Exercices

Ces exercices vous donneront des exemples d'algorithmes répondant à une problématiques données. L'ensemble de ces algorithmes peut former une base de révision algorithmique car ce document couvre les éléments nécessaires pour construire la plupart des algorithmes procéduriaux existant.

Problèmes

Vous trouverez la liste des problèmes et une de leurs solutions algorithmiques ci-dessous.

1. Echanger les valeurs de 2 variables

Problème : Echanger les valeurs de 2 variables R et Q

```
R <- 5
Q <- 34
```

Algorithme :

```
A <- R
R <- Q
Q <- A
```

Traduction JS :

```
// enonce
$R = 5;
$Q = 34;

// algo
$A = $R;
$R = $Q;
$Q = $A;

// tester
var_dump($R);
var_dump($Q);
```

2. Echanger les valeurs de 3 variables

Problème : Echanger les valeurs de 3 variables R, Q et V. Dans V, mettre Q, dans Q, mettre R et dans R, mettre V.

```
R <- 5
Q <- 34
V <- 45
```

Algorithme :

```
A <- V
V <- Q
Q <- R
R <- A
```

Traduction JS :

```
// enonce
$R = 5;
$Q = 34;
$V = 45;

// algo
$A = $V;
$V = $Q;
$Q = $R;
$R = $A;

// tester
var_dump($R);
var_dump($Q);
var_dump($V);
```

3. Vérifier si un utilisateur est majeur

Problème : Vérifier si un utilisateur est majeur grâce à son âge. Mettre une variable `resultat` à `Vrai` si cela est le cas, sinon mettez la à `Faux`.

```
age <- 25
```

Algorithme :

```
@Si age >= 18
    @DebutBloc
    resultat <- Vrai
    @FinBloc
@Sinon
    @DebutBloc
    resultat <- Faux
    @FinBloc
```

Traduction JS :

```
// enonce
$age = 25;

// algo
if ($age >= 18)
{
    $resultat = true;
}
else
{
    $resultat = false;
}

// tester
var_dump($resultat);
```

ou

```
resultat <- Faux
@Si age >= 18
    @DebutBloc
    resultat <- Vrai
    @FinBloc
```

Traduction JS :

```
// enonce
$age = 25;

// algo
$resultat = false;
if ($age >= 18)
{
    $resultat = true;
}

// tester
var_dump($resultat);
```

4. Vérifier si un utilisateur est mineur

Problème : Vérifier si un utilisateur est mineur grâce à son âge. Mettre une variable `resultat` à `Vrai` si cela est le cas, sinon mettez la à `Faux`.

```
age <- 12
```

Algorithme :

```
resultat <- Faux
@Si age < 18
    @DebutBloc
    resultat <- Vrai
    @FinBloc
```

Traduction JS :

```
// enonce
$age = 12;

// algo
$resultat = false;
if ($age < 18)
{
    $resultat = true;
}

// tester
var_dump($resultat);
```

5. Echanger les 2 premiers éléments d'un tableau

Problème : Echanger les 2 premiers éléments d'un tableau

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
A <- tab[0]
tab[0] <- tab[1]
tab[1] <- A
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$A = $tab[0];
$tab[0] = $tab[1];
$tab[1] = $A;


// tester
var_dump($tab);
```

6. Compter le nombre d'éléments dans un tableau

Problème : Compter le nombre d'éléments dans un tableau. Mettre le compte dans `resultat`.

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
resultat <- 0
@PourChaque element @Dans tab
    @DebutBloc
    resultat <- resultat + 1
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$resultat = 0;
foreach ($tab as $element)
{
    $resultat = $resultat + 1;
}

// tester
var_dump($resultat);
```

7. Faire la somme des éléments d'un tableau

Problème : Faire la somme des éléments d'un tableau

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
resultat <- 0
@PourChaque prix_en_cours @Dans tab
    @DebutBloc
    resultat <- resultat + prix_en_cours
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$resultat = 0;
foreach ($tab as $prix_en_cours)
{
    $resultat = $resultat + $prix_en_cours;
}

// tester
var_dump($resultat);
```

ou

```
compte <- 0
@PourChaque element @Dans tab
    @DebutBloc
    compte <- compte + 1
    @FinBloc

resultat <- 0
@Pour i @De 0 @A compte
    @DebutBloc
    resultat <- resultat + tab[i]
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$compte = 0;
foreach ($tab as $element)
{
    $compte = $compte + 1;
}

$resultat = 0;
for ($i = 0; $i < $compte; ++$i)
{
    $resultat = $resultat + $tab[$i];
}

// tester
var_dump($resultat);
```

8. Trouver l'élément maximum d'un tableau

Problème : Trouver l'élément maximum d'un tableau comprenant des nombres de 0 à 1000. Mettre le numbre maximum dans `resultat`.

