# CrazyEights Main

```
setUp   exports: deck, dealer, computerscore, userscore,
                       maxptspergame

do {

     shuffleCards  imports: deck
                   exports: deck

     dealCards     imports: deck, dealer
                   exports: deck, computerhand, userhand, discard,
                            namedsuit, currentplayer

     playOneSet    imports: deck, computerhand, userhand, discard,
                            namedsuit, currentplayer
                   exports: computerhand, userhand

    scoreOneSet   imports: computerhand, userhand, computerscore,
                           userscore, dealer
                  exports:  computerscore, userscore, dealer

}while  ( !  gameOver )  imports: computerscore, userscore,
                                  maxptspergame

 shutDown      imports: computerscore, userscore, maxptspergame
```

For each “bogus” function called, write detailed specifications.
WHY????
(Examples are provided here for three functions:
setUp,  dealCards, and playOneSet.)

```
/*****************************************************************
setUp   exports: deck, dealer, computerscore, userscore,
                              maxptspergame

Initializes a new deck of cards, establishes the dealer by randomly
choosing between the computer or the user, initializes both player
scores to 0, and prompts the user to establish the maximum points
for the overall game.

*/

void setup ()
{
}
```

```
/********************************************************************************
DealCards      imports: deck, dealer
               exports: deck, computerhand, userhand, discard,
                          namedsuit, currentplayer


Deals 7 cards to each hand and one for the discard card.  If the discard
happens to be an 8,  then the first player (not the dealer) establishes the
namedsuit.  The currentplayer is set to be the player that is *not* the dealer.

*/

void dealCards()
{
}


/**********************************************************************
playOneSet    imports: deck, computerhand, userhand, discard,
                                     namedsuit, currentplayer
              exports: computerhand, userhand

Players take turns until a player runs out of cards, or until the deck empties
AND neither player can discard a card.  Both hands are returned once the set
is completed.
*/

void playOneSet()
{
}
```

Once all specs have been written for the “bogus” functions that were called, pick one of those functions to work on next. REPEAT the same process. If you arrive at logic that is too complicated, call (and specifiy) a “bogus” function to simplify the code. For example, try not to have nested control structures to keep the code as easy to read as possible.

Practice the principle of least astonishment!!!

Notice:

Control structures are syntactically complete, and variables in conditional expressions are declared and initialized. It is ONLY the bogus functions that are not implemented.

```
/*******************************************************************
Setup   exports: deck, dealer, computerscore, userscore,
                      maxptspergame

Initializes a new deck of cards, establishes the dealer by randomly
choosing between the computer or the user, initializes both player scores
to 0, and prompts the user to establish the maximum points for the
overall game.

*/

void setup ()
{

     BuildDeck      exports: deck


     PickDealer     exports: dealer


     ZeroScores     exports: computerscore, userscore


     SetMaxPoints   exports: maxptspergame

}
```

```
/********************************************************************
PlayOneSet    imports: deck, computerhand, userhand, discard,
                                       namedsuit, currentplayer
              exports: computerhand, userhand

Players take turns until a player runs out of cards, or until the deck empties
AND neither player can discard a card.  Both hands are returned once the set
is completed.
*/

void PlayOneSet()
{       int computerpass=0, userpass=0;
        do {
                switch (currentplayer) {

                        case user:      UserTurn
                                                imports: userhand, deck, discard,
                                                                namedsuit, userpass
                                                exports: userhand, deck, discard,
                                                namedsuit, userpass

                                        break;

                        case computer: ComputerTurn
                                                imports: computerhand, deck, discard,
                                                                namedsuit, computerpass
                                                exports: computerhand, deck, discard,
                                                                namedsuit, computerpass
                }

                GetNextPlayer           imports:currentplayer
                                        exports:currentplayer

        }while ( ! Setover ) imports:   computerhand, userhand,
                                        computerpass, userpass
                             exports: (returns) boolean (true/false)

}
```