CS 105: Sample Midterm Exam #1

Midterm Exam from Spring 2015
Solutions Included

.

Registered Student Organizations (RSOs) are student-ran organization at The University of Illinois. Fraternities, sororities, interest groups, and student chapters of professional organizations are all part of over the 700+ RSOs that exist on campus. The next thirteen questions will explore creating an application to vote for the president of a RSO.

1. Within our voting application, we represent the candidates for president in the variable **`candidates`**, defined below:

```
var candidates = ["Alice", "Brad", "Cai", "Deri"];
```

What type of data is in the variable **`candidates`**?

A. Number
B. String
C. Boolean
D. Array
E. Object

2. Our voting app collected all of the votes from the members of the RSO and stored their votes in the variable **`votes`**. For one election, the value of the **`votes`** was the following:

```
var votes = ["Brad", "Deri", "Cai", "Cai", "Alice", "Alice",
             "Deri", "Brad", "Alice", "Alice", "Cai"];
```

How many members voted for president in the election?

A. console.log(candidates)
B. candidates.length
C. console.log(votes)
D. votes.length

3. In order to determine a winner, we need to create a variable that can tally the votes for each candidate. This variable, **`tally`**, will initially contain the following data before the votes are counted:

```
var tally = [ { name: "Alice", votes: 0 },
              { name: "Brad",  votes: 0 },
              { name: "Cai",   votes: 0 },
              { name: "Deri",  votes: 0 } ];
```

What type of data is in the variable **`tally`**?

A. Array of objects
B. Array of strings
C. Array of arrays
D. Object of arrays
E. Object of strings

In order to create the `tally` variable discussed on the previous page we must write some code to place the candidates from the `candidates` variable into the `tally` variable. As a reminder, the variable we are aiming to create has been reprinted below:

```
var tally = [ { name: "Alice", votes: 0 },
              { name: "Brad",  votes: 0 },
              { name: "Cai",   votes: 0 },
              { name: "Deri",  votes: 0 } ];
```

Consider the following code snippet to create this initial `tally` variable:

```
 1  // Create the empty tally variable:
 2  _____ Line 2 ______
 3
 4  // Loop through all of the candidates:
 5  _____ Line 5 ______ {
 6
 7     // Create an object with the name and votes variables:
 8     _____ Line 8 ______
 9
10     // Add the newly created object to the tally variable:
11     _____ Line 11 ______
12  }
```

Using the code above along with the `candidates` and `votes` variables defined on the previous page, answer the following four questions:

4. Which line of code should be placed in Line 2?

A. `var tally;`
B. `var tally = 0;`
[✓ C. `var tally = [];`]
D. `var tally = {};`

5. Which line of code should be placed in Line 5?

[✓ A. `for (var i = 0; i < candidates.length; i++)`]
B. `for (var i = 0; i < tally.length; i++)`
C. `for (var i = 0; i < votes.length; i++)`

6. Which line of code should be placed in Line 8?

A. `var obj = [name: candidates[i], votes: 0];`
B. `var obj = [name: candidates.i, votes: 0];`
[✓ C. `var obj = {name: candidates[i], votes: 0};`]
D. `var obj = {name: candidates.i, votes: 0};`

7. Which line of code should be placed in Line 11?

A. `obj += tally;`
B. `obj.push(tally);`
C. `tally += obj;`
[✓ D. `tally.push(obj);`]

Continuing with our application to vote for a president for an RSO, consider a function that takes in two parameters:

- **`vote`**: A single vote, as a String. For example: **`"Brad"`**.
- **`tally`**: The tally variable, as built on the previous page and re-printed below.

This function must return the index in **`tally`** of the entry that should receive the vote. Consider the following skeleton of such a function:

```
 1 /* Example format of tally:
 2  *     var tally = [ { name: "Alice", votes: 0 },
 3  *                   { name: "Brad",  votes: 0 },
 4  *                   { name: "Cai",   votes: 0 },
 5  *                   { name: "Deri",  votes: 0 } ];
 6  */
 7
 8 var findIndex = function( vote, tally ) {
 9   for (var i = 0; i < __Line 9__; i++) {
10     if (__Line 10__) {
11       __Line 11__
12     }
13   }
14   return -1;
15 }
```

As an example, a call to **`findIndex("Brad", tally)`** would return the index 1 since "Brad" is found at index 1.

