```
/*************************************************************************;
* This file computes variables required for the trading strategy tests  *;
* NOTE - this file only includes size and BTM as characteristics        *;
*           Last updated: 31 August 2018                                *;
*************************************************************************;

libname home 'YOUR_DIR';
libname char "YOUR_DIR";
libname master "YOUR_DIR";


*************************************************************************;
** CAR: [y,xx] return minus size-btm matched portfolio return
** Form size breakpoints based on American exchanges and industry-
** adjusted BTM
*************************************************************************;

* Let's do quintiles for portfolio formation *;
%let size = 5;
%let btm_break = 5;

* This file is divided into the following parts:
* I. The Size and BTM portfolios for a given date range.
* II. Determine the portfolio for each firm-quarter and find the long-term (i.e.,
monthly) returns for the portfolio
* III. The short-term (i.e., daily) returns for the portfolio
* IV. The short-run and long-run returns for the sample firms
* V. Calculate the BHAR over the specified ranges;

*********************************************************************************
*
I. Allocate all firms into Size/BTM portfolios for a given date range
*********************************************************************************
*;

rsubmit;
proc sql;
      create table size as select
            date, permno, hexcd, shrout, prc
      from crsp.msf
      /* Specify exchanges */
      where hexcd in (1,2,3)
      /* Our sample begins in 2012, so pull in everything on or after 2010 to be
safe */
      and year(date) >= 2010
      order by permno, date;
quit;

* Ensure no duplicates *;
proc sort data=size out=size nodupkey; by permno date; run;

proc download data=size out=size; run;
endrsubmit;

* Calculate market value and create quintile breakpoint for each month *;
data size; set size;
      mkt_val = abs(prc)*shrout;
run;
proc sort data=size; by date mkt_val; quit;

* Create size quintiles *;
```

```
proc rank data=size out=size2 groups=&size.;
      by date;
      var mkt_val;
      ranks size_quintile;
quit;

* Create size breaks which will be used later for missing values *;
proc means data=size2 noprint min nway missing;
      class date size_quintile;
      var mkt_val;
      output out=size_breaks (drop=_type_ _freq_) min=min_value;
run;

data size_breaks; set size_breaks;
      if size_quintile=. then delete;
run;

* Create a range for each quintile *;
proc sort data=size_breaks; by date size_quintile; quit;
proc expand data=size_breaks out=size_breaks;
      by date;
      id size_quintile;
      convert min_value = max_value /transformout= (lead 1) method=none;
quit;

* Replace the "." for max_value with a large number to make matching easier *;
* Similarly, set min mkt_value for smallest quintile to be 0 *;
data size_breaks; set size_breaks;
      if max_value=. then
            max_value=999999999999;
      if size_quintile=0 then
            min_value =0;
      /* Prevents firms at cutpoint from being double counted */
      max_value = max_value-0.0001;
run;

**** Move onto BTM portfolio ****;
* Calculate BTM for all firms *;
* Download CRSP/Compustat data *;
rsubmit;
proc sql;
      create      table compq as
      select      a.gvkey, a.datadate, a.rdq, (a.fyearq * 10) + a.fqtr as fyearqtr,
a.atq, a.atq - a.ltq as book, a.ceqq, a.prccq
      from  comp.fundq as a
      where ((indfmt  = 'INDL') and (consol = 'C') and (popsrc = 'D') and (datafmt
= 'STD'))
            /* This date range should work fine for BTM */
            and '30jan2011'd le datadate le '31dec2015'd
      order by gvkey, fyearqtr, atq descending;
quit;

* Eliminate duplicate quarters - very small number so just ignore them *;
proc sort data=compq nodupkey; by gvkey fyearqtr; run;

* Bring in permnos from CRSP *;
proc sql;
      create table compq2 as select
            a.*, b.lpermno as permno
      from compq as a left join crsp.ccmxpf_linktable as b
      on a.gvkey eq b.gvkey
```

```
      and not missing(b.lpermno)
      and b.linktype in ('LU', 'LC', 'LD', 'LF', 'LN', 'LO', 'LS', 'LX')
      and b.linkprim in ('C', 'P')
      and (b.linkdt <= a.rdq or b.linkdt = .B)
      and (a.rdq <= b.linkenddt or b.linkenddt = .E);
quit;

* Remove duplicate entries *;
proc sort data=compq2 out=compq2 nodupkey; by gvkey fyearqtr; run;
