Applications Team Skills Test

Cisco Security

Perl

1. What is the difference between the following two declarations?

```
my @thing = ( 1, 2, 3 );
my $thing = [ 1, 2, 3 ];
```

Answer:
In the above question, one is an array (@), while the other one is a reference to an array.

---

2. How would you reference the third (**bold**) element in the following structure?

```
my @things = ( 0, 1, 2, 3, 4, 5 );
```

Answer:

$things[2]

---

3. What kind of data structure is being created in the following code snippet? How would you retrieve the "`Apple`" element from `$structure`?

```
my $structure = { 1 => "Apple", 2 => "Orange", 3 => "Banana" };
```

Answer:

Data Type: Hashes Reference. And not normal Hash.

--> to access Apple : $structure->{1}

---

4. Why do these lines produce differing output?

```
print "My array = @array\n";
print "My array = " . @array . "\n";
```

Answer:

One is accessing the array contents while,
Second one is the accessing the Size of the array.

---

5. The lines in the previous question produce differing output, so why don't these lines?

```
print "My value = $var1\n";
print "My value = " . $var1 . "\n";
```

Answer:
The first statement is interpolation, but both of them give the same output because they are the same variables.

---

6. Given the following data structure, write a statement that sorts the entries by `id` and stores them in a new list:

```
my @users = (
    {id => 1,  name => "Frank"},
    {id => 10, name => "Joe"},
    {id => 5,  name => "Paul"}
);
```

Answer:

my @sorted= sort { $a->{id} <=> $b->{id} } @users;

---

7. There are errors in the `versionChecker` function. Find the errors and provide the correct fixes (with commentary):

```
# Compares two strings of the form '4.0', '3.0.1', etc. and returns:
#    -1 if the first version is less than the second
#     0 if the versions are equal
#     1 if the first version is greater than the second
#
# Note:  Version strings like '4.1.0' and '4.1' can be considered equivalent.
sub versionChecker {
    my ($v1, $v2) = @_;

    my @v1_parts = split (/\./, $v1);
    my @v2_parts = split (/\./, $v2);

    for (my $i = 0; $i < @v1_parts; $i++) {
        if ($v1_parts[$i] < $v2_parts[$i]) {
            return -1;
        } elsif ($v1_parts[$i] > $v2_parts[$i]) {
            return 1;
        }
    }

    # equal
    return 0;
}
```

Answer:
The program would not generate the correct result because it does not consider the length of the 2 versions. It only checks the length of v1 and not v2 if it is long. The program below does that check.

```
sub versionChecker {
        my ($v1, $v2) = @_;
        my @v1_parts = split (/\./, $v1);
        my @v2_parts = split (/\./, $v2);
```

```
        $l1 = @v1_parts;
        $l2 = @v2_parts;
        if($l1>l2){
                $length = $l2;
        } else {
                $length = $l1;
        }

        for (my $i = 0; $i < @length; $i++)
        {
                if ($v1_parts[$i] < $v2_parts[$i]){
                        return -1;
                } elsif ($v1_parts[$i] > $v2_parts[$i]) {
                        return 1;
                }
        }

        if($l1>$l2 && @v1_parts[$l2] > 0){
                return -1;
        } elsif($l1 < $l2 && @v2_parts[$l1] > 0 ) {
                return 1;
        }
        # equal
                return 0;
}


my $v1 = "4.0.0";
my $v2 = "4.0.0.1";
my $ans = versionChecker($v1,$v2);

print "$ans ";
```