8. Which line of code should be placed in the blank on Line 9?

A. `candidate.length`
[✓ B. `tally.length`]
C. `vote.length`
D. `votes.length`

9. Which line of code should be placed in the blank on Line 10?

A. `tally[i] == vote`
B. `tally[i].length == vote`
[✓ C. `tally[i].name == vote`]
D. `tally[i].votes == vote`

10. Which line lines of code should be placed in the blank on Line 11?

[✓ A. `return i;`]
B. `return tally[i];`
C. `return tally[i].name;`
D. `return tally[i].length;`
E. `return tally[i].name[i];`

11. Assuming the function is implemented correctly, what is the return value of the call **`findIndex("Zoey", tally)`**?

[✓ A. `-1`]
B. `0`
C. `4`
D. `undefined`

To finish up our voting application, you will be responsible for writing the last two pieces of code. When writing the code, you should assume that **`candidates`**, **`votes`**, and **`tally`** have all been defined for you. The formats of these variables have been described in the previous three pages and are re-printed for you here:

```
var candidates = ["Alice", "Brad", "Cai", "Deri"];
```

```
var votes = ["Brad", "Deri", "Cai", "Cai", "Alice", "Alice",
             "Deri", "Brad", "Alice", "Alice", "Cai"];
```

```
var tally = [ { name: "Alice", votes: 0 },
              { name: "Brad",  votes: 0 },
              { name: "Cai",   votes: 0 },
              { name: "Deri",  votes: 0 } ];
```

Additionally, you also have the **`findIndex`** function that we defined on the previous page. You should assume that this function is completed correctly and will return the index in **`tally`** of the candidate that was passed in as a parameter.

Finally, each function **must** be written generically so that any list of candidates and votes would be correctly calculated. You must **not** assume that there are always exactly four candidates that are named “Alice”, “Brad”, “Cai” and “Deri”.

FR1. *15 pts.* Using the **`candidate`**, **`tally`**, and **`votes`** variables, along with the **`findIndex`** function, write the JavaScript code that will update the **`tally`** variable with the votes for each candidate based on the recorded votes in the **`votes`** variable.

For example, based on the sample data we have provided you with, after you code runs the **`tally`** variable must look like the following:

```
[ { name: "Alice", votes: 4 },
  { name: "Brad",  votes: 2 },
  { name: "Cai",   votes: 3 },
  { name: "Deri",  votes: 2 } ]
```

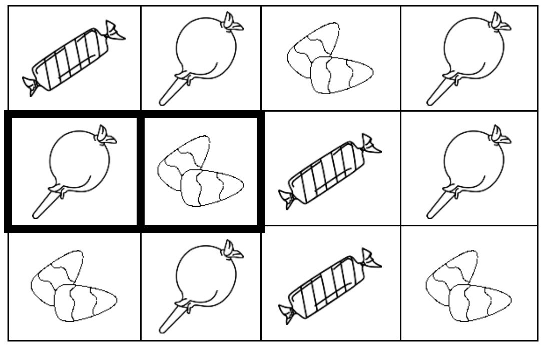
…answer this question on your free response answer sheet as question FR1.

FR2. *21 pts.* Write a function that takes in one parameter, the completed **`tally`** variable (the result of the previous question), and returns the name of the candidate (a String) who received the most votes. You can assume that there will always be exactly one winner *(eg: no two candidates will tie with the most number of votes)*.

Your answer must be a JavaScript function, not just a code snippet, though you may name it anything you would like. From our example data, your function would return “Alice” since she received the most votes (4 votes).

…answer this question on your free response answer sheet as question FR2.

Consider a popular mobile and Facebook game where you swap adjacent squares of candy pieces in order to match three-in-a-row.



As an example, swapping the two cells highlighted in bold in the image to the right (➔) results in three lollypops lining up in the second column.

For processing this grid in JavaScript, we will assign each image a specific letter. Specifically:

- Let be represented by the letter "A" *…possibly for Green **A**pple Jolly Rancher?*
- Let be represented by the letter "B" *…possibly for **B**ubblegum-Pop?*
- Let be represented by the letter "C" *…possibly for **C**andy Corn?*

Based on this translation, we can generate an array that represents the original state of the board. The original board shown above is represented in JavaScript as the following:

```
var board = [ ["A", "B", "C", "B"],
              ["B", "C", "A", "B"],
              ["C", "B", "A", "C"] ];
```

Each element of `board` is a single row within our grid and each element within that row is a letter that represents a candy. The two cells highlighted in bold in the original image, the lollypop ("B") and candy corn ("C") are at `board[1][0]` and `board[1][1]` in our array..

