Lycée Bellevue – Toulouse **5 décembre 2024**
PCSI 2 – Informatique

Devoir sur machine n°1

Durée : 1h50 – Tous documents autorisés

Les questions encadrées et numérotées doivent être validées par le professeur.

Les instructions précédées du signe ▷ doivent être exécutées, mais ne demandent pas de contrôle par le professeur.

Il est autorisé de sauter une ou plusieurs questions, mais il faut alors bien préciser au professeur quelle est la question pour laquelle il est appelé.

I. Échauffement

Exercice 1 : Triangle de Pascal binaire.

Le *triangle de Pascal binaire* est construit sur le même principe que le triangle de Pascal classique, mais suit les règles d'addition suivantes : $0+0=1+1=0$ et $0+1=1+0=1$. Ses premières lignes sont donc :

$$\begin{array}{ccccccc} & & & 1 & & & \\ & & 1 & & 1 & & \\ & 1 & & 0 & & 1 & \\ 1 & & 1 & & 1 & & 1 \end{array}$$

On note L_n la $n^{\text{ème}}$ ligne de ce triangle.

▷ Écrire une fonction `LigneSuivanteBinaire(L)` qui prend en argument une liste `L` binaire (ne contenant que des 0 et des 1) et renvoyant la ligne qui suit L dans le triangle de Pascal binaire.

▷ Afficher les dix premières lignes du triangle de Pascal binaire.

▷ Représenter les points de coordonnées (n,k) vérifiant $n \in [\![1,100]\!]$ et $L_n[k]=1$.

Pour représenter des points, on utilise la fonction `scatter` au lieu de `plot`. Tester, après avoir importé le module `matplotlib.pyplot` :

```
scatter([-1,3],[2,1],s=100)
show()
```

L'option `s=100` permet de contrôler la taille des points.

Question 1

Montrer le résultat au professeur.

II. Autour des nombres premiers

Exercice 2 : Premières fonctions.

On commence par écrire les fonctions suivantes, qui seront utiles dans les prochains exercices :

- ▷ Écrire une fonction `nbdiviseurs(n)` qui renvoie le nombre de diviseurs de n.
- ▷ Écrire une fonction `estpremier(n)` qui, **en utilisant la fonction** `nbdiviseurs(n)`, teste si un entier `n` est premier.
- ▷ En déduire une fonction `pgdp(n)` qui renvoie le plus grand diviseur premier d de n.

Question 2

En déduire une fonction `factorisation(n)` qui renvoie la décomposition en facteurs premiers de n, sous forme de liste ; par exemple, `factorisation(72)` renvoie `[2,2,2,3,3]`.

Exercice 3 : Théorème des nombres premiers.

Pour tout réel x, on note $\pi(x)$ le nombre de nombres premiers inférieurs ou égaux à x.

- ▷ Coder cette fonction.

Question 3

Représenter la courbe de la fonction π pour $x \in [2,1000]$.

- ▷ Représenter, sur le même dessin, la courbe de la fonction $x \mapsto \frac{x}{\ln(x)}$, à l'aide de la fonction `log` du module `numpy`.
- ▷ Toujours sur le même dessin, représenter la courbe de la fonction $x \mapsto \int_2^x \frac{dt}{\ln(t)}$. Pour cela, on utilisera le module `sympy` :

```
import sympy as sp
t = sp.Symbol('t')
Li = lambda x:sp.integrate(1/sp.log(t),(t,2,x))
```

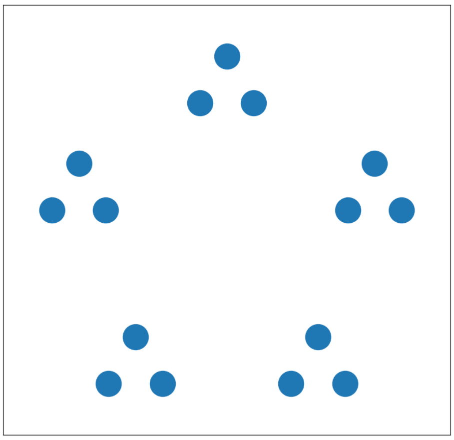
- ▷ Rajouter une légende permettant d'identifier les trois courbes.

Question 4

Montrer le résultat au professeur.

Exercice 4 : Diagramme de factorisation.

On souhaite représenter le *diagramme de factorisation* d'un entier n, sur le modèle ci-dessous (où $n = 15$).



Question 5

Représenter 7 points en cercle.

▷ Écrire une fonction `diagramme(n)` renvoyant les listes X_n et Y_n, respectivement, des abscisses et des ordonnées des points du diagramme de factorisation de n, en suivant l'algorithme suivant :

- si n est premier, les points de ce diagramme sont en cercle,
- sinon, en notant d le plus grand diviseur premier de n : X_n est la liste des $x_d + \frac{x}{d}$ où x_d parcourt X_d et x parcourt $X_{\frac{n}{d}}$; et de même pour Y_n.

Dans cette fonction, X_d et $X_{\frac{n}{d}}$ sont obtenus en appelant la fonction elle-même : `Xd = diagramme(d)` et `Xnd = diagramme(n//d)`.

▷ Afficher le diagramme de 15.

▷ Ajouter comme titre l'écriture de la décomposition de n, sous la forme `15 = 3*5`.

Question 6

Afficher le diagramme de 2025.

Exercice 5 : Spirale d'Ulam. On souhaite représenter les nombres premiers en *spirale*. Pour cela, on associe à chaque entier naturel n un point à coordonnées entières (x,y) du plan. L'entier 1 est associé à l'origine (0,0), puis on suit le parcours suivant :

```
37—36—35—34—33—32—31
|                  |
38  17—16—15—14—13  30
|   |           |   |
39  18  5—4—3   12  29
|   |   |   |   |   |
40  19  6  1—2  11  28
|   |   |       |   |
41  20  7—8—9—10    27
|   |               |
42  21—22—23—24—25—26
|
43—44—45—46—47—48—49...
```

On pourra remarquer qu'en partant de 1, on se déplace 1 fois vers la droite puis 1 fois vers le haut ; puis 2 fois vers la gauche puis 2 fois vers le bas ; puis 3 fois vers la droite puis 3 fois vers le haut, etc.

▷ Effectuer ce parcours jusqu'à $n = 10000$, en stockant les coordonnées des nombres premiers.

Question 7

Représenter ces points.

▷ Reprendre le parcours précédent en représentant chaque entier par un point de taille proportionnelle au nombre de ses diviseurs. On utilisera `scatter(X,Y,S)`, où `S` est la liste des tailles des points.