* Kill obs with missing permnos *;
data compq2; set compq2; where not missing(permno); run;

proc download data=compq2 out=compq2; run;
endrsubmit;

* Save to characteristic-adjusted folder *;
data char.bm_data; set compq2; run;

* Take the most recent datadate for a given RDQ *;
proc sort data=char.bm_data out=btm; by permno rdq descending datadate; quit;
* Remove duplicates *;
proc sort data=btm out=btm nodupkey; by permno rdq; quit;
* Create RDQ 'windows' for which the quarterly BVE is valid *;
proc sort data=btm out=btm nodupkey; by permno datadate; quit;

data btm; set btm;
      /* Make sure we have valid RDQs */
      where not missing(rdq);
      /*  Reset RDQ to be 3 months after fiscal quarter-end if difference is
greater
            than 6 months. This occurs for a small number of observations (around
            95 out of near 100k) */
      if intck('Month',datadate,rdq)>6 then rdq=intnx('Month',datadate,4)-1;
run;

proc expand data=btm method=none out=btm;
      by permno;
      id datadate;
      * Create lead RDQ variable *;
      convert rdq = rdq_lead /transformout= (lead 1) method=none;
quit;

data btm; set btm;
      rdq_lead = rdq_lead-1; /* So there arent two reports for a given date */
      if ceqq=. then ceqq=book;
      if ceqq=. then delete;
      if rdq_lead =. then rdq_lead = intnx('Month',rdq,3);
      if rdq_lead < rdq then delete;
run;

* Bring in calendar month-end dates from CRSP for merging *;
* Since are we forming portfolios as of each month-end *;
rsubmit;
proc sql;
      create table dates as select
            caldt
      from crsp.msp500 (where=(year(caldt)>=2010));
quit;

proc download data=dates out=dates;
quit;
```

```
endrsubmit;

* Join datasets *;
proc sql;
      create table returns as select distinct
            a.*, b.gvkey, b.permno, b.datadate,
            b.rdq, b.rdq_lead, b.ceqq
      from dates as a left join btm as b
      on a.caldt between b.rdq_lead and b.rdq;
quit;

* Check to make sure there are no duplicates *;
proc sort data=returns nodupkey; by permno caldt; quit;

* Pull in monthly returns and price data as of calendar month-end *;
rsubmit;
proc upload data=returns out=returns; run;
proc sql;
      create table returns as select
            a.*, b.ret, abs(b.prc) as prc, b.shrout
      from returns as a, crsp.msf as b
      where a.permno = b.permno
      and month(a.caldt) = month(b.date)
      and year(a.caldt) = year(b.date);
quit;

* Also bring in SIC codes for FF48 industry classifications *;
proc sql;
      create table returns as select distinct
            a.*, b.sic as siccd
      from returns as a left join comp.company as b
      on a.gvkey = b.gvkey;
quit;

proc download data=returns out=returns; run;
endrsubmit;

* Calculate market value and BTM *;
data returns2; set returns;
      mkt_val = prc*shrout;
      label mkt_val = 'Market Value';
      btm = ceqq*1000/(prc*shrout);
      label btm = 'Book-to-market';
run;

* Bring in existing size quintiles *;
proc sql;
      create table returns2 as select distinct
            a.*, b.size_quintile, b.min_value, b.max_value
      from returns2 as a left join size_breaks as b
      on a.caldt = b.date
      and a.mkt_val between b.min_value and b.max_value;
quit;

* Find the mean BTM average for each Fama French 48 industry for each month *;
* Fix a couple of SIC codes *;
data returns2; set returns2;
      if SICCD in (3990,3999) then SICCD = 3991;
run;

* Bring in FF48 codes from macro file *;
```

```
%ff48(data=returns2, out=returns3, sic=SICCD, newvarname=FF_IND);
proc sort data=returns3; by FF_IND caldt; run;

* Deal with extreme BTM values *;
%Winsorize_Truncate(dsetin=returns3, dsetout=returns3, vars=btm, type = W, pctl = 2
98);

* Determine BTM industry average for each calendar month-end *;
proc means data = returns3 noprint;
      by FF_IND caldt;
      var btm;
      output out = BTM_IND (drop=_Type_ _freq_)  mean=bmind;
run;

* Calculate long-term Industry BTM Average *;
data BTM_IND; set BTM_IND;
      by FF_IND caldt;
      retain avg n;
      if first.FF_IND then do;
            avg=bmind;
            n=1;
            bmavg=avg;
      end;
      else do;
      bmavg=((avg*n)+bmind)/(n+1);
      n+1;
      avg=bmavg;
      end;
      format bmavg comma8.2;
      drop avg n bmind;
run;

* Adjust firm-specific BTM for industry averages *;
proc sql;
      create table returns3 as select
            a.*, b.bmavg as BTM_AVG "LT Industry BTM Ratio" format comma8.2,
            (a.btm-b.bmavg) as BTM_ADJ "Adjusted BTM Ratio" format comma8.2