12. What type of variable is `board`?

A. An array of strings
[✓ B. An array of array of strings]
C. An array of objects
D. An array of array of objects
E. An array of objects with arrays

For the next three questions, consider the following answers:

A. 0
B. 1
C. 2
D. 3
E. 4

13. From the above answers, what is the value of `board.length`? **(D) 3**

14. From the above answers, what is the value of `board[0].length`? **(E) 4**

15. From the above answers, what is the value of `board[0][0].length`? **(B) 1**

Continuing from the previous page, we are still working with a board array with the following general layout (reprinted for reference):

```
var board = [ ["A", "B", "C", "B"],
              ["B", "C", "A", "B"],
              ["C", "B", "A", "C"] ];
```

Consider a function designed to check for a three-in-a-row match on a **`board`**:

```
 1  var foundMatch = function(board) {
 2     for (var i = 0; i < board.length - 2; i++) {
 3        for (var j = 0; j < board[i].length - 2; j++) {
 4           // Check vertically
 5           if ( __Line 5__ ) {
 6              return true;
 7           }
 8
 9           // Check horizontally
10           if ( __Line 10___ ) {
11              return true;
12           }
13        }
14     }
15
16     return false;
17  }
```

16. What Boolean expression can be placed inside of Line 5 to complete the conditional in order to check if there are three candies in-a-row vertically?

A. `board[i] == board[i + 1] && board[i + 1] == board[i + 2]`
B. `board[j] == board[j + 1] && board[j + 1] == board[j + 2]`
C. `board[i][j] == board[i][j + 1] && board[i][j + 1] == board[i][j + 2]`
[✓ D.] `board[i][j] == board[i + 1][j] && board[i + 1][j] == board[i + 2][j]`

17. Suppose we want to increase a **`score`** variable by 20 points when a three-in-a-row match is made. Using the function defined above, which snippet of code correctly increases the score when a match was made?

A. `if (foundMatch(board) = true) { score++; }`
B. `if (foundMatch(board) == true) { score++; }`
C. `if (foundMatch(board) = true) { score += 20; }`
[✓ D.] `if (foundMatch(board) == true) { score += 20; }`
E. `if (foundMatch(board) != true) { score += 20; }`

Suppose you design your own turn-based card game similar to many popular card games (Pokémon, Hearthstone, and others). In this game, each card has a name that identifies the card and two attributes:

- Attack Power, the amount of damage the card does when used
- Health, the initial amount of health of the card

As an example, consider several cards:

Bloodfen Raptor	
3 Attack	**2** Health

Chillwind Yeti	
4 Attack	**5** Health

Boulderfist Ogre	
6 Attack	**7** Health

Each card can be represented as a JavaScript object. For example, the JavaScript objects for the cards shown above are each defined in their own variable:

```
var yeti   = { name: "Chillwind Yeti",   attack: 4, health: 5 };
var raptor = { name: "Bloodfen Raptor",  attack: 3, health: 2 };
var ogre   = { name: "Boulderfist Ogre", attack: 6, health: 7 };
```

18. Suppose you wrote the following line of code:

```
var deck = [ yeti, raptor, ogre ];
```

What type of variable is **`deck`**?

A. An array of strings
B. An array of arrays
[✓ C. An array of objects]
D. A string of arrays
E. An object of arrays

19. Consider the following function:

```
var f = function(deck) {
   var card = deck[0];
   for (var i = 0; i < deck.length; i++) {
      if (deck[i].health > card.health) {
         card = deck[i];
      }
   }
   return card.health;
}
```

If the **`deck`** variable from the previous question was passed into this function as **`deck`**, what is the return value of this function?

A. 0
B. 1
C. 3
D. 5
[✓ E. 7]

Continuing with the card game from the previous page, suppose our game has only one interaction: two cards may attack each other. When this occurs, we define one card as the **attacker** and the other as the **defender**. The rules are:

- The attacker subtracts from its health the attack value of the defender **AND**
- The defender subtracts from its health the attack value of the attacker

Simply stated, both the attacker and defender lose health equal to the attack value of the other card involved. As an example, if a Boulderfest Ogre (6 attack, 7 health) attacks a Chillwind Yeti (4 attack, 5 health):

- The Ogre loses 4 health (attack value of the Yeti), bringing the health of the Ogre down from 7 health to 3 remaining health.
- The Yeti loses 6 health (attack value of the Ogre), bringing the health of the Yeti down from 5 health to -1.