8. Given a complex Perl data structure, define the function `ids` below such that it returns a list of all “id” values in the data structure at any level, sorted numerically. Make sure that if the same value is found in multiple locations in the data structure it is only included in the returned list once.
   Answer:

```
use Scalar::Util qw(looks_like_number);

sub ids {
   my ($data) = @_;
   if(ref $data eq 'ARRAY'){
      foreach my $i (@$data) {
        if (ref $i eq 'HASH') {
          ids($i);
        } elsif (ref $data[$i] eq 'ARRAY') {
          ids($i);
        } else {
            push @result,$data{id} ;
        }
      }
    } else {
      foreach my $key (keys %$data) {
        if (ref $data->{$key} eq 'HASH') {
                        ids($data->{$key});
                  }
        elsif (ref $data->{$key} eq 'ARRAY') {
                    ids($data->{$key});
            }   else {
                        push @result,$data->{id} ;
            }
      }
    }

   my %hash   = map { $_ => 1 } @result;
   my @unique = keys %hash;
   my @sorted = sort { $a <=> $b } @unique;

   return @sorted ; # Define this function
}

my $data = {
    'top' => {
        'window' => {
            'elements' => [
                { id => 44, name => 'link', value => 'www.cisco.com' },
                { id => 48, name => 'title', value => 'Cisco Home Page' },
                { id => 100, name => 'author', value => 'Admin' }
            ],
            id => 19
        },
        'cache' => [
            { id => 199, data => '9E27F645-2B37-46F3-B60C-36B1EFD235A2' },
            { id => 40, data => 'F35FD610-A0AE-4C71-871D-6D19685D01A6' },
            { id => 100, data => { name => 'author', value => 'Admin' } }
        ],
        id => 55
    },
    id => 1
};

# Global variable
my @result= ids($data);
my @final = ();
foreach $n (@result){
  if(looks_like_number($n)){
    push @final, $n;
  }
}

# should print “1, 19, 40, 44, 49, 55, 100, 199”
print join(', ', @final);
```

Javascript/HTML

All answers should be cross-browser compatible.

1. How would you retrieve a reference to the following object?

   ```
   <div id="mydiv">Lorem Ipsum</div>
   ```

   Answer:

   ```
   var x = document.getElementById("mydiv");
   ```

---

2. How would you programmatically retrieve the text inside the `div` in question J1 without using the `innerHTML` property?

   Answer:

   ```
   var x = document.getElementById("mydiv").textContent;
   ```

   `x will contain the data in the div with id 'mydiv'.`

---

3. How would you retrieve a reference to the following object (assume this is the only object named `"myinput"`)?

   ```
   <input name="myinput" type="text" />
   ```

   Answer:

   ```
   var x = document.getElementsByName("myinput");
   ```

---

4. Briefly describe how AJAX works and how a developer might use it to enhance the user experience on a given page.

   Answer:

   Asysnchronous javascript and XML .

Communicating with the server when the client send requests in the form of HTTP, and also working with the servers response, WITHOUT HAVING TO RELOAD the page and dynamically update it is the basic working of AJAX. Hence called Asynchronous.

There are a few basic Elements of AJAX ;

1. XMLHTTPRequest Object
2. JavaScript/DOM
3. CSS
4. XML

The working :

Whenever there is a change at the browser side(CLIENT), the request is sent in the form of XMLHTTPRequest to the Server to handle asynchronously . The JavaScript/DOM takes care of the interaction with this information. The Servers process the request and sends back its response in XMLHTTPRequest Format to the Browser to update the change without having to reload the page altogether..

Developer might use this for better interactivity, ease of navigation, single webpage to handle various applications instead of a bundle of several web pages.

---

5. Define the function `addOnLoad` below without creating any global variables:

6. Answer:

```
<!DOCTYPE html>
<html>
<script>

// fname will be executed when the <body> tag loads
function addOnLoad(fname){
   window.onload = fname;
}
    // init1 and init2 should be alerted when the body tag loads
    addOnLoad(function (){ window.alert('init2!'); });

</script>

<head>

</head>
<body>

</body>
</html>
```

---

7. Define the function `pos` below to return exactly where an element is on the page (X,Y coordinates):

```
// takes an element
// returns an array where the first value is the X position and the
// second value is the Y position relative to the page.
function pos(obj) {
    // Define this function
}
```

Answer:

```
function getOffset(el) {
   el = el.getBoundingClientRect();
   return {     left: el.left + window.scrollX,
                top: el.top + window.scrollY
          }
}
```

Reference: http://stackoverflow.com/questions/442404/retrieve-the-position-x-y-of-an-html-element

---

8. Explain how the `DOCTYPE` declaration can help or hurt your application development. Why would you choose one `DOCTYPE` over another?

