Lab 2

Common problems –

- missing categories of test cases
- missing test cases within a category
  - most found the bugs in variant 1, but may not have caught other problems
- test case names not fully descriptive of what is being tested
- test cases for invalid input
  - not needed!
- incorrect test cases
  - make sure the expected result is actually the right answer!
  - note that the `get*Winner` routines for RPS return -1 for a tie
- technical aspects of tester subroutines (`RPSTester`)
  - print enough to be able to determine if a test passed or failed
  - `get*Winner` routines return a value so the tester subroutine can actually check and print passed/failed instead of requiring a manual inspection


```
public static void sort3(int a,
                         int b,
                         int c)
```

Print the three values a, b, c in increasing order, separated by spaces. For example, if a, b, and c have the values 10, 30, 20, respectively, the output will be the line

```
10 20 30
```

Parameters:

a - first value to sort

b - second value to sort

c - third value to sort

- sorting is about ordering
  - need to make sure that each possible ordering of smallest, middle, largest values is correctly sorted
- what about equal values?
  - check orderings involving duplicate values, including smallest-is-duplicate, largest-is-duplicate, and all the same
- what about 0 or negative values?
  - no reason to expect something other than <, <=, >, >=, == are used to compare values and those don't require special handling for 0 or negative values
- what about large numbers or small numbers? what about numbers that are close together or far apart?
  - ditto and ditto

Lab 2

Constructor Details

Time

```
public Time(String zulu)
```

Create a time object using Zulu time.

Parameters:

`zulu` - time in the form HHmmZ e.g. 0815Z

Method Details

convert

```
public abstract String convert(String timezone)
```

Return a formatted string for the time in the specified timezone.

Parameters:

`timezone` - one of EDT, EST, CDT, CST, MDT, MST, PDT, PST

Returns:

a string in the form 4:15 AM (EDT)

- check all behaviors – there are 8 legal timezone values
  - all should be checked
  - it is unknown whether there is separate handling for the timezone (E,C,M,P) and daylight/standard time (D,S) so best to check all the combos
- what is tricky about the 12 hour clock that might result in handling errors?
  - there are AM and PM times – check both
  - going past 12 – check conversions that cross midnight and noon
  - handling 12, compared to 00 or 24 in Zulu time – check conversions that end at midnight (and noon)

getRoundWinner

```
public static int getRoundWinner(char player1,
                                 char player2)
```

Get the winner for a single round. Rock breaks scissors, scissors cuts paper, paper covers rock.

Parameters:

`player1` - player 1's play: R, P, or S

`player2` - player 2's play: R, P, or S

Returns:

the winning player (1 or 2), or -1 if there is a tie

- check all behaviors
  - there are three legal values for `player1` and `player2` – check all combinations
  - there are three possible outcomes – checking all combinations for `player1` and `player2` covers these

getGameWinner

```
public static int getGameWinner(String player1,
                                String player2,
                                int numwins)
```

Get the winner over multiple rounds. The winner is the first player to reach the specified number of wins. If either player runs out of plays before there's a winner, the game is deemed a tie.

Parameters:

`player1` - player 1's plays, specified by a series of R, P, S characters

`player2` - player 2's plays, specified by a series of R, P, S characters

`numwins` - number of round wins needed to win the game

Returns:

the winning player (1 or 2), or -1 if the required number of wins isn't reached

- check all behaviors
  - there is not a finite set of inputs, so it is not possible to check all valid inputs
  - check all outcomes – player 1 wins, player 2 wins, tie because the required number of wins isn't reached
    - "if either player runs out of plays before there's a winner" – so there are actually three tie cases (player 1 runs out of plays first, player 2 runs out of plays first, and they both run out of plays at the same time)
- what other things could go awry?
  - the winner is the first player to reach the specified number of wins – which means additional plays past that point should be ignored
    - check case(s) where the winner if all of the plays are considered is not the first player to reach numwins

Lab 2

- things to revise in a resubmit
  - missing some more subtle categories of tests
  - missing some test cases within a category
  - other things noted in individual feedback

  → not noted in individual feedback – see the discussion on the preceding slides to identify what is missing
- reasons to resubmit even if you got 10
  - missing some more subtle categories of tests
  - missing some test cases within a category
  - other things noted in individual feedback
  - extra point for substantive resubmit