```
tab <- [...]
```

Algorithme :

```
resultat <- 0
@PourChaque prix_en_cours @Dans tab
    @Debutbloc
    @Si prix_en_cours > resultat
        @DebutBloc
        resultat <- prix_en_cours
        @FinBloc
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$resultat = 0;
foreach ($tab as $prix_en_cours)
{
    if ($prix_en_cours > $resultat)
    {
        $resultat = $prix_en_cours;
    }
}

// tester
var_dump($resultat);
```

9. Trouver l'élément minimum d'un tableau

Problème : Trouver l'élément minimum d'un tableau comprenant des nombres de 0 à 1000

```
tab <- [...]
```

Algorithme :

```
resultat <- tab[0]
@PourChaque prix_en_cours @Dans tab
    @Debutbloc
    @Si prix_en_cours < resultat
        @DebutBloc
        resultat <- prix_en_cours
        @FinBloc
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$resultat = $tab[0];
foreach ($tab as $prix_en_cours)
{
    if ($prix_en_cours < $resultat)
    {
        $resultat = $prix_en_cours;
    }
}

// tester
var_dump($resultat);
```

10. Trouver le premier élément supérieur à 500

Problème : Trouver le premier élément supérieur à 500. S'il n'y en a pas, le resultat doit être 0

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
resultat <- 0
@PourChaque element @Dans tab
    @DebutBloc
    @Si element > 500
        @DebutBloc
        resultat <- element
        @Stop
        @FinBloc
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$resultat = 0;
foreach ($tab as $element)
{
    if ($element > 500)
    {
        $resultat = $element;
        break;
    }
}

// tester
var_dump($resultat);
```

ou

```
longueur <- 0
@PourChaque element @Dans tab
    @DebutBloc
    longueur <- longueur + 1
    @FinBloc

i <- 0
@TantQue tab[i] < 500 @Et i <= longueur - 1
    @DebutBloc
    i <- i + 1
    @FinBloc

resultat <- 0
@Si tab[i] >= 500
@DebutBloc
    resultat <- tab[i]
@FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$longueur = 0;
foreach ($tab as $element)
{
    $longueur = $longueur + 1;
}

$i = 0;
while ($tab[$i] < 500 && $i <= $longueur -1)
{
    $i = $i + 1;
}

$resultat = 0;
if ($tab[$i] >= 500)
{
    $resultat = $tab[$i];
}

// tester
var_dump($resultat);
```

11. Copier un tableau

Problème : Copier le tableau `tab` dans un autre tableau `autre`

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
autre <- []
@PourChaque element @Dans tab
    @DebutBloc
    Ajouter element Dans autre
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$autre = [];
foreach ($tab as $element)
{
    $autre[] = $element;
}

// tester
var_dump($autre);
```

ou

```
autre <- tab
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$autre = $tab;

// tester
var_dump($autre);
```

12. Copier les premiers éléments d'un tableau dont la somme fait au moins 500

Problème : Copier les premiers éléments d'un tableau dont la somme fait au moins 500

```
tab <- [23, 4, 2, 543, 34, ...]
```

Algorithme :

```
autre_tableau <- []
somme <- 0
@PourChaque prix_en_cours @Dans tab
    @DebutBloc
    somme <- somme + prix_en_cours
    Ajouter prix_en_cours Dans autre_tableau
    @Si somme >= 500
        @DebutBloc
        @Stop
        @FinBloc
    @FinBloc
```

Traduction JS :

```
// enonce
$tab = [23, 4, 2, 543, 34, 12, 4];

// algo
$autre_tableau = [];
$somme = 0;
foreach ($tab as $prix_en_cours)
{
    $somme = $somme + $prix_en_cours;
    $autre_tableau[] = $prix_en_cours;
    if ($somme >= 500)
    {
        break;
    }
}

// tester
var_dump($autre_tableau);
```