Nom:
Prénom:
N°d'étudiant :

Examen du 5 mars 2019 de

Technologies avancées du *e-learning* 2
JavaScript

Master DDL — Parcours DILIPEM
Université Grenoble-Alpes

Durée 55 minutes.
Supports Tous documents autorisés. En revanche, il est interdit d'exécuter du code.
Remarque Les questions munies d'un signe ♣ peuvent avoir zéro, une ou plusieurs réponses correctes. Les autres questions n'ont qu'une réponse correcte.
Le barème est sur 24, mais des points négatifs peuvent être attribués pour les *très mauvaises* réponses.
Les questions et réponses aux questions n'apparaissent pas dans le même ordre sur toutes les copies...

1 Types (5 pts)

Toutes les questions de cette section portent sur le code suivant :

```
1 var uneVariable = true ;
2 var test = 1 ;
3 var essai = "A" ;
4 var varPourExam = [test, essai] ;
5 var complex = varPourExam[test]+2 ;
6 var aTester = uneVariable || false ;
```

Question 1 (0,5 pts) Dans le code ci-dessus, quel est le type de la variable `uneVariable` ?

☐ `string` ☐ `object` (tableau) ☐ `number` ☐ `object` (autre) ■ `boolean`

Question 2 (1 pt) Dans le code ci-dessus, quel est le type de la variable `essai` ?

■ `string` ☐ `object` (autre) ☐ `object` (tableau) ☐ `number` ☐ `boolean`

Question 3 (0,5 pts) Dans le code ci-dessus, quel est le type de la variable `test` ?

■ `number` ☐ `string` ☐ `boolean` ☐ `object` (autre) ☐ `object` (tableau)

Question 4 (1 pt) Dans le code ci-dessus, quel est le type de la variable `varPourExam` ?

■ `object` (tableau) ☐ `string` ☐ `number` ☐ `boolean` ☐ `object` (autre)

Question 5 (1 pt) Dans le code ci-dessus, quel est le type de la variable `complex` ?

☐ `number` ☐ `object` (tableau) ■ `string` ☐ `boolean` ☐ `object` (autre)

Explication : `varPourExam[test]` revient à `varPourExam[1]` est donc la seconde case du tableau, qui a été remplie avec `essai`, une `string`. L'opérateur + concatène et renvoie une chaine de caractères quand l'un des opérandes est une chaine de caractères.

Question 6 (1 pt) Dans le code ci-dessus, quel est le type de la variable `aTester` ?

☐ `string` ☐ `number` ☐ `object` (tableau) ■ `boolean` ☐ `object` (autre)

Explication : Les deux opérandes sont des booléens.

2 Débuggage (9 pts)

Question 7 ♣ (5 points, -1 point par erreur...)
Cochez tous les numéros de lignes *contenant une erreur* dans le programme suivant

```
1  function itemize(elt){
2     var ps ;
3     ps = elt.getElementsByTagName("p")
4     list = document.createElement("ul");
5     while(ps.length > 0){
6        var newElt = document.createElement(li);
7        newElt.innerHTML = ps[0].innerHTML;
8        ps[0].remove();
9        list.appendChild(newElt);
10    }
11    elt.appendChild(liste);
12 }
```

☐ 1 ☐ 2 ■ 3 ■ 4 ☐ 5 ■ 6 ☐ 7 ☐ 8
☐ 9 ■ 11

Explication :

- Ligne 3 : pas de ;
- Ligne 4 : pas de `var`
- Ligne 6 : `li` n'est pas entre guillemets (string attendue)
- Ligne 11 : `liste` au lieu de `list`.

Question 8 (4 points, -1 point si réponse trop fausse...)
Le programme fonctionne presque... À quelle ligne l'erreur se trouve-t-elle ?

```
1  function maximum(tab){
2     var max = false;
3     if(tab.length > 0){
4        max = tab[tab.length-1];
5        for(let i=0;
6            i<tab.length-2;
7            i++){
8           if(max < tab[i]){
9              max = tab[i];
10          }
11       }
12    }
13    return max ;
14 }
```

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ■ 6 ☐ 7 ☐ 8
☐ 9 ☐ 13

Explication : On ne traite pas l'avant dernière case, il faut soit `i<=tab.length-2` soit `i<tab.length-1`.

3 Compréhension de code (7 pts)

Question 9 (2 pts pour la/les réponse(s) juste(s), -1 pour une réponse scandaleuse...)

```
1 function c(x){
2    var interm = 2 * x;
3    return 2 * interm ;
4 }
5 console.log(c(3));
```

Que va afficher la console suite à l'exécution du code ci-dessus ?

- [x] 12
- [ ] "223"
- [ ] "12"
- [ ] "6"
- [ ] false
- [ ] "23"
- [ ] 6

Question 10 ♣ (4 pts)

```
1  function d(elt){
2     if(elt instanceof HTMLElement){//test de la classe de la variable
3        var newElt = document.createElement(elt.tagName) ;
4        newElt.innerHTML = elt.innerHTML ;
5        elt.parentNode.appendChild(newElt) ;
6     }
7     else{
8        console.error("'d' n'a pas ete appelee avec un element du DOM...") ;
9     }
10 }
```

Cochez les propositions décrivant le programme ci-dessus.