      from returns3 as a, BTM_IND as b
      where a.caldt=b.caldt and a.FF_IND=b.FF_IND;
quit;

* Determine BTM quintiles within each size quintile *;
proc sort data=returns3; by caldt size_quintile BTM_ADJ; quit;

* Create BTM quintiles *;
* Output to characteristic-adjusted folder *;
proc rank data=returns3 out=char.port_form1 groups=&btm_break.;
      by caldt size_quintile;
      var BTM_ADJ;
      ranks btm_quintile;
quit;

* Run this code to address observations that potentially fall off *;
* We create this now and pull it back in later *;
proc means data=char.port_form1 noprint min nway missing;
      class caldt size_quintile btm_quintile;
      var BTM_ADJ;
      output out=btm_breaks (drop=_type_ _freq_) min=min_value;
run;

data btm_breaks; set btm_breaks;
```

```
306         if btm_quintile=. then delete;
307   run;
308
309   * Create a range for each quintile *;
310   proc sort data=btm_breaks; by caldt size_quintile btm_quintile; quit;
311
312   proc expand data=btm_breaks out=btm_breaks;
313         by caldt size_quintile;
314         id btm_quintile;
315         convert min_value = max_value /transformout= (lead 1) method=none;
316   quit;
317
318   * Replace the "." for max_value with a large number to make matching easier *;
319   * Similarly, set min BTM_ADJ for smallest quintile to be -99999999999 *;
320   data btm_breaks; set btm_breaks;
321         if max_value=. then
322               max_value=999999999999;
323         if btm_quintile=0 then
324               min_value =-9999999999;
325         /* Prevents firms at cutpoint from being double counted */
326         max_value = max_value-0.0001;
327   run;
328   * Around 1650 obs, which is consistent with 5x5x[5x12] as we have about 5 years of
329   data here for now *;
330
331   * Clean up variables, re-order, and only keep the observations which are non-
332   missing *;
333   * We do this now and then again after we do momentum *;
334   proc sql;
335         create table char.port_form2 as select distinct
336               caldt, permno, size_quintile, btm_quintile, FF_IND, *
337         from char.port_form1 (where=(not missing(size_quintile+btm_quintile)));
338   quit;
339
340   * Check for duplicates *;
341   %dupl(dset=char.port_form2,keys=permno caldt);
342
343   data char.port_form3; set char.port_form2; run;
344   proc sort data=char.port_form3 out=temp; by size_quintile btm_quintile; run;
345
346   ********************************************************************************
347   *;
348   *II. Find what size/BTM portfolio each sample firm falls into and calculate
349          benchmark returns for the portfolio;
350   ********************************************************************************
351   *;
352
353   * Bring in the master dataset at this point *;
354   * This is our main in-sample dataset, but only bring in a few variables *;
355   data ret_data; set master.Step2_final(keep=gvkey fyearqtr rdq_to_use btm mkt_cap
356   permno); run;
357
358   * Bring in SIC codes again *;
359   rsubmit;
360   proc upload data=ret_data; run;
361   proc sql;
362         create table ret_data2 as select
363               a.*, b.sic as siccd
364         from ret_data as a left join comp.company as b
365         on a.gvkey = b.gvkey;
366   quit;
```

```
367
368  proc download data=ret_data2 out=ret_data2; run;
369  endrsubmit;
370
371  * Fix one block of SICs *;
372  data ret_data2; set ret_data2;
373        if SICCD in (3990,3999) then SICCD = 3991;
374  run;
375
376  * Create industry codes again *;
377  %ff48(data=ret_data2, out=ret_data2, sic=SICCD, newvarname=FF_IND);
378
379  * Find portfolios that will be used as a benchmark based on date, size, and BTM
380  portfolio *;
381  proc sql;
382        create table target_benchmark as select distinct
383              a.gvkey, a.fyearqtr, a.rdq_to_use, a.btm as btm_current, a.mkt_cap*1000
384  as mkt_cap_current, a.permno, a.FF_IND,
385              b.caldt, b.size_quintile, b.btm_quintile
386        from ret_data2 as a left join char.port_form3 as b
387        on a.permno = b.permno
388        /* Want to merge such that the portfolios are formed for the month preceding
389  a given EA */
390        and intck('Month',a.rdq_to_use,b.caldt)=-1;
391  quit;
392
393  * Next blocks of code are to minimize loss of observations *;
394  * Merge in caldt *;
395  proc sql;
396        create table target_benchmark as select distinct
397              a.*, b.trading_date format YYMMDDN8.
