BCPST 2$_A$

DEVOIR SURVEILLE

MATHÉMATIQUES - INFORMATIQUE

Samedi 6 septembre 2025

(2 heures)

Si au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, il en fait mention dans sa copie et poursuit sa composition. Dans ce cas, il indique clairement la raison des initiatives qu'il est amené à prendre.
La présentation, la qualité de la rédaction, la clarté et la précision des raisonnements *entrent pour une part importante dans l'appréciation des copies. Les candidats sont invités* ***à encadrer*** *leurs résultats.*

L'évaluation en informatique valorisera les solutions efficaces, évitant les calculs superflus.

La calculatrice n'est pas autorisée.

Ce sujet comporte 3 pages.

– –

EXERCICE 1

On considère la suite (u_n) définie par :

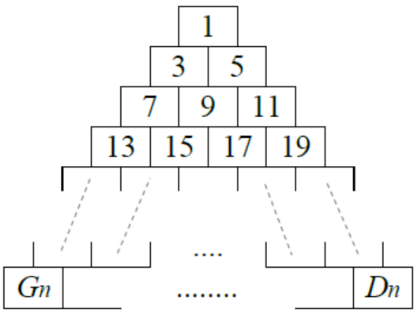
$$u_0 = -2\ ,\ u_1 = -2\ , \qquad \forall n \in \mathbb{N}, \quad u_{n+2} = 4\,u_{n+1} - 4\,u_n$$

1) Ecrire une fonction Python permettant de calculer u_n, sans avoir exprimé u_n en fonction de n.
2) Exprimer u_n en fonction d'un entier naturel n.
3) On note $C_n = \displaystyle\sum_{k=0}^{n} \binom{n}{k} u_k$
   a) Ecrire une fonction Python permettant de calculer C_n en fonction de l'entier n.
   (On pourra ici donner une approche naïve utilisant la fonction `math.comb`[1] et la fonction de la question 1))
   b) Reprendre la question précédente mais cette fois la fonction ne doit contenir qu'une seule boucle et ne doit pas utiliser la fonction `math.comb` et la fonction de la question 1) .
   On pourra remarquer que pour k et n deux entiers naturels : $\binom{n}{k+1} = \dfrac{n-k}{k+1}\binom{n}{k}$
   c) Simplifier les sommes $\displaystyle\sum_{k=0}^{n} \binom{n}{k} 2^k$ et $\displaystyle\sum_{k=0}^{n} k\binom{n}{k} 2^k$.
   d) En déduire $\displaystyle\sum_{k=0}^{n} \binom{n}{k} u_k$ en fonction de l'entier n.
   e) Quelle est la limite de (C_n) ?
   f) Ecrire un programme Python qui permet de déterminer le premier entier n pour lequel $C_n > 100000$.
   On pourra ici utiliser le résultat de la question 3) d)
4) On cherche ici à calculer $S_n = \displaystyle\sum_{k=0}^{n} u_k$.
   a) Simplifier la somme $\displaystyle\sum_{k=0}^{n-2} (4\,u_{k+1} - 4\,u_k)$
   b) En déduire S_n en fonction de l'entier n.

1. `math.comb(n, p)` renvoie la valeur $\binom{n}{k}$

EXERCICE 2

Le but de cet exercice est de retrouver la formule de la somme des premiers cubes à l'aide de la pyramide ci-dessous formée avec la suite des entiers naturels impairs. Le nombre de lignes de cette pyramide est fixé et désigné par n ($n \in \mathbb{N}^*$).



On note N_n le nombre d'entiers dans cette pyramide, G_n l'entier le plus à gauche de la dernière ligne et D_n celui le plus à droite (ainsi $N_4 = 10$, $G_4 = 13$ et $D_4 = 19$).

1) Ecrire une fonction `pyramide(n)` prenant un entier n et qui renvoie une liste de listes contenant les n lignes de cette pyramide.
   Par exemple avec $n = 5$: `In [1]: pyramide(5)`

   `Out[1]: [[1], [3, 5], [7, 9, 11], [13, 15, 17, 19], [21, 23, 25, 27, 29]]`
2) Exprimer N_n en fonction de n.
3) Exprimer D_n en fonction de n.
4) Exprimer G_n en fonction de n.
5) Calculer S_n la somme de tous les entiers écrits dans cette pyramide, c'est à dire $S_n = 1+3+5+\cdots+D_n$.
6) Calculer la somme ℓ_n de tous les entiers écrits dans la dernière ligne (la n-ième) de cette pyramide, c'est à dire $\ell_n = G_n + \cdots + D_n$.
7) Conclure en montrant que : $\displaystyle\sum_{k=1}^{n} \ell_k = \frac{n^2(n+1)^2}{4}$

