Exercice 1

1) $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 1 \times 8 + 0 \times 4 + 0 \times 2 + 0 \times 1$

Donc 1000_2 vaut 8_{10}

$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 1 \times 16 + 0 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

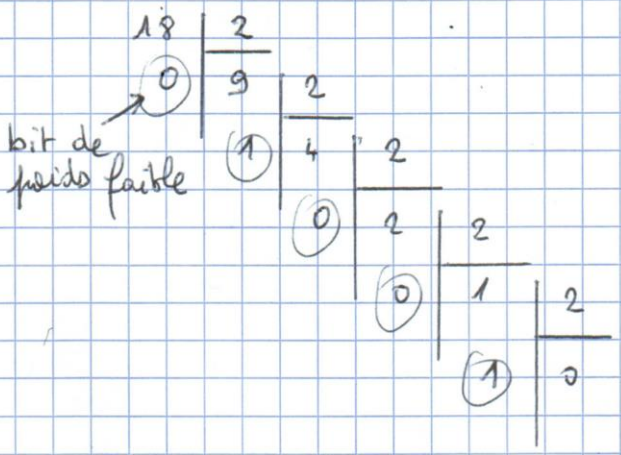
Donc 10011_2 vaut $16 + 2 + 1$ soit 19_{10}

$1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 4 + 1$

Donc 100101_2 vaut 37_{10}

2) $4_{10} = 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$ Donc 4_{10} vaut 100_2

$12_{10} = 1 \times 2^3 + \times 2^2 + 0 \times 2^1 + 0 \times 2^0$ Donc 12_{10} vaut 1100_2



Donc 18_{10} vaut 10010_2

3) a) Compléter le fichier HTML : voir la feuille de l'énoncé ci-après

b) Compléter le fichier JavaScript : voir la feuille de l'énoncé ci-après

- script.js

```
function leibniz(){
   var s = .0...;  // Initialisation de la variable qui cumule la somme des bits.
    if (document.getElementById("b0").checked){s = s + 1}
    if (document.getElementById("b1").checked){s = s + 2}
    if (document.getElementById("b2").checked){S.=.S.+.4..}
    if (document.getElementById("b3").checked){S.=.S.+.8..}
    if (document.getElementById("b4").checked){S.=.S.+16.}
    if (document.getElementById("b5").checked){S.=.S.+32.}
    if (document.getElementById("b6").checked){S.=.S.+.64.}
    if (document.getElementById("b7").checked){S.=..S+128.}
    document.getElementById(".Nombre....").innerHTML = s
}
```

- binaire.html

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Convertisseur binaire</title>
  <link href="..style.css..." type="text/css" rel="stylesheet">
</head>
<body>
  <h3>Convertisseur binaire vers d&eacutecimal &agrave 8 bits</h3>
  <label>Cochez :</label>
  <form autocomplete="off"> <!-- Pour effacer les cases lors du rafraichissement -->
  <div class="ligne">
    <input type="checkbox" id="b7" onChange="..leibniz().">
    <label for="b7"><var>2<sup>7</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b6" onChange="..leibniz(.)">
    <label for="b6"><var>2<sup>6</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b5" onChange="...leibniz(.)">
    <label for="b5"><var>2<sup>5</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b4" onChange="...leibniz(.).">
    <label for="b4"><var>2<sup>4</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b3" onChange="...leibniz()">
    <label for="b3"><var>2<sup>3</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b2" onChange="...leibniz()">
    <label for="b2"><var>2<sup>2</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b1" onChange="...leibniz()">
    <label for="b1"><var>2<sup>1</sup></var></label>
  </div>
  <div class="ligne">
    <input type="checkbox" id="b0" onChange="...leibniz(.).">
    <label for="b0"><var>2<sup>0</sup></var></label>
  </div>
  </form>
  <p>Le nombre repr&eacutesent&eacute en binaire ci-dessus est
  <code id="nombre">0</code>.</p>
  <script src="..script.js..."></script>
</body>
</html>
```

Exercice 2

1) Algorithme : tri_insertion (tableau)

```
pour i allant de 1 à longueur(tableau) -1
    j ← i
    x ← tableau[j]
    tant que j > 0 et tableau[j-1] > x
        tableau[j] ← tableau[j-1]
        j ← j-1
    tableau[j] ← x
renvoyer tableau
```

2)

```
def tri_insertion(tableau_a_trier):
    for i in range(1, len(tableau_a_trier)):
        j = i
        x = tableau_a_trier[j]
        while j > 0 and tableau_a_trier[j-1] > x :
            tableau_a_trier[j] = tableau_a_trier[j-1]
            j = j-1
        tableau_a_trier[j] = x
    return tableau_a_trier
```

3) a)

```
def tri_insertion(liste):
    #  Tri par insertion
    tableau_a_trier = list(liste)  #  Crée une copie de liste
    for i in range(1, len(tableau_a_trier)):
        j = i
        x = tableau_a_trier[j]
        while j > 0 and tableau_a_trier[j-1][0] < x[0]:
            tableau_a_trier[j] = tableau_a_trier[j-1]
            j = j-1
        tableau_a_trier[j] = x
    return tableau_a_trier
```

3) b)
```
def plus_hauts1(liste_triee, n):
    liste_des_n_plus_hauts = liste_triee[:n]
    return liste_des_n_plus_hauts
```

3) c)
```
def plus_hauts2(liste_triee, altitude):
    L = [x for x in liste_triee if int(x[0]) >= altitude]
    return L
```

Exercice 3

```
def recherche_indices_classement(elt, tab):
    L1 = []
    L2 = []
    L3 = []
    for i in range(len(tab)):
        if tab[i] < elt:
            L1.append(i)
        elif elt == tab[i]:
            L2.append(i)
        else:
            L3.append(i)

    return L1, L2, L3
```

Exercice 4

```
def moyenne(nom, dico_result):
    if nom in dico_result
        notes = dico_result[nom]
        total_points = 0.....
        total_coefficients = 0.....
        for valeurs....  in notes.values():
            note, coefficient = valeurs
            total_points = total_points + ..note..... * coefficient
            total_coefficients = total_coefficients.... + coefficient
        return round(total_points/ total_coefficients, 1 )
    else:
        return -1
```