Exercice 1 *Suite arithmétique*

Soit $(u_n)_{n\geqslant 1}$ une suite arithmétique de raison 4 et telle que $u_5 = 30$.

1. Calculer u_{10}.
2. Soit un entier $n \geqslant 1$, exprimer u_n en fonction de n.
3. Soit un entier $n \geqslant 1$, exprimer la somme de termes consécutifs $\sum_{k=1}^{n} u_k = u_1 + u_2 + \ldots + u_{n-1} + u_n$ en fonction de n.

Exercice 2 *Suite géométrique*

Soit $(v_n)_{n\geqslant 0}$ une suite géométrique de raison 0,25 telle que $v_4 = 10$.

1. Calculer la valeur exacte de v_1.
2. Soit un entier $n \geqslant 0$, exprimer v_n en fonction de n.
3. Soit un entier $n \geqslant 1$, exprimer la somme de termes consécutifs $\sum_{k=0}^{n} v_k = v_0 + v_1 + \ldots + v_{n-1} + v_n$ en fonction de n.

Exercice 3

Ce QCM comporte quatre questions ; pour chacune d'elles, quatre réponses sont proposées : une seule est exacte.

1. Soit (v_n) une suite définie par $v_0 = 8$ et telle pour tout entier naturel n, $v_{n+1} = 2v_n - n + 3$.

   Réponse A : pour tout entier naturel $n \geqslant 1$, $v_n = 2v_{n-1} - n + 3$

   Réponse B : $v_2 = 37$

   Réponse C : $v_2 = 40$

   Réponse D : la suite (v_n) est géométrique de raison 2

2. Soit (u_n) une suite arithmétique de raison 4 et telle que $u_1 = 20$.
   Pour tout entier $n \geqslant 0$, on a :

   Réponse A : $u_n = 16 + 4n$

   Réponse B : $u_n = 5 \times 4^n$

   Réponse C : $u_n = 20 + 4n$

   Réponse D : $u_n = 20 \times 4^n$

3. Soit (u_n) une suite géométrique de raison 4 et telle que $u_1 = 20$.
   Pour tout entier $n \geqslant 0$, on a :

   Réponse A : $u_n = 16 + 4n$

   Réponse B : $u_n = 5 \times 4^n$

   Réponse C : $u_n = 20 + 4n$

   Réponse D : $u_n = 20 \times 4^n$

4. Soit (u_n) une suite arithmétique de raison $q \neq 1$ et de premier terme $u_0 \neq 0$.
   La somme de termes consécutifs $u_{40} + u_{41} + \ldots + u_{59}$ est égale à :

Réponse A : $u_{40} \times \dfrac{1-q^{19}}{1-q}$ **Réponse B :** $u_{40} \times \dfrac{1-q^{20}}{1-q}$

Réponse C : $19 \times \dfrac{u_{40}+u_{59}}{2}$ **Réponse D :** $20 \times \dfrac{u_{40}+u_{59}}{2}$

5. Soit (u_n) une suite géométrique de raison $q \neq 1$ et de premier terme $u_0 \neq 0$.

La somme de termes consécutifs $u_{40}+u_{41}+\ldots+u_{59}$ est égale à :

Réponse A : $u_{40} \times \dfrac{1-q^{19}}{1-q}$ **Réponse B :** $u_{40} \times \dfrac{1-q^{20}}{1-q}$

Réponse C : $19 \times \dfrac{u_{40}+u_{59}}{2}$ **Réponse D :** $20 \times \dfrac{u_{40}+u_{59}}{2}$

Exercice 4

Une collectivité locale octroie une subvention de 116 610 € pour le forage d'une nappe d'eau souterraine. Une entreprise estime que le forage du premier mètre coûte 130 €; le forage du deuxième mètre coûte 52 € de plus que celui du premier mètre; le forage du troisième mètre coûte 52 € de plus que celui du deuxième mètre, etc. Plus généralement, le forage de chaque mètre supplémentaire coûte 52 € de plus que celui du mètre précédent.
Pour tout entier n supérieur ou égal à 1, on note : u_n le coût du forage du n-ième mètre en euros et S_n le coût du forage de n mètres en euros; ainsi $u_1 = 130$.

