateIncTax peopl;
output out=weightedtax&z sum=;
run;
%end;
%mend loop2;
%loop2;
```

```
*Average the data from the seven imputation groups;

Data one;
set weightedtax1 weightedtax2 weightedtax3 weightedtax4
weightedtax5 weightedtax6 weightedtax7;
proc summary  data=one;
var salestax alt1tax alt2tax liquorsalestax_container liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax households EarnIncCredit GradRateIncTax Bracket1 Bracket2
Bracket3
 FlatrateIncTax peopl;
output out=Aveweightedtax mean=;
run;
Title1 "Total tax amounts for all groups (Average)";
proc print data=Aveweightedtax;
var salestax alt1tax alt2tax liquorsalestax_container liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax EarnIncCredit GradRateIncTax Bracket1 Bracket2 Bracket3
 FlatrateIncTax peopl;
format salestax alt1tax alt2tax liquorsalestax_container
liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax EarnIncCredit GradRateIncTax Bracket1 Bracket2 Bracket3
 FlatrateIncTax peopl comma15.;
run;

*-----------------------------------------------------------------------;
*Calculate Tax by income group for each imputation group;
*-----------------------------------------------------------------------;

OPTIONS ls=130 ps=50 pageno=1;
%macro loop3;
%do z=1 %to 7;
proc summary nway data=extaxmdl.excisetaxmerged&z;
class hhinc ;
format hhinc incfmtd.;
weight fnlwgt;
```

```
var TotalExciseTaxes AltTotalExciseTaxes Alcoholtaxes AltAlcoholtaxes
Tobaccotaxes
AltTobaccotaxes UtilityTaxes AltUtilityTaxes
salestax alt1tax alt2tax liquorsalestax_container liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax households EarnIncCredit GradRateIncTax Bracket1 Bracket2
Bracket3
 FlatrateIncTax peopl;
output out=weightedtaxbyinc&z sum=;
run;
%end;
%mend loop3;
%loop3;

*Add tax from each group and take average;

Data two;
set weightedtaxbyinc1 weightedtaxbyinc2 weightedtaxbyinc3
weightedtaxbyinc4 weightedtaxbyinc5 weightedtaxbyinc6
weightedtaxbyinc7;
run;
Proc summary data = two nway;
class hhinc;
format hhinc incfmtd.;
var TotalExciseTaxes AltTotalExciseTaxes Alcoholtaxes AltAlcoholtaxes
Tobaccotaxes
AltTobaccotaxes UtilityTaxes AltUtilityTaxes
salestax alt1tax alt2tax liquorsalestax_container liquorvoltax_container
Altliquorsalestax_container Altliquorvoltax_container liquorsalestax_drink
liquorvoltax_drink Altliquorsalestax_drink Altliquorvoltax_drink
Winelitertax AltWinelitertax Beertax AltBeertax Instax AltInstax
cigtax Altcigtax othertobtax altothertobtax
ElecPUT AltElecPUT NatgasPUT AltNatgasPUT WaterseweragePUT
AltWaterseweragePUT
GarbagePUT AltGarbagePUT IntercityPUT AltIntercityPUT IntracityPUT
AltIntracityPUT
Gastax Altgastax households EarnIncCredit GradRateIncTax Bracket1 Bracket2
Bracket3
 FlatrateIncTax peopl;
output out=Aveweightedtaxbyinc mean=;
run;

*Divide by number of households by income group
and calc avg tax per household;

data avgtax;
set Aveweightedtaxbyinc;
AvgTotalExciseTaxes = TotalExciseTaxes/households;
AvgAltTotalExciseTaxes = AltTotalExciseTaxes/households;
Avgsalestax = salestax/households;
Avgalt1tax = alt1tax/households;
Avgalt2tax = alt2tax/households;
```