Answer:

Doctype defines how the browser will render the HTML code that is written. Thus, to Standardise the interface operations , we use DOCTYPE declarations. If we done use DOCTYPE, the HTML page is rendered based on quirks mode which then might not look the way you thought it would.

Advantages: Cross-Browser Support because of Standardization

Disadvantage: Page may break, if your code isn't organized properly.

---

9. Describe how inheritance works in JavaScript.

Answer:

In JavaScript, due to the absence of Classes, inheritance is Prototype-Based. Thus an object inherits of another object. Thus a new method(In JavaScript its also called Property) added in a parent object is also available in the derived object. hasOwnProperty() function can be used to determine if the property is defined in that prototype or a parent or derived one.

---

10. How can the scope of a JavaScript function be modified when it is called?

Answer:

A function defined by a function expression inherits the current scope. That is, the function forms a closure. On the other hand, a function defined by a Function constructor does not inherit any scope other than the global scope. (which all functions inherit). The function scope can be changed by using "this" keyword .We can also use call() and apply() to this.

---

11. How would you create a "private" member of a Javascript object?

Answer:

```
function Constructor(...) {

      var that = this;

      var membername = value;

      function membername(...) {...}

}
```

---

1. When would you use the style `visibility:hidden` versus `display:none`?

Answer:

When we need to hide an element as if there is no element on the webpage, we use display:none. However if we just want to hide the element but still show that there is an element that exists and takes up space, we use visibility:hidden.

---

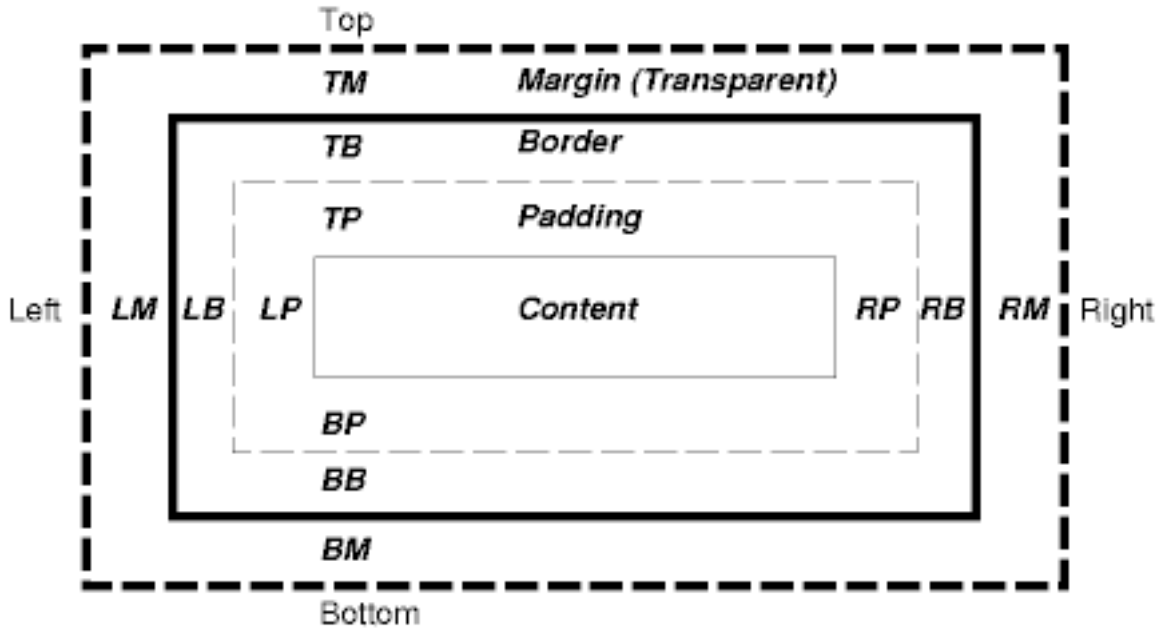
2. Describe the differences between padding, borders, and margins.

