UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Licence 1 INFO
INFO0101

TP n°5

Le triangle de Pascal

1 Premiers pas avec le triangle de Pascal

Pour rappel, le triangle de Pascal est une présentation des coefficients binomiaux sous la forme d'un triangle. La première ligne du triangle est la ligne numéro 0. Elle comporte uniquement une valeur : 1. Les cinq premières lignes sont donc les suivantes :

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
```

Questions

1. Créez une classe (un programme) nommée `TrianglePascal`.
2. Ajoutez la fonction/procédure `factorielle` et testez-la dans votre `main`.
3. Ajoutez la fonction/procédure `coeffBinomial` permettant de calculer le coefficient binomial. Elle prend les valeurs `n` et `p` en paramètre.
4. Écrivez la fonction/procédure `afficherLigne` qui affiche la nième ligne du triangle de Pascal. La valeur `n` est passée en paramètre.
5. Écrivez la fonction/procédure `afficherTriangle` qui prend en paramètre `n` et qui affiche les `n` premières lignes du triangle de Pascal.
6. Si ce n'est pas déjà fait, modifiez votre `main`. Vous demanderez à l'utilisateur combien de lignes il souhaite puis vous afficherez le triangle demandé.

2 Le beau triangle de Pascal

Normalement, il est plus joli d'afficher le triangle de Pascal de la manière suivante :

```
   1
  1 1
 1 2 1
1 3 3 1
```

Pour réaliser un tel affichage, il y a deux difficultés. Tout d'abord, les lignes doivent être décalées. Ensuite, à partir de la ligne 5, certains coefficients possèdent deux chiffres (deux digits) et tout l'alignement se retrouve chamboulé.

Nous proposons donc, la solution suivante. Tout d'abord, il faut connaître la valeur maximale que peuvent prendre les coefficients binomiaux dans tout le triangle (il est forcément situé sur la dernière ligne). On détermine ensuite son nombre de digits. Ensuite, tous les nombres seront affichés avec le même nombre de digits, en ajoutant des espaces si nécessaire.

Questions

1. Écrivez la fonction/procédure `nbDigits` qui prend en paramètre un entier et qui retourne son nombre de digits (i.e. le nombre de chiffres nécessaires pour l'afficher).

Le nombre de digits d'un nombre correspond au nombre de fois où il est possible de le diviser par 10 avant d'obtenir 0.

2. Écrivez la fonction/procédure `coeffMaximum` qui calcule la valeur du plus grand coefficient binomial pour une ligne `n` donnée. Le numéro de ligne est passé en paramètre.
3. Écrivez la fonction/procédure `afficherEspaces` qui affiche `n` espaces contigus à l'écran avec `n` passé en paramètre.
4. Écrivez la fonction/procédure `afficherValeur` qui prend en paramètre un nombre et un nombre de digits et qui affiche ce nombre sur le nombre de digits spécifiés, en ajoutant des espaces avant si nécessaire. Ainsi, si on affiche 4 sur 3 digits, on affichera 2 espaces puis 4, si on affiche 12 sur 2 digits, on n'affichera pas d'espace avant. On suppose que le nombre de digits est supérieur ou égal au nombre de digits du nombre spécifié.
5. Modifiez la fonction/procédure `afficherLigne`. Elle prend en paramètre un nombre de digits et affiche maintenant les coefficients binomiaux sur le nombre de digits spécifié.

Normalement, la fonction/procédure doit afficher les coefficients binomiaux séparés par un espace. Maintenant, il suffit de séparer les coefficients binomiaux par autant d'espaces que de digits.

6. Modifiez maintenant la fonction `afficherTriangle`. Elle doit calculer le nombre de digits nécessaires pour l'affichage de chaque coefficient binomial. Avant d'appeler la fonction `afficherLigne`, elle doit ajouter les espaces nécessaires : si elle doit afficher `n` lignes, elle affichera à la $i^{ième}$ ligne, `(n - i) x nbDigits` espaces.

3 Algorithmique autour du triangle de Pascal

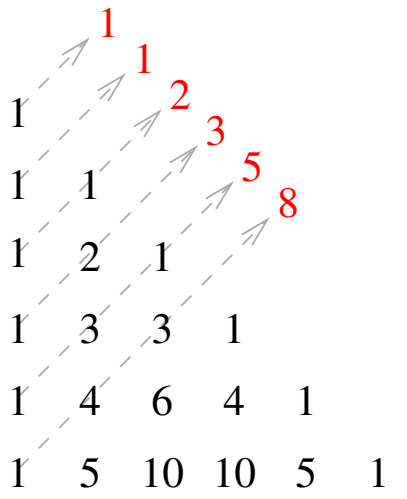
3.1 Généralités

1. Écrivez la fonction/procédure `ligneMax` qui calcule le numéro de la ligne du triangle de Pascal qui possède un coefficient binomial supérieur ou égal à un seuil fixé.
2. Écrivez la fonction/procédure `puissance2` qui retourne la valeur de 2^n. Il s'agit de la somme des coefficients binomiaux de la ligne `n`.
3. Écrivez la fonction/procédure `sommeCoefficients` qui calcule la somme des coefficients binomiaux des `n` premières lignes.

3.2 Fibonacci

Supposons que nous plaçons un couple de lapins dans un enclos fermé. Un couple de lapins génère un nouveau couple de lapins par mois, dès qu'ils ont deux mois. Nous aimerions savoir combien il y a de lapins 1 an après. Le premier mois, il y aura 1 couple, le deuxième mois, 1 couple, le troisième mois, 2 couples, le quatrième mois, 3 couples, le cinquième mois, 5 couples, le sixième mois 8 couples, etc.

La suite 0, 1, 1, 2, 3, 5, etc. est la suite de Fibonacci. Le terme est calculé comme la somme des deux termes précédents avec généralement comme deux premiers termes 0 et 1. Or, en calculant la somme des coefficients binomiaux d'une diagonale ascendante (de droite à gauche), nous obtenons les termes de la suite de Fibonacci. La figure ci-dessous illustre ce principe.



4. Écrivez la fonction `nbLapins` qui calcule le nombre de couples de lapins obtenus après un nombre de mois spécifié.
5. Écrivez la fonction `fibonacci` qui affiche la suite de Fibonacci jusqu'au terme `n`.