EXERCICE 3

On considère une suite $(u_n)_{n\in\mathbb{N}}$ vérifiant :

$$\begin{cases} u_0 \geqslant 0 \\ \forall n \in \mathbb{N}, \quad u_{n+1} = \sqrt{u_n} + \dfrac{1}{n+1} \end{cases}$$

1) Démontrer que, pour tout entier naturel $n \geqslant 2$, $\quad u_n > 1$.
2) Etablir que si $(u_n)_{n\in\mathbb{N}}$ converge vers un réel ℓ, on a nécessairement $\ell = 1$.
3) On suppose dans cette question que $(u_n)_{n\in\mathbb{N}}$ est croissante.
   a) Montrer que cette hypothèse entraîne que : $\lim\limits_{n\to+\infty} u_n = +\infty$.
   b) Etablir pour tout entier naturel n, l'égalité : $u_{n+1} - u_n = \sqrt{u_n}(1 - \sqrt{u_n}) + \dfrac{1}{n+1}$.
   c) En déduire que : $\lim\limits_{n\to+\infty} (u_{n+1} - u_n) = -\infty$.
   d) En déduire une contradiction et conclure.
4) a) Justifier qu'il existe un entier naturel n_0 pour lequel : $u_{n_0+1} < u_{n_0}$.
   b) Etablir, pour tout entier naturel non nul n, l'égalité : $u_{n+1} - u_n = \dfrac{u_n - u_{n-1}}{\sqrt{u_n} + \sqrt{u_{n-1}}} - \dfrac{1}{n(n+1)}$.
   c) Montrer que la suite $(u_n)_{n\in\mathbb{N}}$ est décroissante à partir du rang n_0.
   d) En déduire la limite de la suite $(u_n)_{n\in\mathbb{N}}$.

EXERCICE 4

Pour tout $(n,p) \in \mathbb{N}^* \times \mathbb{N}$, on définit : $S_p(n) = \sum_{k=1}^{n} k^p$.

Le but de cet exercice est de présenter une méthode permettant de calculer $S_p(n)$ par récurrence sur p.

1) Rappeler, sans démonstration, les expressions de $S_0(n)$, $S_1(n)$, $S_2(n)$ et $S_3(n)$ en fonction de l'entier n.
2) Ecrire une fonction Python retournant l'entier $S_p(n)$ (n et p étant donnés en arguments)
3) On fixe dans cette question (n,p) dans $\mathbb{N}^* \times \mathbb{N}^*$.
   a) Montrer que :
   $$\sum_{k=1}^{n}(k+1)^{p+1} = S_{p+1}(n) + (n+1)^{p+1} - 1$$
   b) Soit $k \in \mathbb{N}^*$, développer $(k+1)^{p+1}$ en utilisant la formule du binôme.
   c) En déduire que :
   $$\sum_{k=1}^{n}(k+1)^{p+1} = S_{p+1}(n) + \sum_{\ell=0}^{p}\binom{p+1}{\ell} S_\ell(n)$$
   d) A l'aide des questions précédentes montrer que :
   $$S_p(n) = \frac{1}{p+1}\left((n+1)^{p+1} - 1 - \sum_{\ell=0}^{p-1}\binom{p+1}{\ell} S_\ell(n)\right)$$
4) En utilisant le résultat précédent montrer que :
   $$\forall n \in \mathbb{N}^*, \quad S_4(n) = \frac{n(n+1)(2n+1)(3n^2+3n-1)}{30}$$

EXERCICE 5

Le but de cet exercice est d'étudier le dernier chiffre non nul (de l'écriture en base 10) de la factorielle d'un entier naturel. Exemple : Le dernier chiffre non nul de 10! (= 3628800) *vaut 8.*

1) Rappeler la définition de la factorielle d'un entier naturel. (*On donnera $n!$ pour n allant de* 0 *à* 6)
2) Compléter le programme suivant : *(Il suffit d'écrire la ligne* `4:` *complétée)*

```
1:  n = int(input(" Entrer un nombre entier naturel  : ? "))
2:  P = 1
3:  for k in range(n):
4:      P = ....
5:  print("La factorielle de l'entier " + str(n) + "est égale à " + str(P) )
```

3) Ecrire une fonction `facto(n)` prenant un entier n comme argument et qui renvoie l'entier $n!$.
4) Ecrire une fonction `dernier_chiffre(n)` prenant un entier n comme argument et qui renvoie le dernier chiffre non nul de son écriture en base 10.
5) Ecrire un programme qui détermine combien d'entiers n, compris entre 1 et 1000 inclus, ont une factorielle dont le dernier chiffre non nul le est 2.