- [ ] `d` ajoute un élément frère immédiatement après `elt`.
- [x] `d` modifie le DOM.
- [ ] `d` supprime `elt`.
- [x] L'élément créé par `d` n'a aucun attribut.
- [ ] `d` modifie `elt`.
- [x] `d` crée un nouvel élément.
- [ ] L'élément créé par `d` est identique à `elt`.
- [x] `d` modifie l'élément parent de `elt`.
- [ ] *Aucune de ces réponses n'est correcte.*

Explication : Si `elt` n'est pas le dernier fils de son élément parent, l'élément recopié (qui n'a aucun des attributs de `elt`) n'est pas situé juste après ce dernier. Comme on ajoute un frère à `elt`, on modifie son parent et donc le DOM.

4 Problème (5 pts)

Question 11 (4 pts)

On a à notre disposition la fonction `selectCells(tab,even,reverse)` (cf. annexes), qui prend le tableau `tab` et renvoie ses cases paires (si `even` vaut `true`) ou impaires (si `even` vaut `false`). Les cases sélectionnées sont renvoyées en partant du dernier élément si `reverse` est spécifié et vaut `true`.

- `selectCells([1,2,3,4],true)` renvoie `[2,4]`
- `selectCells([1,2,3,4],true,true)` renvoie `[4,2]`
- `selectCells([1,2,3,4],false)` renvoie `[1,3]`
- `selectCells([1,2,3,4],false,true)` renvoie `[3,1]`
- `selectCells([1,2,3,4,5],true)` renvoie`[2,4]`
- `selectCells([1,2,3,4,5],true,true)` renvoie `[4,2]`
- `selectCells([1,2,3,4,5],false)` renvoie `[1,3,5]`
- `selectCells([1,2,3,4,5],false,true)` renvoie `[5,3,1]`

Dans une veine « *oulipienne* », on voudrait pouvoir prendre un tableau de lignes de poème et les réordonner. On veut prendre les lignes impaires dans le sens inverse puis les faire suivre par toutes les lignes paires. Par exemple, à partir du sublime poème suivant :

L'informatique n'est pas facile
L'ordinateur n'en fait qu'à sa tête
Je ne veux pas me faire de bile
Quand ça marche c'est la fête

on voudrait pouvoir obtenir :

Je ne veux pas me faire de bile
L'informatique n'est pas facile
L'ordinateur n'en fait qu'à sa tête
Quand ça marche c'est la fête

Cochez la série d'instructions permettant d'obtenir un programme résolvant notre problème de poète...

```
1  var maPoesie = ["L'informatique n'est pas facile",
2     "L'ordinateur n'en fait qu'a sa tete",
3     "Je ne veux pas me faire de bile",
4     "Quand ca marche c'est la fete"] ;
5  //2 lignes manquantes
6  for(let i=0;i<tab2.length;i++){
7     maPoesieTransformee.push(tab2[i]);
8  }
9  var texte = maPoesieTransformee.toString();
10 alert(texte);
```

- ☐
```
var texte = selectCells(maPoesie,false,true);
var maPoesieTransformee = selectCells(maPoesie,true);
```
- ■
```
var maPoesieTransformee = selectCells(maPoesie,false,true);
var tab2 = selectCells(maPoesie,true);
```
- ☐
```
var tab2 = selectCells(maPoesie,false,true);
var maPoesieTransformee = selectCells(maPoesie,true);
```
- ☐
```
var maPoesieTransformee = maPoesie[2];
var tab2 = selectCells(maPoesie,true);
```
- ☐
```
var maPoesieTransformee = selectCells(maPoesie,false);
var tab2 = selectCells(maPoesie,true,true);
```

Annexes

Ci-dessous le code de la fonction `selectCells` (problème) fourni à titre indicatif (il n'est pas nécessaire de comprendre le code pour faire l'exercice):

```
function selectCells(tab,even,reverse){
   //renvoie un tableau avec les cases paires ou impaires
   //dans l'ordre ou en partant de la fin
   var res = false;
   if(typeof reverse == "undefined"){
      reverse = false;
   }
   if(tab.length > 0){
      var i;
      if(reverse){//on part de la derniere case...
         if(even){//...paire (d'index impair)
            i = Math.floor((tab.length)/2) * 2 - 1;
         }
         else{//...impaire (d'index pair)
            i = Math.floor((tab.length-1)/2) * 2;
         }
      }
      else{
         if(even){
            //on veut commencer par la 2e case
            i = 1;
         }
         else{
            //on veut commencer par la 1re case
            i = 0;
         }
      }
      res=[];
      while((!even && res.length < tab.length/2)
            ||(even && res.length < Math.floor(tab.length/2))) {
         res.push(tab[i]);
         if(reverse){
            i = i - 2 ;
         }
         else{
            i = i + 2 ;
         }
      }
   }
   return res;
}
```