Answer:

Padding is the area around the context of the Element.

Border is the area between the Outline and the Padding of the Element.

Margin is the Area outside of the Element .



---

3. In the following example, explain why `#inner` is not fully contained within `#outer`.

```
<div id="container" style="height:200px;">
    <div id="outer" style="height:100%; border:1px solid black">
        <div id="inner" style="float:left; height:100%; border:1px solid blue">
            CONTENT
        </div>
    </div>
</div>
```

Answer:

Here the height of the inner is the same as that of the outer, and further more the border adds to its size. Thus it cannot be completely inside of #outer. However if we change the height to a lesser percentage, we can say that #inner is completely inside outer.

---

4. Define styles for the following elements such that `#UL` is in the upper left corner of the page, `#UR` is in the upper right, `#LL` is in the bottom left, and `#LR` is in the bottom right:

```
<style type="text/css">
    /* Define styles here */
</style>
<body>
    <div id="UL">Upper Left</div>
    <div id="UR">Upper Right</div>
    <div id="LL">Lower Left</div>
    <div id="LR">Lower Right</div>
</body>
```

Answer:

```
<!DOCTYPE html>
<html>
<head>  </head>
<style type="text/css">
    /* Define styles here */
#UL
{
 position: fixed;
top: 0;
left: 0;
}
#UR
{
position: fixed;
 top: 0;
 right: 0;
}
#LL
{
position: fixed;
 bottom: 0;
 left: 0;
}
#LR
{
position: fixed;
 bottom: 0;
 right: 0;
}
</style>
<body>
    <div id="UL">Upper Left</div>
    <div id="UR">Upper Right</div>
    <div id="LL">Lower Left</div>
    <div id="LR">Lower Right</div>
</body>
</html>
```

---

5. CSS Stands for Cascading Style Sheets. What is meant by cascading? Why would you take advantage of this behavior?

Answer:

CSS is basically used to design specific Styles to specific elements of the webpages. Thus many elements of the same webpage can have their own specific styles and properties. But the browser in itself does not know when and which property to apply. Thus comes in the term Cascading. Cascading is a kind of sorting algorithm which decides the order in which the properties are applied.

So when you visit a page in a web browser, the browser uses its own style sheet. The order of precedence is :

Author Style Sheet
User Style Sheet
User Agent Style Sheet.

The only exception is when !important is used, the User Style Sheet gets precedence over Author Style sheets.

Regular Expressions

1. Describe what the following Perl regular expression is doing:

```
/^(\d+)\.(\d+)\.(\d+)\.(\d+)$/
```

Answer:

The regex wants 1 or more digits followed by a period then one or more digits followed by a period then a series of one or more digits followed by a period and again one or more digits followed by a period . Used mainly for IP addresses.

Example – 1.2.3.4 or 1111.2222.33.4

2. What might the regular expression in the previous question be matching? How could you improve it to perform this task better?

Answer:

The regex is looking for an IPv4 Address. We can improve this by limiting the number of digits allowed to be between 1 and 3.

```
1.
/^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$/

Or

^(?:[0-9]{1,3}\.){3}[0-9]{1,3}$
```

Reference: https://www.safaribooksonline.com/library/view/regular-expressions-cookbook/9780596802837/ch07s16.html

3. What is wrong with the following Perl regular expression? How would you fix it?

```
# Get a token from a string separated by the '|' character
# example string: one|two|three|four
if ($string =~ /(.*)\|/) {
    print "token = $1\n";
}
```

Answer:

