Statistics/Data analysis

```
use "${preSimulationData}\\01_${countryName}_${simulationName}_${cleanedSurveyData_d
> em_inc}.dta", clear


*******************************************************************************
codebook hhID memberID
isid hhID memberID

merge m:m hhID using "${preSimulationData}\\02_${countryName}_${simulationName}_${cl
> eanedSurveyData_exp}.dta", keepusing(pl_abs pl_food)          // importing local p
> overty lines
drop _merge

/* Returning back to the original number of households/members since there are a lar
> ge # of expenditure obs per household/members in the expenditure dataset */
bysort hhID memberID: keep if _n==1

codebook hhID memberID
isid hhID memberID
isid hhID memberID
*******************************************************************************


* dealing with -ve income (eg, entr_inc/business loss), renaming market income to ne
> t, and assigning their missing values to zero
foreach z in $market_income {
        replace `z' =0  if `z'<0                // if income is negative, then make
> it zero
        rename `z' `z'_net
        replace `z'_net = 0 if missing(`z'_net) // Both missing and zero incomes are
>  assumed to be zero. Any implication on results' interpretation?
}

gen unem_ben = 500              // This is for the time being, since I couldn't find
>  this var yet. DELETE this line later!

* hh size (we count only those who are present)
bysort hhID: egen double hh_size_test = count(memberID)
        assert hh_size == hh_size_test // if !mi(ind_weight)


* age
assert !mi(age)                                                         // ensuring
> age is not missing

gen kid_age=(age<18) if  !mi(ind_weight)
gen pens_age=(age>64) if !mi(ind_weight)
gen work_age=1-abs(kid_age-pens_age) if !mi(ind_weight)


* number of members
foreach var in kid_age work_age pens_age  {
bysort hhID: egen double n_`var'=total(`var')
}

assert n_kid_age+n_work_age+n_pens_age == hh_size if !mi(ind_weight)
```

```
**household type
gen hh_type=.
replace hh_type=1 if n_kid_age==1 & n_work_age+n_pens_age>=2
replace hh_type=2 if n_kid_age==2 & n_work_age+n_pens_age>=2
replace hh_type=3 if n_kid_age>2 & n_work_age+n_pens_age>=2
replace hh_type=4 if n_kid_age>0 & n_work_age+n_pens_age<2
replace hh_type=5 if n_work_age==hh_size //& n_pens_age==0 & n_kid_age==0
replace hh_type=6 if n_pens_age==hh_size //& n_work_age==0 & n_kid_age==0
replace hh_type=7 if n_work_age+n_pens_age==hh_size & n_pens_age>0 & n_work_age>0 //
> & n_kid_age==0
assert !missing(hh_type)

#delimit ;
label def hh_type
1       "two adults (or more) and 1 child"
2       "two adults (or more) and 2 children"
3       "two adults (or more) and 3+ children"
4       "one adult and child(ren)"
5       "only working age adults"
6       "only pensioners"
7       "mixed adults, no children"
, replace ;
#delimit cr
la val hh_type hh_type
label variable hh_type "Household composition"

*Poverty lines - local/national from the hh survey data
sum pl_abs                                                      // Absolute poverty
> line using food poverty line and Ravallion method (lower bound), annualized
gen double povline_nat = `r(mean)'
label variable povline_nat "National poverty line per capita for 12 months"

sum pl_food                                                     // Food poverty line
> , annualized
gen double povline_nat_food = `r(mean)'
label variable povline_nat_food "National food-poverty line per capita for 12 months
> "

/*              * Will do this later!
*Poverty lines - international
global icp_2011 = 4 // put values for your country
global cpi_cumulative = 2 // cumulative CPI between 2011 and Survey Year (SY)

gen double povline_int32 = 3.2 * 365 * ${icp_2011} * ${cpi_cumulative}
gen double povline_int55 = 5.5 * 365 * ${icp_2011} * ${cpi_cumulative}

label variable povline_int32 "International povert line 3.2 USD/day in 2011 PPP, ann
> ualized"
label variable povline_int55 "International povert line 5.5 USD/day in 2011 PPP, ann
> ualized"
*/

isid hhID memberID

des, full
sum
save "${simulationData}\01_${countryName}_${simulationName}_${dem_inc_SY}.dta", repl
> ace

keep hhID memberID ${dem_list}
order hhID memberID ${dem_list}
```

```
des, full
sum
```