Informatique en CPGE (2014-2015)
Devoir maison 4

Ce devoir sera fait en groupe (un devoir par groupe de colles).

Chacun des noms des fichiers programmes, un fichier par exercice, doit commencer par le nom d'un élève (les noms des autres élèves ayant participé seront écrits en commentaire au début de chaque programme). La présentation est importante ; pour écrire un commentaire sur une ligne, on commence par le caractère #, pour des commentaires sur plusieurs lignes, on les entoure par des triples guillemets ("""commentaires ...""").

Les programmes seront envoyés en une seule fois par email.

Exercice 1

Algorithme d'Euclide : cet algorithme permet de déterminer le pgcd de deux entiers naturels strictement positifs a et b. Pour cela, on effectue la division euclidienne de a par b, soit $a = bq + r$; si $r = 0$, le pgcd est b, sinon on recommence en remplaçant a et b respectivement par b et r. On reproduit ce schéma tant que le reste est non nul. Le pgcd est le dernier reste non nul.

Par exemple : si $a = 40$ et $b = 25$, alors $a = 25 \times 1 + 15$, puis $25 = 15 \times 1 + 10$, puis $15 = 10 \times 1 + 5$, et enfin $10 = 5 \times 2 + 0$, donc le pgcd est 5.

Ecrire une fonction **pgcd** qui prend en argument deux nombres entiers naturels non nuls et renvoie le pgcd de ces deux nombres. On utilise l'algorithme d'Euclide avec une boucle conditionnelle "while" qui teste le reste à chaque étape.

Exercice 2

Décomposition d'un entier naturel en produit de facteurs premiers.

Partie 1

Crible d'Erastosthène.

Le crible d'Eratosthène est un procédé qui permet de construire la liste des nombres premiers jusqu'à un entier n fixé. Pour cela on procède par élimination : à partir de la liste d'entiers, on enlève 0 et 1, puis on garde 2 et on enlève de la liste les multiples de 2, puis on garde 3 et on retire les multiples de 3 ; 4 a été enlevé, donc le nombre restant qui suit est 5 que l'on garde et on retire les multiples de 5. On procède ainsi en gardant à chaque étape le premier nombre qui suit restant et en éliminant ses multiples.

1. Ecrire une fonction **crible** qui prend en argument un entier naturel n et renvoie une liste de booléens. Chaque booléen sera égal à True si l'indice correspondant est premier et False sinon.

   Le corps de la fonction comportera les instructions qui suivent :

   (a) création d'une liste de n+1 (de 0 à n) booléens tous égaux à True.

   (b) affectation de la valeur False (0 et 1 ne sont pas premiers) aux deux premiers éléments de la liste.

   (c) à l'aide d'une boucle while, affectation de la valeur False à tous les éléments dont l'indice est multiple de deux.

   (d) pour chaque élément d'indice k dont la valeur est True, affectation de la valeur False à tous les booléens indexés par un multiple de k, hormis celui indexé par k. On répète cette étape tant que $k^2 \leq n$.

   (e) renvoie de la liste des n+1 booléens.

2. Ecrire une fonction **premiers** qui prend en argument un entier n et construit, à l'aide de la fonction **crible**, la liste formée de tous les nombres premiers inférieurs ou égaux à n et renvoie cette liste.

Partie 2

Factorisation d'un entier.

Ecrire une fonction **factorise** qui prend en argument un entier n, crée, à l'aide de la fonction **premiers**, la liste des nombres premiers jusqu'à n, puis teste si n est premier et affiche un message en ce sens. Si n n'est pas premier, la fonction détermine alors les différents facteurs premiers et renvoie la liste de ces facteurs.

Partie 3

Décomposition en facteurs premiers.

Ecrire une fonction **decomposition** qui prend en argument un entier n, crée la liste des facteurs premiers de n, à l'aide de la fonction **factorise**. Ensuite la fonction détermine la décomposition de n en facteurs premiers et renvoie la réponse sous forme d'une chaîne de caractères.

Par exemple **factorise(200)** renvoie la chaîne " $200 = 2^3 \times 5^2$ ".