The `*` quantifier is greedy, that means it consumes as many as possible. So it will try to read the entire string and not part by part. We need to make this optional and so we will use the non-greedy version `*?`

```
if ($string =~ /(.*?)\|/)
      {  print "token = $1\n"; }
```

---

4. Write a Perl or Javascript regular expression to extract the path from a full linux file location.

```
# /usr/local/bin/mybinary
# regular expression should store "/usr/local/bin" in a variable.
```

Answer:
Regex:

```
my $str = '/var/log/sadasd/10032008.log';
$str =~ m#((?:[^/]*/)*)(.*)#;
my $filepath = $1;
print "$filepath";
```

Reference: http://stackoverflow.com/questions/169008/regex-for-parsing-directory-and-filename

We can also use:

```
use File::Basename;
my $dirname = dirname ("lib/this/that/other.txt");
print "$dirname";
```

Reference:
http://www.perlmonks.org/bare/?abspart=1;displaytype=displaycode;node_id=334461;part=1

Database/SQL

Given the SQL tables below, write an SQL query that will return the `id` and `description` of all `file_type` entries that are in a given `file_type_category`.

```
desc file_type;
+-------------+----------------------+------+-----+---------+-------+
```

```
| Field       | Type                 | Null | Key | Default | Extra |
+-------------+----------------------+------+-----+---------+-------+
| id          | smallint(5) unsigned | NO   | PRI | 0       |       |
| name        | varchar(64)          | YES  |     | NULL    |       |
| description | varchar(256)         | NO   |     |         |       |
+-------------+----------------------+------+-----+---------+-------+

desc file_type_category_map;
+------------------+----------------------+------+-----+---------+-------+
| Field            | Type                 | Null | Key | Default | Extra |
+------------------+----------------------+------+-----+---------+-------+
| file_type_id     | smallint(5) unsigned | NO   | MUL | NULL    |       |
| category_id      | smallint(5) unsigned | NO   |     | 0       |       |
+------------------+----------------------+------+-----+---------+-------+

desc file_type_category;
+-------+----------------------+------+-----+---------+-------+
| Field | Type                 | Null | Key | Default | Extra |
+-------+----------------------+------+-----+---------+-------+
| id    | smallint(5) unsigned | NO   | PRI | 0       |       |
| name  | varchar(256)         | NO   |     |         |       |
+-------+----------------------+------+-----+---------+-------+
```

Sample Data

```
file_type                              file_type_category_map         file_type_category
+-----+-------+----------------------+ +--------------+-------------+ +----+------------------+
| id  | name  | description          | | file_type_id | category_id | | id | name             |
+-----+-------+----------------------+ +--------------+-------------+ +----+------------------+
| 234 | REC   | Realplayer REC       | |          234 |           1 | |  5 | PDF files        |
| 233 | MNY   | MS Money file        | |          234 |           2 | |  4 | Executables      |
| 232 | ACCDB | MS Access 2007 file  | |          233 |           1 | |  3 | Multimedia       |
| 231 | MDB   | MS Access file       | |          232 |           3 | |  2 | Archive          |
+-----+-------+----------------------+ |          232 |           4 | |  1 | Office Documents |
                                       |          232 |           5 | +----+------------------+
                                       |          231 |           1 |
                                       +--------------+-------------+
```

Hints

1. `file_type.id = file_type_category_map.file_type_id`
2. `file_type_category_map.category_id = file_type_category.id`
3. Use `'?'` for the category name you're searching for.

Answer:

```
SELECT id,description
FROM (SELECT file_type_id fid
       FROM file_type_category_map,file_type_category
       WHERE file_type_category_map.category_id = file_type_category.id) AS
buffer WHERE id=fid
```

----------OR----------

```
SELECT id,description
FROM file_type
INNER JOIN file_type_category_map ON
file_type.id=file_type_category_map.file_type_id
```

```
INNER JOIN file_type_category ON
file_type_category_map.catepgry_id=file_type_category.id
```

Reference: http://www.dofactory.com/sql/join