398        from target_benchmark as a left join char.trading_dates_final as b
399        on a.rdq_to_use = b.orig_date;
400  quit;
401
402  proc sql;
403        create table target_benchmark as select distinct
404              a.*, b.caldt as caldt2
405        from target_benchmark as a left join dates as b
406        on intck('month', b.caldt, a.trading_date) = 1;
407  quit;
408
409  * Merge in mkt_cap at caldt2 and BTM from most recent rdq before caldt2 *;
410  rsubmit;
411  proc upload data=target_benchmark; quit;
412  proc sql;
413        create table target_benchmark2 as select
414              a.*, abs(b.prc)*shrout as mkt_cap
415        from target_benchmark as a, crsp.msf as b
416        where a.permno = b.permno
417        and month(a.caldt2) = month(b.date)
418        and year(a.caldt2) = year(b.date);
419  quit;
420
421  proc sql;
422        create      table compq as
423        select      a.gvkey, a.datadate, a.rdq, (a.fyearq * 10) + a.fqtr as fyearqtr,
424  a.atq, a.atq - a.ltq as book, a.ceqq, a.prccq*a.cshoq*1000 as mkt_cap2
425        from  comp.fundq as a
426        where ((indfmt  = 'INDL') and (consol = 'C') and (popsrc = 'D') and (datafmt
427  = 'STD'))
```

```
            and '30jan2011'd le datadate le '31dec2015'd
      order by gvkey, fyearqtr, atq descending;
quit;

* Eliminate duplicate quarters - very small number so just ignore them *;
proc sort data=compq nodupkey; by gvkey fyearqtr; run;

proc sql;
      create table target_benchmark2 as select
            a.*, b.ceqq, b.book, b.mkt_cap2, b.rdq as rdq_prior
      from target_benchmark2 as a left join compq as b
      on a.gvkey = b.gvkey
      /* Six-month look back period for BTM */
      and intck('month',a.caldt2,b.rdq) between -6 and 0
      order by gvkey, fyearqtr, rdq descending;
quit;

proc sort data=target_benchmark2 nodupkey; by gvkey fyearqtr; run;

* Bring back down locally *;
proc download data=target_benchmark2; quit;
endrsubmit;

* Re-create BTM variable for missing observations *;
data target_benchmark3 (drop=btm_current ceqq book mkt_cap2 mkt_cap_current
rdq_prior);
      set target_benchmark2;
      if mkt_cap=. then mkt_cap=mkt_cap2;
      if mkt_cap=. then mkt_cap=mkt_cap_current;
      if ceqq=. then ceqq=book;
      btm=ceqq/mkt_cap*1000;
      if btm=. then btm=btm_current;
run;

* Merge in industry average BTM *;
proc sql;
      create table target_benchmark3 as select distinct
            a.*, b.bmavg as BTM_AVG "LT Industry BTM Ratio",
            a.btm-b.bmavg as BTM_ADJ "Adjusted BTM Ratio" format comma8.2
      from target_benchmark3 as a left join BTM_IND as b
      on a.caldt2 = b.caldt
      and a.FF_IND = b.FF_IND;
quit;

data target_benchmark5; set target_benchmark3; run;

* Now merge in breakpoints *;
* First do size_quintile *;
proc sql;
      create table target_benchmark6 as select
            a.*, b.size_quintile as size_quintile2, b.min_value, b.max_value
      from target_benchmark5 as a left join size_breaks as b
      on a.caldt2 = b.date
      and a.mkt_cap between b.min_value and b.max_value;
quit;

* Merge in btm_quintile based on caldt and size_quintile *;
proc sql;
      create table target_benchmark6 as select
            a.*, b.btm_quintile as btm_quintile2, b.min_value as min_value_btm,
b.max_value as max_value_btm
```

```
      from target_benchmark6 as a left join btm_breaks as b
      on a.caldt2 = b.caldt
      and a.size_quintile2 = b.size_quintile
      and a.BTM_ADJ between b.min_value and b.max_value;
quit;

* Keep relevant variables and assign quintiles in the event they were originally
missing *;
data target_benchmark6 (keep=rdq_to_use permno trading_date caldt size_quintile
btm_quintile);
      set target_benchmark6;
      caldt=caldt2;
      if size_quintile=. then size_quintile=size_quintile2;
      if btm_quintile=. then btm_quintile=btm_quintile2;
run;

* Save down target_benchmark6 *;
data char.target_benchmark1; set target_benchmark6; run;

* Only keep portfolio data that will be used as a target benchmark *;
proc sql;
      create table benchmark_portfolio as select distinct
            a.*,
            case
                  when b.caldt=.