```
avgAlcoholtaxes = Alcoholtaxes/households;
avgAltAlcoholtaxes = AltAlcoholtaxes/households;
avgInstax = instax/households;
avgAltinstax = altinstax/households;
avgTobaccotaxes = Tobaccotaxes/households;
avgAltTobaccotaxes = AltTobaccotaxes/households;
avgUtilityTaxes = UtilityTaxes/households;
avgAltUtilityTaxes = AltUtilityTaxes/households;
avgGastax = gastax/households;
avgAltgastax = altgastax/households;
avgEarnIncCredit= EarnIncCredit/households;
avgGradRateIncTax= GradRateIncTax/households;
avgBracket1= Bracket1/households;
avgBracket2= Bracket2/households;
avgBracket3= Bracket3/households;
avgFlatrateIncTax= FlatrateIncTax/households;
avgpeopl = peopl/households;
drop _freq_ _type_;
run;

title1 "Average Tax for all Groups (Average)" ;

Proc tabulate data = avgtax;
class hhinc;
format hhinc incfmtd.;
var AvgTotalExciseTaxes AvgAltTotalExciseTaxes Avgsalestax Avgalt1tax
Avgalt2tax
avgAlcoholtaxes avgAltAlcoholtaxes avgInstax avgAltinstax
avgTobaccotaxes avgAltTobaccotaxes
avgUtilityTaxes avgAltUtilityTaxes avgGastax avgAltgastax
avgEarnIncCredit avgGradRateIncTax avgBracket1 avgBracket2 avgBracket3
 avgFlatrateIncTax avgpeopl;
table hhinc,(AvgTotalExciseTaxes AvgAltTotalExciseTaxes Avgsalestax
Avgalt1tax Avgalt2tax
avgAlcoholtaxes avgAltAlcoholtaxes avgInstax avgAltinstax
avgTobaccotaxes avgAltTobaccotaxes
avgUtilityTaxes avgAltUtilityTaxes avgGastax avgAltgastax
avgEarnIncCredit avgGradRateIncTax avgBracket1 avgBracket2 avgBracket3
 avgFlatrateIncTax)*f=comma10. avgpeopl*f=comma10.2;
run;

*Merge in total income by income group
and calc avg tax as percent of income;

data avgtax;
merge aveweightedtaxbyinc incomebyclass;
by hhinc;
AvgTotalExciseTaxes = TotalExciseTaxes/income;
AvgAltTotalExciseTaxes = AltTotalExciseTaxes/income;
Avgsalestax = salestax/income;
Avgalt1tax = alt1tax/income;
Avgalt2tax = alt2tax/income;
avgAlcoholtaxes = Alcoholtaxes/income;
avgAltAlcoholtaxes = AltAlcoholtaxes/income;
avgInstax = instax/income;
avgAltinstax = altinstax/income;
avgTobaccotaxes = Tobaccotaxes/income;
avgAltTobaccotaxes = AltTobaccotaxes/income;
avgUtilityTaxes = UtilityTaxes/income;
```

```
avgAltUtilityTaxes = AltUtilityTaxes/income;
avgGastax = gastax/income;
avgAltgastax = altgastax/income;
avgEarnIncCredit= EarnIncCredit/income;
avgGradRateIncTax= GradRateIncTax/income;
avgBracket1= Bracket1/income;
avgBracket2= Bracket2/income;
avgBracket3= Bracket3/income;
avgFlatrateIncTax= FlatrateIncTax/income;
drop _freq_ _type_;
run;


title1 "Tax as Percent of Income for all Groups (Average)" ;

Proc tabulate data = avgtax ;
class hhinc;
format hhinc incfmtd.;
var AvgTotalExciseTaxes AvgAltTotalExciseTaxes Avgsalestax Avgalt1tax
Avgalt2tax
avgAlcoholtaxes avgAltAlcoholtaxes avgInstax avgAltinstax
avgTobaccotaxes avgAltTobaccotaxes
avgUtilityTaxes avgAltUtilityTaxes avgGastax avgAltgastax
avgEarnIncCredit avgGradRateIncTax avgBracket1 avgBracket2 avgBracket3
 avgFlatrateIncTax;
table hhinc,(AvgTotalExciseTaxes AvgAltTotalExciseTaxes Avgsalestax
Avgalt1tax Avgalt2tax
avgAlcoholtaxes avgAltAlcoholtaxes avgInstax avgAltinstax
avgTobaccotaxes avgAltTobaccotaxes
avgUtilityTaxes avgAltUtilityTaxes avgGastax avgAltgastax
avgEarnIncCredit avgGradRateIncTax avgBracket1 avgBracket2 avgBracket3
 avgFlatrateIncTax)*f=percent10.2;
run;


