PIC 40A: Homework 2 (due 4/21 at 5pm)

Just for this homework on pure JavaScript, no submission to the PIC server is necessary! In this assignment you will make one file called `hw2.js` and all you need to do is submit this file to Gradescope before the deadline.

Important... delete all your test cases so that executing `hw2.js` does not cause anything to be printed to the console.

Reminder...

- NEVER use `var`; always use `let` or `const` to declare variables.
- Avoid `==` and `!=`. Use `===` and `!==` instead.
- Use semicolons correctly.

1. Define a function called `roll_dice` whose function comment reads as follows.

```
/**
Returns the result of rolling two dice.
Here are some possible return values.
. '1 + 2 = 3'
. '6 + 4 = 10'
. '3 + 5 = 8'
. '2 + 2 = 4'

The probability distribution of the return values
is the same as rolling two fair dice in real life.

@return {string} The result of rolling two dice.
*/
```

2. Define a function called `first_occurrences` whose function comment reads as follows.

```
/**
Deletes duplicates in an array of numbers.
Calling first_occurrences(arr) does not mutate the array referenced by arr.

@param {Array} arr : An array of numbers.
@return {Array} An array consisting of the same numbers in the same order as in arr.
                However, only the first occurrence of each number is included.
*/
```

   As an example of this function being called...

```
const arr = [1,1,1,2,1,1,2,2,3,3,3,2,1,4,1,2,3,5,1,2,3,4,5,6,6,6,5,4,3,2,1];
console.log(first_occurrences(arr));
```

   should print `[1, 2, 3, 4, 5, 6]`.

3. Define a function called `first_minus_second` whose function comment reads as follows.

```
/**
Removes elements of one array in accordance with another.
No new arrays are created during the function call.

Calling first_minus_second(arr1, arr2)
 . mutates the array referenced by arr1:
     all elements of arr1 that occur in arr2 are removed and
     the order of the remaining elements is preserved.
 . does not mutate the array referenced by arr2.

@param {Array} arr1 : The array to remove elements from.
@param {Array} arr2 : The array indicating which elements to remove.
*/
```

As an example of this function being called...

```
let arr1 = [8, 8, 1, 2, 7, 3, 4, 7, 5, 6, 11, 8, 13, 8, 12, 8, 11, 8, 18, 18];
const arr2 = [7, 11, 19, 19, 19, 13, 13, 12, 12];

first_minus_second(arr1, arr2);
console.log(arr1);
```

should print `[8, 8, 1, 2, 3, 4, 5, 6, 8, 8, 8, 8, 18, 18]`.

4. For this question I need to make a definition. I promise it is not as complicated as it looks!

   I have not made this definition as general as possible because otherwise you might hate me: https://stackoverflow.com/questions/1969232/.

   Definition. A **cookie** is a special type of `string`.

   It has the form `"name1=value1; name2=value2; ...; lastName=lastValue"`.

   The `name`s and `value`s can be any **non-empty** sequence of ASCII characters which are:

   (a) alphanumeric characters: `a`, `b`, `c`, ..., `z`, `A`, `B`, `C`, ..., `Z`, `0`, `1`, `2`, ..., `9`, or

   (b) a character appearing in the following string: `"!#%'*+-.^_`|~"`.

   (c) `=` is allowed in a `value`, but not a `name`.

   In particular, the following characters are **not allowed**:

   - white space
   - `,` (commas)
   - `;` (semicolons)

For example, the following are examples of cookies:

- `"first_name=Michael; last_name=Andrews; username=mjandr"`
- `"username=mjandr; first_name=Michael; last_name=Andrews"`

- `"_ga=GA1.2.34.56; dwf_sg_task_complete=False; lux_uid=888; _gid=GA1.2.88.88"`
- `"__stripe_mid=c4d6a-723-3ee-54d-640e5af; csrftoken=Kger31Gtcvyt%2F2ILWuQJoJ"`

The following is **not** a cookie `"name=Michael Andrews; position=PIC;assistant,adjunct"` because of the space between `Michael` and `Andrews`, as well as the semi-colon and comma in `PIC;assistant,adjunct`.

Finally, here is the actual question...

Define a function called `extract_username` whose function comment reads as follows.

```
/**
This function extracts from a given cookie
the 'value' corresponding to the 'name' "username".

For example, both of the following function calls return "mj=andr":
. extract_username("first_name=Michael; last_name=Andrews; username=mj=andr");
. extract_username("username=mj=andr; first_name=Michael; last_name=Andrews");

If the given cookie has no 'name' called "username",
then the function returns the empty string.

For example, we have
. extract_username("common_error=Micheal; " +
                   "another_one=Andrew; another=Andrew_Michaels") === ""

@param  {string} cookie : The cookie to extract information from.
@return {string} The 'value' corresponding to the 'name' "username";
                 the empty string if "username" is not a 'name'.
*/
```