                  then 0
            else 1
            end
      as valid_benchmark
      from char.port_form3 as a left join char.target_benchmark1 as b
      on (intck('month',a.caldt,b.caldt) between 0 and 2)
      and a.size_quintile = b.size_quintile
      and a.btm_quintile = b.btm_quintile;
quit;

data benchmark_portfolio; set benchmark_portfolio;
      if valid_benchmark;
run;


*************************************************************************
*;
*III. Calculate benchmark daily returns for the portfolio;
*************************************************************************
*;

proc sort data=benchmark_portfolio nodupkey; by caldt permno; run;

* Keep only necessary variables *;
data char.benchmark_portfolio;
      set benchmark_portfolio(drop=RET prc SHROUT ceqq min_value max_value);
run;
data benchmark_portfolio_temp; set char.benchmark_portfolio; run;

* Need to do all of these next steps in WRDS because datasets get huge *;
rsubmit;
proc upload data=benchmark_portfolio_temp out=benchmark_portfolio_temp; quit;
proc upload data=char.trading_dates_final out=trading_dates_final; quit;
* Pull in simple base returns first *;
proc sql;
      create table benchmark_portfolio_temp2 as select
```

```
            a.*, b.date, b.ret
      from benchmark_portfolio_temp as a left join crsp.dsf
(where=(year(date)>=2010)) as b
      on a.permno=b.permno
      and intck('month',a.caldt,b.date) between 1 and 5;
quit;

* Short-term Delisting Returns *;
proc sql;
      create table benchmark_portfolio_temp2 as select distinct
            a.*, b.dlret
      from benchmark_portfolio_temp2 a left join crsp.DSEDELIST (where=(dlret^=.
and DLSTDT^=.)) b
      on a.permno = b.permno
      and intck('month', a.caldt, b.DLSTDT) between 1 and 5;
quit;

* Reset returns for delisting *;
data benchmark_portfolio_temp2;
      set benchmark_portfolio_temp2;
      ret2 = ret;
      if ret2=. then
            ret2=dlret;
      drop dlret ret;
      rename ret2 = ret;
run;

proc sort data=benchmark_portfolio_temp2 nodupkey; by permno date descending caldt;
run;
proc sort data=benchmark_portfolio_temp2 nodupkey; by permno date; run;

* Trading date ranges *;
proc sql;
      create table benchmark_portfolio_temp2 as select distinct
            a.*, b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
b.trading_date_plus4,
            b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
b.trading_date_plus40,
            b.trading_date_plus60, b.trading_date_plus75
      from benchmark_portfolio_temp2 as a left join trading_dates_final as b
      on a.date = b.orig_date;
quit;

* Need to partition this dataset for computational purposes *;
* Before calculating buy-and-hold returns *;
data temp1; set benchmark_portfolio_temp2;
      where permno <= 32299;
run;
data temp2; set benchmark_portfolio_temp2;
      where (permno>32299 and permno<=81138);
run;
data temp3; set benchmark_portfolio_temp2;
      where (permno>81138 and permno<=89500);
run;
data temp4; set benchmark_portfolio_temp2;
      where permno>89500;
run;

proc download data=temp1; run;
proc download data=temp2; run;
proc download data=temp3; run;
```

```
611  proc download data=temp4; run;
612  endrsubmit;
613
614  * Need to get mean return per portfolio-date *;
615  * Do the first sample partition *;
616  rsubmit;
617  proc upload data=temp1; run;
618  * For now, start with [2,60] window *;
619  proc sql;
620        create table temp1 as select distinct
621              a.*, exp(sum(log(1+b.ret)))-1 as ret260
622        from temp1 as a left join temp1 as b
623        on a.permno = b.permno
624        and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
625        group by a.permno, a.date;
626  quit;
627  * Also do [2,40] *;
628  proc sql;
629        create table temp1 as select distinct
630              a.*, exp(sum(log(1+b.ret)))-1 as ret240
631        from temp1 as a left join temp1 as b
632        on a.permno = b.permno
633        and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
634        group by a.permno, a.date;
635  quit;
636  * And also [2,20] *;
637  proc sql;
638        create table temp1 as select distinct
639              a.*, exp(sum(log(1+b.ret)))-1 as ret220
640        from temp1 as a left join temp1 as b
641        on a.permno = b.permno
642        and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
643        group by a.permno, a.date;
644  quit;
645  * Also do [2,10] *;
646  proc sql;
647        create table temp1 as select distinct
648              a.*, exp(sum(log(1+b.ret)))-1 as ret210
649        from temp1 as a left join temp1 as b
650        on a.permno = b.permno
651        and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
652        group by a.permno, a.date;
653  quit;
654  * Also do [2,5] *;
655  proc sql;
656        create table temp1 as select distinct
657              a.*, exp(sum(log(1+b.ret)))-1 as ret205
658        from temp1 as a left join temp1 as b