1. Calculer u_2 et u_3.

2. Préciser la nature de la suite (u_n). En déduire l'expression de u_n en fonction de n, pour tout n entier naturel non nul.

3. Calculer S_2 puis S_3.

4. Afin de déterminer le nombre maximal de mètres que l'entreprise peut forer avec la subvention qui est octroyée, on considère la fonction Python suivante :

```
def nombre_metre(S) :
  C = 130
  n = 1
  while C <= S :
    n = n + 1
    C = C + ...
  return ...
```

Compléter cet algorithme de sorte que l'exécution de la fonction `nombre_metre(S)` renvoie le nombre maximal de mètres que l'entreprise peut forer avec la subvention octroyée. Justifier votre réponse.

5. On admet que, pour tout entier naturel non nul, $S_n = 26n^2 + 104n$. En déduire la valeur de n que fournit la fonction Python donnée à la question 4. On expliquera la démarche utilisée.

Exercice 5

Bob s'est fixé un objectif : participer à un marathon qui aura lieu très bientôt dans sa ville.
Pour cela, il désire programmer sa préparation au marathon de la manière suivante :

- lors du premier entraînement, il décide de courir 20 km ;
- il augmente ensuite, à chaque entraînement, la distance à courir de 5 %.

On peut modéliser la distance parcourue lors de ses entraînements par une suite (d_n), où, pour tout entier naturel n non nul, le nombre d_n désigne la distance à courir en kilomètre, lors de son n-ième entraînement.
On a ainsi $d_1 = 20$.

1. Calculer d_2, puis vérifier que $d_3 = 22,05$.

2. Pour tout entier naturel n non nul, exprimer d_{n+1} en fonction de d_n.

3. Justifier que, pour tout entier naturel $n \geqslant 1$, $d_n = 20 \times 1,05^{n-1}$.

4. Quelle distance, arrondie à 1 m près, va courir Bob lors de son 10^{e} entraînement ?

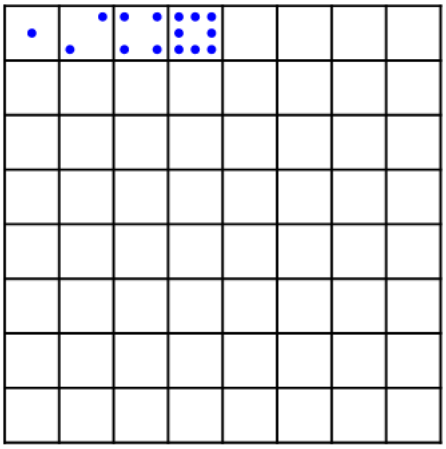
5. La distance à courir lors d'un marathon est de 42,195 km. Bob estime qu'il sera prêt pour la course, s'il parvient à courir au moins 43 km lors d'un de ses entraînements.

Recopier et compléter le script suivant, écrit en langage Python, dont la valeur de n, après exécution de ce script, est le nombre minimal d'entraînements permettant à Bob d'être prêt pour le marathon.

```
n = 1
d = 20
while ......... :
  n = ......
  d = 1.05*d
```

Exercice 6

Une ancienne légende raconte que le jeu d'échecs a été inventé par un vieux sage.
Son roi voulut le remercier en lui accordant n'importe quel cadeau en récompense.
Le vieux sage demanda qu'on lui fournisse un peu de riz pour ses vieux jours, et plus précisément qu'on place : un grain de riz sur la première case du jeu qu'il venait d'inventer, puis deux grains de riz sur la case suivante, puis quatre grains de riz sur la troisième case, et ainsi de suite, en doublant le nombre de grain de riz entre une case et la suivante, et ce jusqu'à la 64e case (puisqu'un plateau de jeu d'échecs comporte 64 cases).



On note u_1 le nombre de grains de riz présents sur la première case, u_2 le nombre de grains sur la deuxième case, et ainsi de suite jusqu'à la 64e case.

1. Déterminer u_1, u_2, u_3, u_4 et u_5.
2. Exprimer, pour tout entier naturel n non nul, u_{n+1} en fonction de u_n.
3. En déduire la nature de la suite (u_n) et en préciser les éléments caractéristiques.