*************************************************************************;
* program:    Second Earner Wages for Married Households                 ;
* programmer: Rick Peterson                                              ;
* project:    Washington Excise Tax Microsimulation Model                ;
* date:       Nov 21, 2002                                               ;
*                                                                        ;
* purpose:    Calculates relationship between first and second earner    ;
*             wages for married households.  Creates frequency           ;
*             distribution for use in the Department of Revenue's        ;
*             Income Tax Model and data set for use in 'Tax Calculator   ;
*             for SimTax Model.                                          ;
;
*-----------------------------------------------------------------------;
* libraries:   extaxmdl - location of excise tax model data sets         ;
*              popsur - location of Washington Population Survey data    ;
*-----------------------------------------------------------------------;
* incoming:    popsur.sps00f04 Washington Population Survey              ;
*-----------------------------------------------------------------------;
* formats:     'Formats for microsimulation model 1'                     ;
*              'Formats for microsimulation model 2'                     ;
*-----------------------------------------------------------------------;
* outgoing:    extaxmdl.secondwage - for use in 'Tax Calculator for      ;
*              SimTax Model'                                             ;
*-----------------------------------------------------------------------;
```

```
* reports:     Frequency Distributin of Ratio of Second Wage Earner's     ;
*              Wage to 1st Earner's Wage by income class                  ;
*-------------------------------------------------------------------------;
* changes:                                                                ;
*                                                                         ;
*-------------------------------------------------------------------------;
* notes:
***************************************************************************;

*-------------------------------------------------------------------------;
*Read married household data from WAPOP;
*-------------------------------------------------------------------------;

Data x;
set popsur.sps00f04;
Where hhtype =1;
earned = 0;
if q6p1a not in (.A, .D, .N, .R, .S) and q6p4a not in (.A, .D, .N, .R, .S)
 then earned = q6p1a + q6p4a;
if q6p1a in (.A, .D, .N, .R, .S) and q6p4a not in (.A, .D, .N, .R, .S)
 then earned =  q6p4a;
if q6p1a not in (.A, .D, .N, .R, .S) and q6p4a in (.A, .D, .N, .R, .S)
 then earned = q6p1a;

keep id pnum fnlwgt hhearn99
q2p10 hhtype earned q6p1a q6p4a hhinc nwageinc;
run;

Proc sort data=x out=xx;
by id decending earned ;
run;

data xxx a;
set xx ;
by id;
if first.id eq 1 then output a;
if first.id ne 1 then output xxx;
run;

Proc summary data =xxx;
by id;
var earned ;
output out=xxxx sum=earned2;
run;

data aa aaa;
merge a xxxx;
by id;
if earned > 0 then do;
      ratio = earned2/earned;
      output aa;
      end;
else do;
      ratio = 0;
      output aaa;
      end;
label ratio='Ratio of 2nd earner wage to 1st earner wage';
run;
Title1 "Ratio of Second Wage Earner's Wage to 1st Earner's Wage";
```

```
footnote;
Proc format;
 value ratiofmt
                    low -<.01 = 'Zero'
                     .01-<.25 ='Zero+ to 25%'
              .25-<.5   ='25% to 50%'
              .5-<.75   ='50% to 75%'
              .75-high  ='Over 75%';
                    run;

Title1 h=2 "Ratio of Second Wage Earner's Wage to 1st Earner's Wage";
footnote;

Proc freq data=aa;
table hhinc*ratio / nocol nocum nofreq out=earn2ratio;
weight fnlwgt;
format hhinc incfmtd.;
format ratio ratiofmt.;
run;

*-----------------------------------------------------------------------;
*Create data set to assign earned income to 1st
and 2nd worker in married households without wage data;
*-----------------------------------------------------------------------;

Proc summary data=earn2ratio;
by hhinc;
format hhinc incfmtd.;
var count;
output out=sumearn sum=total;
run;


data bb;
Merge earn2ratio sumearn;
by hhinc;
format hhinc incfmtd.;
ratio2=count/total;
keep hhinc ratio2 ratio;
run;