659        on a.permno = b.permno
660        and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
661        group by a.permno, a.date;
662  quit;
663  * Also do [2,4] *;
664  proc sql;
665        create table temp1 as select distinct
666              a.*, exp(sum(log(1+b.ret)))-1 as ret204
667        from temp1 as a left join temp1 as b
668        on a.permno = b.permno
669        and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
670        group by a.permno, a.date;
671  quit;
```

```
* Also do [2,3] *;
proc sql;
      create table temp1 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret203
      from temp1 as a left join temp1 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
      group by a.permno, a.date;
quit;

proc download data=temp1 out=temp1; run;
endrsubmit;

* Save down first group *;
data char.port_ret_group1; set temp1; run;

* Do this for the other temporary groups *;
rsubmit;
proc upload data=temp2; run;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret260
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret240
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret220
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret210
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret205
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
```

```
733             a.*, exp(sum(log(1+b.ret)))-1 as ret204
734       from temp2 as a left join temp2 as b
735       on a.permno = b.permno
736       and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
737       group by a.permno, a.date;
738 quit;
739 proc sql;
740       create table temp2 as select distinct
741             a.*, exp(sum(log(1+b.ret)))-1 as ret203
742       from temp2 as a left join temp2 as b
743       on a.permno = b.permno
744       and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
745       group by a.permno, a.date;
746 quit;
747
748 proc download data=temp2 out=temp2; run;
749 endrsubmit;
750
751 * Save down second partition *;
752 data char.port_ret_group2; set temp2; run;
753
754 * Third group *;
755 rsubmit;
756 proc upload data=temp3; run;
757 proc sql;
758       create table temp3 as select distinct
759             a.*, exp(sum(log(1+b.ret)))-1 as ret260
760       from temp3 as a left join temp3 as b
761       on a.permno = b.permno
762       and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
763       group by a.permno, a.date;
764 quit;
765 proc sql;
766       create table temp3 as select distinct
767             a.*, exp(sum(log(1+b.ret)))-1 as ret240
768       from temp3 as a left join temp3 as b
769       on a.permno = b.permno
770       and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
771       group by a.permno, a.date;
772 quit;
773 proc sql;
774       create table temp3 as select distinct
775             a.*, exp(sum(log(1+b.ret)))-1 as ret220
776       from temp3 as a left join temp3 as b
777       on a.permno = b.permno
778       and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
779       group by a.permno, a.date;
780 quit;
781 proc sql;
782       create table temp3 as select distinct
783             a.*, exp(sum(log(1+b.ret)))-1 as ret210
784       from temp3 as a left join temp3 as b
785       on a.permno = b.permno
786       and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
787       group by a.permno, a.date;
788 quit;
789 proc sql;
790       create table temp3 as select distinct
791             a.*, exp(sum(log(1+b.ret)))-1 as ret205
792       from temp3 as a left join temp3 as b
793       on a.permno = b.permno
```

```
794         and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
795         group by a.permno, a.date;
796   quit;
797   proc sql;
798         create table temp3 as select distinct
799               a.*, exp(sum(log(1+b.ret)))-1 as ret204
800         from temp3 as a left join temp3 as b
801         on a.permno = b.permno
802         and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
803         group by a.permno, a.date;
804   quit;
805   proc sql;
806         create table temp3 as select distinct
807               a.*, exp(sum(log(1+b.ret)))-1 as ret203
808         from temp3 as a left join temp3 as b
809         on a.permno = b.permno
810         and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
811         group by a.permno, a.date;
812   quit;
813
814   proc download data=temp3 out=temp3; run;
815   endrsubmit;
816
817   * Save down third partition *;
818   data char.port_ret_group3; set temp3; run;
819
820   * Final group *;
821   rsubmit;
822   proc upload data=temp4; run;
823   proc sql;
824         create table temp4 as select distinct
825               a.*, exp(sum(log(1+b.ret)))-1 as ret260
826         from temp4 as a left join temp4 as b
827         on a.permno = b.permno
828         and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
829         group by a.permno, a.date;
830   quit;
831   proc sql;
832         create table temp4 as select distinct