   Exprimer, pour tout entier naturel n non nul, u_n en fonction de n.
4. Calculer le nombre de grains de riz qui doivent être disposés sur le plateau pour satisfaire à la demande du vieux sage.
5. On veut écrire une fonction en langage Python qui détermine à partir de quelle case, le vieux sage disposera d'au moins R grains de riz.

   Une ébauche de cette fonction est donnée ci-contre.

   Recopier et compléter cette fonction afin qu'elle renvoie le résultat désiré.

```
def nb_cases (R) :
    case = 1
    u = 1
    somme = u
    while somme ..... :
        u = ...
        somme = ...
        case = case + 1
    return case
```

Corrigés gracieusement mis à disposition par les collègues de l'APMEP.

Corrigé 1

Une collectivité locale octroie une subvention de 116 610 € pour le forage d'une nappe d'eau souterraine. Une entreprise estime que le forage du premier mètre coûte 130 €; le forage du deuxième mètre coûte 52 € de plus que celui du premier mètre; le forage du troisième mètre coûte 52 € de plus que celui du deuxième mètre, etc. Plus généralement, le forage de chaque mètre supplémentaire coûte 52 € de plus que celui du mètre précédent.
Pour tout entier n supérieur ou égal à 1, on note : u_n le coût du forage du n-ième mètre en euros et S_n le coût du forage de n mètres en euros; ainsi $u_1 = 130$.

1. $u_2 = 130 + 52 = 182$ et $u_3 = 182 + 52 = 234$.

2. La suite (u_n) est arithmétique de premier terme $u_1 = 130$ et de raison $r = 52$.

On en déduit que pour tout $n \geqslant 1$, on a : $u_n = u_1 + (n-1) \times r$ donc $u_n = 130 + 52(n-1)$.

3. $S_2 = u_1 + u_2 = 130 + 182 = 312$ et $S_3 = u_1 + u_2 + u_3 = S_2 + u_3 = 312 + 234 = 546$.

4. Afin de déterminer le nombre maximal de mètres que l'entreprise peut forer avec la subvention qui est octroyée, on considère la fonction Python suivante :

```
def nombre_metre(S) :
  C = 130
  n = 1
  while C < S :
    C = C + ...
    n = n + 1
  return n
```

On veut compléter cet algorithme de sorte que l'exécution de la fonction `nombre_metre(S)` renvoie le nombre maximal de mètres que l'entreprise peut forer avec la subvention octroyée.

La variable `C` contient le coût du forage de `n` mètres.

Pour tout $n \geqslant 1$, on a $S_{n+1} = u_1 + u_2 + \cdots + u_n + u_{n+1} = S_n + u_{n+1}$; or $u_n = 130 + 52\,(n-1)$ donc $u_{n+1} = 130 + 52n$. On a donc $S_{n+1} = S_n + 130 + 52n$.

Il faut donc ajouter $130 + 52n$ à la variable `C` à chaque tour de boucle.

```
def nombre_metre(S) :
  C = 130
  n = 1
  while C < S :
    C = C + 130 + 52*n
    n = n + 1
  return n
```

5. On admet que, pour tout entier naturel non nul, $S_n = 26n^2 + 104n$.

On cherche le plus grand entier n tel que $S_n \leqslant 116\,610$.

On résout l'équation $26n^2 + 104n - 116\,610 = 0$.

$\Delta = 104^2 - 4 \times 26 \times (-116\,610) = 12\,138\,256 = 3\,484^2$

Deux solutions : $n' = \dfrac{-104 + 3\,484}{2 \times 26} = 65$ et $n'' = \dfrac{-104 - 3\,484}{2 \times 26} = -69 < 0$.

Avec 116 610 €, on pourra creuser 65 mètres.

Corrigé 2

Bob s'est fixé un objectif : participer à un marathon qui aura lieu très bientôt dans sa ville.
Pour cela, il désire programmer sa préparation au marathon de la manière suivante :

- lors du premier entraînement, il décide de courir 20 km ;
- il augmente ensuite, à chaque entraînement, la distance à courir de 5 %.

