### SER421: Web Applications and Mobile Systems Fall 2015

### Lab 1, due Monday, 9/14/15 at 11:59pm via online submission

The goal Lab 1 is to get you “warmed up” for the coming semester by implementing Javascript code that works with XML and JSON data. **For this lab only, you will work by yourself.**

For this lab you will work with XML documents representing golf tournament results, adapted from a news reporting standard known as SportsML. Each document represents a specific round of a specific golf tournament. This information is contained in a top-level element tournament-metadata. Example:

<tournament-metadata tournament-name="British Open" start-date-time="20010716" end-date-time="20010722" site-name="Royal Lytham &amp; St. Annes Golf Club" site-city="Lytham St. Annes" site-country="England">

<award value="840000" currency="USD"/>

<course yardage="6905" par="71"/>

<round number="1" status="completed"/>

</tournament-metadata>

Following the tournament meta-data is a sequence of per player data. There is no DTD or XSD for these files, but it is straightforward to infer the structure by inspection.

**Activity 1: Parse XML in Javascript and return results in JSON**

The requirements of your first task are:

1. Use the xmlDoc built-in DOM parser of a web browser to create a DOM from the XML document
2. Write the code to output to the screen a JSON string that represents the following:
   1. The tournament name, start and end dates, golf course site, award value, course yardage, par, the current round, and a count of the players in the tournament
   2. The last name and first initial of the player
   3. The player’s current score to par
   4. The current hole the player is playing, or “finished”

Example: Consider the sample file golf1.xml. You should return **(NOTE the slight change of players to an array!)**

{ "tournament": {

"name": "British Open",

"start date": " ",

"end date": " ",

"award": 840000,

"yardage": 6905,

"par": 71,

"round": 1,

"players": [

{"lastname": "Montgomerie", "firstinitial": "C", "score": -3, "hole": 17 },

{"lastname": "Fulke", "firstinitial": "P", "score": -5, "hole": "finished"}

]

}

}

Implement a function *validateData* that validates the semantics of the data by implementing the following checks:

1. A round cannot be completed if all the players are not on hole “finished”
2. A tournament may have at most 4 rounds. The completion of the 4th round marks the end of the tournament
3. A player cannot have completed more than the 17th hole (if they completed 18 then they are “finished”, except in the last round in case of tiebreakers.

*validateData* should work from the DOM and return true or false (Boolean).

Note that the indentation above is merely for stylistic purposes. For reasons that will be clear when you read Activity 2, please implement this by creating objects *Tournament* and *Player* with the corresponding properties.

**Activity 2: Fun with functions and objects**

For this activity you will implement a collection of Javascript functions and manipulate Javascript objects.

1. Implement a function that takes the JSON string above and returns the JSON string back except with the player entries sorted first by score (lowest score first), then by hole (later holes first). This function should be named *leaderboard*.
2. Implement a function named *projectScoreByIndividual* that accepts the above JSON string for a player (e.g. {"lastname": "Montgomerie", "firstinitial": "C", "score": -3, "hole": 17}) and returns a projected score for the player when the round is completed. Project the player’s final score based on his current score and hole (individual rate of progress). For example, if the player has a score of -2 after 12 holes, then project that he has a final score of -3 at the end of the round. For the example above, the projected score would be -3.176 🡪 (-3/17\*(18-17) + -3) = -3.176
3. Implement a function named *projectScoreByHole* that accepts the above JSON string for a player (e.g. {"lastname": "Montgomerie", "firstinitial": "C", "score": -3, "hole": 17}) and returns a projected score for the player at the end of the round by projecting the player’s final score based on his current score and hole plus the collective rate of progress (the average score per hole for all golfers). For example, if the player’s score is -2 on he has finished the 9th hole, and on average golfers are scoring +0.11 per hole then add -2 plus 9 \* +0.11 to get a projected score of -1. For the example above, the returned score would be -3 + (1 \* 0.11) = -2.89.
4. Implement a function named *projectedLeaderboard* that does exactly what *leaderboard* does except it takes another argument representing a function (such as *projectScoreByXXX*) and uses this function to determine a leaderboard based on each player’s projected finishing score. If you are really clever you can reuse the leaderboard function.

Using the objects *Tournament* and *Player* suggested above, do the following:

1. Add to the *Tournament* object a property named *players* that is an array of the *Player* objects in the tournament.
2. Modify the array property you just created so it’s *sort* operation sorts the players according to the same logic as the aforementioned *leaderboard* function.
3. Modify the prototype of the *Tournament* object by adding a function *printLeaderboard* that prints the leaderboard to the console using console.log().
4. If the tournament is completed, create a new property on Tournament named *winner* that is assigned the player that is the winner of the tournament. Provide a *getWinner* function on the object that returns the winner’s last name.
5. Also, when the tournament is over, assign to every player a property named *winnings*. Set the value of winnings to 50% of the tournament award for the winner, 30% for 2nd place, and 20% for 3rd place.