833               a.*, exp(sum(log(1+b.ret)))-1 as ret240
834         from temp4 as a left join temp4 as b
835         on a.permno = b.permno
836         and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
837         group by a.permno, a.date;
838   quit;
839   proc sql;
840         create table temp4 as select distinct
841               a.*, exp(sum(log(1+b.ret)))-1 as ret220
842         from temp4 as a left join temp4 as b
843         on a.permno = b.permno
844         and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
845         group by a.permno, a.date;
846   quit;
847   proc sql;
848         create table temp4 as select distinct
849               a.*, exp(sum(log(1+b.ret)))-1 as ret210
850         from temp4 as a left join temp4 as b
851         on a.permno = b.permno
852         and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
853         group by a.permno, a.date;
854   quit;
```

```
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret205
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret204
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret203
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
      group by a.permno, a.date;
quit;

proc download data=temp4 out=temp4; run;
endrsubmit;

* Save down fourth and final partition *;
data char.port_ret_group4; set temp4; run;

data temp1; set char.port_ret_group1; run;
data temp2; set char.port_ret_group2; run;
data temp3; set char.port_ret_group3; run;
data temp4; set char.port_ret_group4; run;

* Combine the partitioned datasets *;
data benchmark_portfolio_temp3;
      set temp1 temp2 temp3 temp4;
run;

data benchmark_portfolio_temp3;
      set benchmark_portfolio_temp3;
      if caldt=. or size_quintile=. or btm_quintile=. or date=. then delete;
run;

* Calculate equal-weighted returns by portfolio *;
proc means data=benchmark_portfolio_temp3 mean noprint;
      class caldt date size_quintile btm_quintile;
      var ret ret260 ret240 ret220 ret210 ret205 ret204 ret203;
      output out=portfolio_returns (drop=_type_ _freq_) mean=ew_ret ew_ret_260
ew_ret_240 ew_ret_220 ew_ret_210 ew_ret_205

      ew_ret_204 ew_ret_203;
run;

data portfolio_returns; set portfolio_returns;
      if caldt=. or size_quintile=. or btm_quintile=. or date=. then delete;
run;

* Save down *;
```

```
data char.port_ret_combined; set portfolio_returns; run;
data char.benchmark_portfolio; set benchmark_portfolio_temp3; run;

*********************************************************************************
*;
* IV. Calculate firm returns
*********************************************************************************
*;

data benchmark_portfolio_temp3; set char.benchmark_portfolio; run;
data target_benchmark6; set char.target_benchmark1; run;

* Let's get the firm raw returns *;
proc sort data=target_benchmark6; by permno rdq_to_use; run;
proc sql;
      create table target_benchmark7 as select
            a.*, b.ret260, b.ret240, b.ret220, b.ret210, b.ret205, b.ret204,
b.ret203,
            b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
b.trading_date_plus4,
            b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
b.trading_date_plus40,
            b.trading_date_plus60
      from target_benchmark6 as a left join benchmark_portfolio_temp3 as b
      on a.permno = b.permno
      and a.trading_date = b.date;
quit;

proc sql;
      create table target_benchmark7 as select
            a.*, b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
b.trading_date_plus4,
            b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
b.trading_date_plus40,
            b.trading_date_plus60
      from target_benchmark7 as a left join char.trading_dates_final as b
      on a.rdq_to_use = b.orig_date;
quit;

* Daily returns are missing sometimes, so double check the DSF and pull the returns
in as applicable *;
rsubmit;
proc upload data=target_benchmark7; run;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret260_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret240_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
```

```
977               a.*, exp(sum(log(1+b.ret)))-1 as ret220_alt
978         from target_benchmark7 as a left join crsp.dsf as b
979         on a.permno = b.permno
980         and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
981         group by a.permno, a.trading_date;
982   quit;
983   proc sql;
984         create table target_benchmark7 as select distinct
985               a.*, exp(sum(log(1+b.ret)))-1 as ret210_alt
986         from target_benchmark7 as a left join crsp.dsf as b
987         on a.permno = b.permno
988         and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
989         group by a.permno, a.trading_date;
990   quit;
991   proc sql;
992         create table target_benchmark7 as select distinct
993               a.*, exp(sum(log(1+b.ret)))-1 as ret205_alt
994         from target_benchmark7 as a left join crsp.dsf as b