On peut modéliser la distance parcourue lors de ses entraînements par une suite (d_n), où, pour tout entier naturel n non nul, le nombre d_n désigne la distance à courir en kilomètre, lors de son n-ième entraînement.
On a ainsi $d_1 = 20$.

1. $d_2 = d_1 + d_1 \times \dfrac{5}{100} = 20 + 20 \times \dfrac{5}{100} = 21$, et $d_3 = d_2 + d_2 \times \dfrac{5}{100} = 21 + 21 \times \dfrac{5}{100} = 22{,}05$.

2. On passe de d_n à d_{n+1} en ajoutant 5 %, donc en multipliant par $1 + \dfrac{5}{100} = 1{,}05$.

Donc pour tout entier naturel n non nul, $d_{n+1} = 1{,}05 \times d_n$

3. La suite (d_n) est donc géométrique de premier terme $d_1 = 20$, et de raison $q = 1{,}05$.

On en déduit que, pour tout entier naturel $n \geqslant 1$, $d_n = d_1 \times q^{n-1} = 20 \times 1{,}05^{n-1}$.

4. La distance, arrondie à 1 m près, que va courir Bob lors de son 10^{e} entraînement est $d_{10} = 20 \times 1{,}05^9$ soit 31,027 km.

5. La distance à courir lors d'un marathon est de 42,195 km. Bob estime qu'il sera prêt pour la course, s'il parvient à courir au moins 43 km lors d'un de ses entraînements.

On complète le script suivant, écrit en langage Python, dont la valeur de n, après exécution de ce script, est le nombre minimal d'entraînements permettant à Bob d'être prêt pour le marathon.

```
n = 1
d = 20
while d < 43 :
  n = n + 1
  d = 1.05*d
```

Corrigé 3 *Légende de Sissa*

Une ancienne légende raconte que le jeu d'échecs a été inventé par un vieux sage.
Son roi voulut le remercier en lui accordant n'importe quel cadeau en récompense.

Le vieux sage demanda qu'on lui fournisse un peu de riz pour ses vieux jours, et plus précisément qu'on place : un grain de riz sur la première case du jeu qu'il venait d'inventer, puis deux grains de riz sur la case suivante, puis quatre grains de riz sur la troisième case, et ainsi de suite, en doublant le nombre de grain de riz entre une case et la suivante, et ce jusqu'à la 64e case (puisqu'un plateau de jeu d'échecs comporte 64 cases).
On note u_1 le nombre de grains de riz présents sur la première case, u_2 le nombre de grains sur la deuxième case, et ainsi de suite jusqu'à la 64e case.

1. $u_1 = 1$, $u_2 = 2 \times u_1 = 2$, $u_3 = 2 \times u_2 = 4$, $u_4 = 2 \times u_3 = 8$ et $u_5 = 2 \times u_4 = 16$.

2. On double le nombre de grain de riz entre une case et la suivante, donc pour tout entier naturel n non nul, on a $u_{n+1} = 2 \times u_n$.

3. La suite (u_n) est donc géométrique de premier terme $u_1 = 1$ et de raison $q = 2$.
On en déduit que, pour tout n non nul, $u_n = u_1 \times q^{n-1} = 1 \times 2^{n-1} = 2^{n-1}$.

4. Le nombre de grains de riz qui doivent être disposés sur le plateau pour satisfaire à la demande du vieux sage est :

$$S = u_1 + u_2 + u_3 + \cdots + u_{64} = 1 + 2 + 2^2 + \cdots + 2^{63} = 1 \times \frac{1 - 2^{64}}{1 - 2} = 2^{64} - 1.$$

5. On veut écrire une fonction en langage Python qui détermine à partir de quelle case, le vieux sage disposera d'au moins R grains de riz.

Une ébauche de cette fonction est donnée ci-contre.

On complète cette fonction afin qu'elle renvoie le résultat désiré.

```
def nb_cases (R) :
    case = 1
    u = 1
    somme = u
    while somme < R :
        u = 2 * u
        somme = somme + u
        case = case + 1
    return case
```