*Transpose the data making each percentage a variable;
*Each percentage represents the share of households with ratios
 equal to the ranges assigned to ratiofmt above;

Proc transpose  name=prob prefix=p data=bb out=bbb;
var ratio2;
run;

*Merge the prob data with the married households without
data on wage earnings;
*Assign earnings for 1st earner and 2nd earner based on
above data;

data B;
if _n_=1 then set bbb;
set aaa;
array r{5} r1-r5 (0 .125 .375 .625 .875);
retain p1-p50 r1-r5;
```

```
if nwageinc not in (.A, .D, .N, .R, .S)
then hhearn99 = hhinc - nwageinc;
else hhearn99 = hhinc;

if hhinc<20000 then do;
y = rantbl(25,of p1-p5);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 20000<=hhinc<30000 then do;
y = rantbl(25,of p6-p10);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 30000<=hhinc<40000 then do;
y = rantbl(25,of p11-p15);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 40000<=hhinc<50000 then do;
y = rantbl(25,of p16-p20);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 50000<=hhinc<60000 then do;
y = rantbl(25,of p21-p25);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 60000<=hhinc<70000 then do;
y = rantbl(25,of p26-p30);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 70000<=hhinc<80000 then do;
y = rantbl(25,of p31-p35);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if 80000<=hhinc<100000 then do;
y = rantbl(25,of p36-p40);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
```

```
end;

if 100000<=hhinc<130000 then do;
y = rantbl(25,of p41-p45);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if hhinc>=130000 then do;
y = rantbl(25,of p46-p50);
Ratio= r(y);
earned=hhearn99/(1+ratio);
earned2 = ratio*earned;
end;

if nwageinc not in (.A, .D, .N, .R, .S) then do;
      test = hhinc-nwageinc-earned-earned2;
      test2 = hhearn99 - earned - earned2;
      end;
else do;
      test = hhinc-earned-earned2;
      test2 = hhearn99 - earned - earned2;
      end;
keep id hhinc hhearn99 nwageinc earned earned2
test test2;
run;

*Create data set for married families with earnings for
2nd earner for use in Tax Calculator for SimTax Model;

data extaxmdl.secondwage;
set aa b;
keep id earned2;
run;

Proc sort data=extaxmdl.secondwage;
by id;
run;
```

Ratio of Second Wage Earner's Wage to 1st Earner's Wage
The FREQ Procedure

Percent Row Pct

Table of HHINC by ratio						
HHINC(1999 HOUSEHOLD TOTAL INCOME)	ratio(Ratio of 2nd earner wage to 1st earner wage)					Total
	Zero	Zero+ to 25%	25% to 50%	50% to 75%	Over 75%	
$0 to $20,000	3.03 81.19	0.08 2.20	0.33 8.86	0.24 6.35	0.05 1.39	3.73
$20,000 to $30,000	3.50 65.07	0.96 17.82	0.31 5.83	0.22 4.15	0.38 7.13	5.38
$30,000 to $40,000	4.79 58.53	1.12 13.69	0.88 10.80	0.38 4.62	1.01 12.36	8.18
$40,000 to $50,000	4.52 40.25	2.21 19.67	1.42 12.68	1.55 13.81	1.52 13.58	11.22
$50,000 to $60,000	4.59 38.34	1.33 11.14	2.25 18.81	1.41 11.79	2.38 19.92	11.96
$60,000 to $70,000	3.36 29.81	1.57 13.93	1.40 12.40	2.34 20.70	2.61 23.17	11.29
$70,000 to $80,000	2.83 28.90	1.78 18.12	1.53 15.62	0.98 9.98	2.69 27.38	9.81
$80,000 to $100,000	3.11 21.18	1.78 12.14	2.17 14.80	3.01 20.47	4.62 31.42	14.69
$100,000 to $130,000	2.50 22.15	1.22 10.84	1.51 13.40	2.56 22.72	3.49 30.89	11.29
Over $130,000	4.00 32.15	2.20 17.70	2.58 20.70	1.52 12.24	2.14 17.21	12.45
Total	298572 36.24	117500 14.26	118603 14.39	117073 14.21	172179 20.90	823927 100.00