995         on a.permno = b.permno
996         and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
997         group by a.permno, a.trading_date;
998   quit;
999   proc sql;
1000        create table target_benchmark7 as select distinct
1001              a.*, exp(sum(log(1+b.ret)))-1 as ret204_alt
1002        from target_benchmark7 as a left join crsp.dsf as b
1003        on a.permno = b.permno
1004        and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
1005        group by a.permno, a.trading_date;
1006  quit;
1007  proc sql;
1008        create table target_benchmark7 as select distinct
1009              a.*, exp(sum(log(1+b.ret)))-1 as ret203_alt
1010        from target_benchmark7 as a left join crsp.dsf as b
1011        on a.permno = b.permno
1012        and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
1013        group by a.permno, a.trading_date;
1014  quit;
1015
1016  * Check for de-listing returns *;
1017  proc sql;
1018        create table target_benchmark7 as select
1019              a.*, b.dlret
1020        from target_benchmark7 as a left join crsp.DSEDELIST (where=(dlret^=. and
1021  DLSTDT^=.)) as b
1022        on a.permno=b.permno
1023        and b.dlstdt between a.rdq_to_use and a.trading_date_plus2;
1024  quit;
1025
1026  proc download data=target_benchmark7 out=target_benchmark7; run;
1027  endrsubmit;
1028
1029  * If original returns were missing or delisted then update as applicable *;
1030  data target_benchmark7 (drop=ret260_alt ret240_alt ret220_alt ret210_alt ret205_alt
1031  ret204_alt
1032                                      ret203_alt dlret); set target_benchmark7;
1033        if ret260=. then ret260=ret260_alt;
1034        if ret260=. then ret260=dlret;
1035        if ret240=. then ret240=ret240_alt;
1036        if ret240=. then ret240=dlret;
1037        if ret220=. then ret220=ret220_alt;
```

```
      if ret220=. then ret220=dlret;
      if ret210=. then ret210=ret210_alt;
      if ret210=. then ret210=dlret;
      if ret205=. then ret205=ret205_alt;
      if ret205=. then ret205=dlret;
      if ret204=. then ret204=ret204_alt;
      if ret204=. then ret204=dlret;
      if ret203=. then ret203=ret203_alt;
      if ret203=. then ret203=dlret;
run;

* Merge in portfolio returns *;
proc sql;
      create table target_returns as select distinct
            a.*, b.date, b.ew_ret_260, b.ew_ret_240, b.ew_ret_220, b.ew_ret_210,
b.ew_ret_205, b.ew_ret_204,
            b.ew_ret_203
      from target_benchmark7 as a left join char.port_ret_combined as b
      on a.size_quintile = b.size_quintile
      and a.btm_quintile = b.btm_quintile
      and a.caldt = b.caldt
      and intck('day',a.trading_date,b.date) = 0;
quit;

** Save CAR and portfolio returns *;
data char.returns_main(keep=permno rdq_to_use trading_date ret260 ret240 ret220
ret210 ret205 ret204 ret203
                                    ret105 ret103 ret102 ret360 ew_ret_260
ew_ret_240 ew_ret_220 ew_ret_210 ew_ret_205
                                    ew_ret_204 ew_ret_203);
      set target_returns;
run;


*********************************************************************;
** V. Calculate BHAR over specified range
*********************************************************************;

* Bring in main dataset again *;
data main; set master.step2_final; run;
* Fix dates for merging *;
data main; set main;
      rdq2 = input(put(rdq,8.),yymmdd8.);
      drop rdq;
run;
data main; set main;
      rdq = rdq2;
      drop rdq2;
      format rdq yymmddn8.;
run;

* Merge firm and portfolio returns into main dataset *;
proc sql;
      create table main2 as select
            a.*, b.ret260, b.ret240, b.ret220, b.ret210, b.ret205, b.ret204,
b.ret203,
            b.ew_ret_260, b.ew_ret_240, b.ew_ret_220, b.ew_ret_210, b.ew_ret_205,
b.ew_ret_204, b.ew_ret_203
      from main as a left join char.returns_main as b
      on a.permno = b.permno
      and a.rdq_to_use = b.rdq_to_use;
```

```
1099  quit;
1100
1101  * Calculate portfolio-adjusted returns *;
1102  * Multiply by 100 to express in terms of basis points *;
1103  data main2(drop=ew_ret_260 ew_ret_240 ew_ret_220 ew_ret_210 ew_ret_205 ew_ret_204
1104  ew_ret_203); set main2;
1105        ar_260_sb = 100*(ret260-ew_ret_260);
1106        ar_240_sb = 100*(ret240-ew_ret_240);
1107        ar_220_sb = 100*(ret220-ew_ret_220);
1108        ar_210_sb = 100*(ret210-ew_ret_210);
1109        ar_205_sb = 100*(ret205-ew_ret_205);
1110        ar_204_sb = 100*(ret204-ew_ret_204);
1111        ar_203_sb = 100*(ret203-ew_ret_203);
1112  run;
1113
1114  * Save down *;
1115  * Use this in Stata for the LT returns tests *;
1116  data master.sizebtm; set main2(keep=gvkey fyearqtr ar_260_sb ar_240_sb ar_220_sb
1117  ar_210_sb
1118                                                  ar_205_sb ar_204_sb ar_203_sb);
1119  run;
1120
1121
1122  ******************;
1123  *** END OF FILE ***;
1124  ******************;
1125
```