When the health of a card is at or below zero, we consider that card to be destroyed. In the previous example, the Yeti is destroyed since its health ended at -1.

20. If the cards involved in the attack are stored in the variables `attacker` and `defender`, which conditional will always be true when **both** the attacker and the defender **are <u>not</u> destroyed**?

A. `if (attacker.health < defender.attack &&`
`    defender.health < attacker.attack)`

B. `if (attacker.health <= defender.attack &&`
`    defender.health <= attacker.attack)`

C. `if (attacker.health == defender.attack &&`
`    defender.health == attacker.attack)`

D. `if (attacker.health >= defender.attack &&`
`    defender.health >= attacker.attack)`

[✓ E. `if (attacker.health > defender.attack &&`
`    defender.health > attacker.attack)`]

21. Which line of JavaScript correctly updates the health of the `attacker` after the attack?

[✓ A. `attacker.health = attacker.health - defender.attack;`]

B. `attacker.health = defender.attack - attacker.health;`

C. `attacker.health = attacker.health - attacker.attack;`

D. `attacker.health = attacker.attack - attacker.health;`

22. After the health is updated (from the code in the previous question), which line of JavaScript checks if the `attacker` has been destroyed?

A. `if (attacker.health > 0)`

B. `if (attacker.health >= 0)`

C. `if (attacker.health == 0)`

[✓ D. `if (attacker.health <= 0)`]

E. `if (attacker.health < 0)`

23. Suppose we replace `attacker` with `defender` and `defender` with `attacker` in the previous two questions. Would making **only this change** correctly update the defender after the attack?

[✓ A. Yes]

B. No

One recent trend in social networking is location-aware or location-based social networking. In these apps, a post can only be made and/or seen if the user is within a certain distance from a set point. For example, in order to post as part of the University of Illinois, you must be within 10 miles of the University of Illinois within a location-aware app.

In order to check the distance, we have a `findDistance()` function has already been defined for you. This function takes in two JavaScript objects that define a location and returns the approximate distance between the two locations in miles:

```
 1  var findDistance = function( location1, location2 ) {
 2     /* Find the distance in the x-direction */
 3     var dx = location2.x - location1.x;
 4
 5     /* Find the distance in the y-direction */
 6     var dy = location2.y - location1.y;
 7
 8     /* Use the Pythagorean Theorem to find the distance */
 9     ___Line 9___
10
11     /* Each latitude/longitude degree is about 65 miles */
12     return (dist * 65);
13  }
```

24. In mathematics, the Pythagorean Theorem, $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} = \sqrt{x^2 + y^2}$, tells us the distance between two points in 2D-space. If `Math.sqrt()` is a JavaScript function to find the square root of an input parameter, what line of code should be placed in Line 10 to find the distance between the two locations passed into the function `findDistance()` abvoe?

A. `var dist = Math.sqrt( location1 + location2 );`
B. `var dist = Math.sqrt( location1.x + location2.y );`
C. `var dist = Math.sqrt( (location1.x)^2 + (location2.y)^2 );`
D. `var dist = Math.sqrt( dx + dy );`
[✓ E. `var dist = Math.sqrt( (dx * dx) + (dy * dy) );`]

25. Given the location of The University of Illinois is at 40.1105° N, 88.2284° W, which of the following lines of code correctly defines a JavaScript object that can be passed into `findDistance()` as one of its two input parameters?

A. `var location = 40.1105, -88.2284;`
B. `var location = [40.1105, -88.2284];`
C. `var location = [x: 40.1105, y: -88.2284];`
D. `var location = <x>40.1105</x><y>-88.2284</y>;`
[✓ E. `var location = { x: 40.1105, y: -88.2284 };`]

26. What type of variable is returned by the function `findDistance()`?

A. Boolean
[✓ B. Number]
C. String
D. Array
E. Object

Consider the following representation of a hand of standard playing cards:

```
var hand = [
  { suit: "Diamond", rank: "K" },
  { suit: "Club", rank: "10" }
];
```

27. What type of variable is `hand`?

A. A string
B. A number
C. A function
[✓ D. An array]
E. An object

28. Consider the following code snippet:

```
var s = "";
for (var i = 0; i < hand.length; i++) {
   s += hand[i].rank;
}
```

Using the `hand` variable form the previous question, what is the value of `s` after the code has completed?

