1 Entrée en scène à la Samuel Beckett

Dans sa pièce *Quad*, S. Beckett fait entrer et sortir, un par un, les n personnages de sorte que, la scène étant initialement vide, tous les sous-ensembles possibles (il y en a 2^n) de personnages apparaissent exactement une et une seule fois sur la scène. Par exemple (on notera $+i$ l'entrée en scène du personnage i et $-i$ sa sortie de scène), pour quatre personnage le scénario est : $+1, +2, -1, +3, +1, -2, -1, +4, +1, +2, -1, -3, +1, -2, -1$. Dans la suite, on représentera un sous-ensemble de personnage comme un mot binaire dans lequel le bit d'indice i est à `1` si le personnage est présent sur scène et à `0` sinon. Ainsi, le problème est d'énumérer les entiers dont l'écriture en binaire est de longueur n de sorte que d'un entier à l'autre on ne modifie qu'un seul chiffre à la fois (on remarquera au passage qu'énumérer les entiers en ajoutant 1 à chaque fois ne produit pas une suite avec une telle propriété).

1. Donner la suite des entiers correspondant à l'exemple ci-dessus.
2. Montrer que si l'on connaît une solution S_n pour un n donné, une solution pour $n+1$ personnages, S_{n+1} est obtenue de la manière suivante :
   (a) ajouter le caractère `0` à la fin (ou au début) de chaque mot de la solution S_n.
   (b) renverser la suite des mots engendrés par S_n et ajouter le caractère `1` à la fin (ou au début, même choix que précédemment) de chaque mot.
   (c) concaténer les deux listes obtenues.
3. Construire entièrement à la main deux solutions pour S_4 avec le procédé ci-dessus décrit.
4. Écrire la fonction `genFin` permettant de réaliser le procédé ci-dessus (version on ajoute à la fin).
5. Expliquer en quoi le procédé ci-dessus est correct.

2 Coefficients binomiaux

1. Rappeler la définition récursive du calcul des coefficients binomiaux défini pour les entiers naturels n et k.
2. Écrire une version récursive `binomial` du calcul du coefficient binomial de n et k.
3. Combien d'appels à la fonction `binomial` sont-ils effectués si l'on appelle `binomial(3,4)` ? Généraliser sur n et k.
4. Utiliser la technique de programmation dynamique étudiée en cours pour écrire `binomialRapide` améliorant le calcul du coefficient binomial de deux entiers.
5. Combien d'appels à `binomialRapide` sont-ils effectués si l'on appelle `binomialRapide(3,4)` ? Généraliser sur n et k.

3 Calcul d'itinéraire

Vous êtes programmeur chez MaPie, société spécialisée dans le calcul d'itinéraire. La société permet de calculer le plus court chemin entre deux villes de France (on rappelle qu'il y a environ 36 000 communes en France, on supposera qu'elles sont numérotées de 0 à 36 000 dans la suite). Malin, vous récupérez sur Internet de quoi calculer le plus court chemin entre deux villes données à l'aide de la fonction `plusCourt(int v1,int v2)`. Le seul problème est que la fonction qui calcule l'itinéraire est relativement longue en temps de calcul. Comme vous envisagez de contribuer à la fortune des dirigeants de la société pour laquelle vous travaillez, et que vous vous souvenez un peu des cours dispensés autrefois à l'Université, vous pensez utiliser un mécanisme proche de la programmation dynamique pour améliorer le temps de réponse moyen du service. Quoi ? Comment ? Écrivez la fonction `plusCourtRapide(int v1,int v2)`. L'Europe contient environ 100 000 communes. Peut-on encore espérer agir ainsi ?