A. `""`, (An empty string)
[✓ B. `"K10"`]
C. `"10K"`
D. `"DiamondClub"`
E. `"ClubDiamond"`

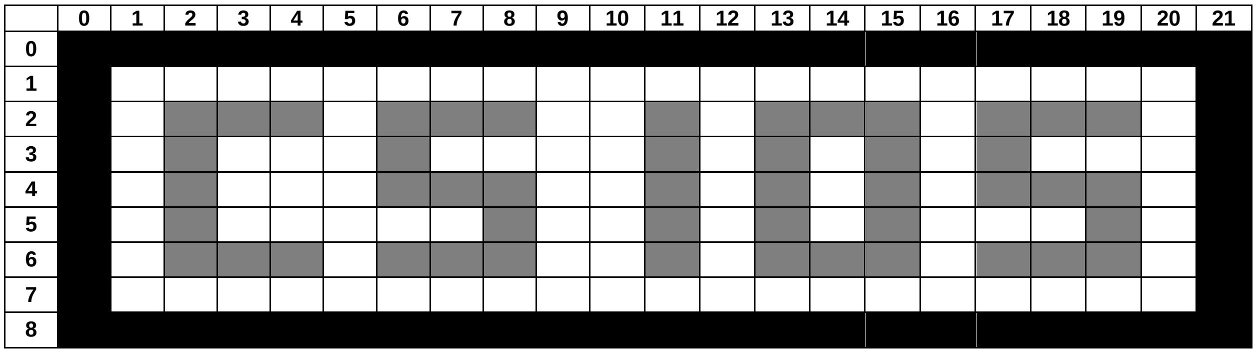
29. Consider the following code snippet that calculates the score of a hand based on the rank of a card. If the card's rank is a “K”, “Q”, or “J”, it is scored as 10; if the card's rank is “A”, it is scored as 1; otherwise, the card is scored the value of its rank. You should assume that the `parseInt` function will take a String and covert it to a Number (eg: “8” becomes 8).

```
var score = 0;
for (var i = 0; i < hand.length; i++) {
   var c = hand[i].rank[0];
   if (c == "J" || c == "Q" || c == "K") { score += 10; }
   else if (c == "A") { score += 1; }
   else { score += parseInt(c); }
}
```

Which of the following is true for a standard deck of playing cards?

A. The score is **always** calculated correctly for hands of **at least one card**
[✓ B. The score is **sometimes** calculates correctly for certain hands of **at least one card**]
C. The score is **never** calculated correctly for any hand with **at least one card**
D. The score is **sometimes** calculated correctly for hands with **zero cards**
E. The score is **never** calculated correctly for hands with **zero cards**

Thinking back to MP2 and Lab 3, you manipulated images similar to the image shown below:



This image is made up of three different colored pixels:

- A black pixel, `hsl(0, 0, 0.0)`, such as pixel `(0, 0)` which borders the image
- A grey pixel, `hsl(0, 0, 0.5)`, such as pixel `(2, 2)` which writes "CS 105" in the image
- A white pixel, `hsl(0, 0, 1.0)`, such as pixel `(4, 4)` which makes up all other pixels in the image

For each pixels, the color of the pixel is given in HSL values in the format `hsl(100, 0.2, 0.4)` where the first number (`100`) is the hue, the second number (`0.2`) is the saturation, and the third number is the luminance (`0.4`). As a reminder, the hue ranges from [0, 360], the saturation ranges from [0, 1], and the luminance ranges from [0, 1].

30. What is the hue of the pixel at (3, 4)?

[✓ A. 0.0]
B. 0.5
C. 1.0
D. 100
E. 360

31. What is the luminance of the pixel at (3, 4)?

A. 0.0
B. 0.5
[✓ C. 1.0]
D. 100
E. 360

FR3.
15 pts.
Write a function that takes in one parameter, a `SimpleImage`, that represents the image shown at the top of this page and returns the number of grey pixels.

As with MP2 and Lab 3, the data type `SimpleImage` has the method `getHSL(x, y)` that returns an object with three properties: `h`, `s`, and `l`, representing the hue, saturation and luminance. Additionally, the data type `SimpleImage` has the properties `height` and `width`, each returning the height and width of the `SimpleImage` as a Number.

…answer this question on your free response answer sheet as question FR3.

This page was intentionally left blank.