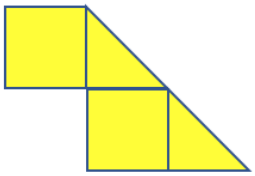
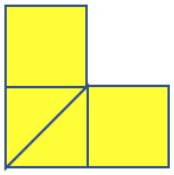
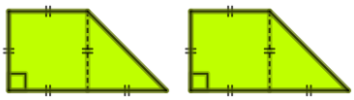
DÉFI N°137 - LES BIMI « L »





Les pièces ci-dessus sont des assemblages de ces deux trapèzes rectangles :



Chaque trapèze rectangle est formé par un carré et un demi-carré accolés.

Dessiner tous les assemblages possibles (par un côté entier). Les trapèzes peuvent être retournés.

DÉFI ALGORITHMIQUE N°137

Certaines énigmes du rallye mathématique de Lorraine auraient certainement été plus simples à résoudre à l'aide d'un petit programme informatique.

Nous vous proposons ici, comme défi, de résoudre l'exercice ci-dessous à l'aide d'un programme.

L'énoncé avait été donné en 2009, il est ici actualisé pour 2019. Proposez une fonction qui renvoie le résultat quel que soit le numéro de l'année.

Dans mon village, les heures s'écoulent au son des cloches : un coup pour 1 et 13 heures, 2 coups pour 2 et 14 heures, etc. jusqu'à 12 coups pour midi et minuit.

Le premier des coups de minuit, dans la nuit du 31 décembre au 1er janvier fut aussi le premier des coups sonnés pour l'année 2019. Quand donc a retenti le 2019ième coup de cette même année ?

SOLUTION DU DÉFI N°136 - NOËL 2018

En 2018, Noël est un mardi et le premier janvier 2019 est aussi un mardi. Dans combien d'années ces deux jours de fête seront-ils de nouveau des mardis ?

En ajoutant un jour à chaque année bissextile, on trouve que Noël 2029 et le 1er janvier 2030 tombent un mardi.
En fait, il y a 7 jours entre Noël et le 1er de l'an qui suit donc ces deux dates tombent toujours le même jour de la semaine.

Retour au sommaire

SOLUTION DU DÉFI ALGORITHMIQUE N°136

Le défi algorithmique du PV 136 demandait de trouver la probabilité que l'heure de réveil en sursaut, entre 22 h 30 et 7 h 30, du commissaire Girard soit un carré parfait. La fonction probaCarreParfait() renvoie la proportion de carrés parfaits entre 22:30 et 07:30 avec une remise à zéro après 23 :59. L'algorithme tient compte des données de l'énoncé sans proposer de généralisation à des heures de début et de fin quelconques. Elle utilise la fonction carréParfait renvoie un booléen indiquant si l'entier passé en paramètre est un carré parfait. Effectuer probaCarreParfait() permet d'obtenir la réponse cherchée.

Pseudo-code :

Fonction carreParfait(n : entier ; booléen)

dec ← $\sqrt{n} - \left[\sqrt{n}\right]$;

renvoyer (dec < 0,0000001).

NB : $[x]$ est la partie entière du réel x.

Fonction probaCarreParfait(; flottant)

h ← 22 ;
m ← 30 ;
nbCarres ← 0 ;
tant que h ≠ 7, **faire** :
si carreParfait($100\mathrm{h} + \mathrm{m}$), **alors** :
nbCarres ← nbCarres +1 ;
finSi ;
m ← m + 1 ;
si $\mathrm{m} = 60$, **alors** :
h ← h+1 ;
m ← 0 ;
si h = 24, **alors** :
h ← 0 ;
finSi ;
finSi ;
finTantque ;
renvoyer $\frac{\mathrm{nbCarres}}{540}$

Retour au sommaire

Code Python :

```
""" Fonction carreParfait(n : entier ; booléen)
n : entier dont on veut savoir si c'est un carré parfait
renvoie un booléen, vrai si n est un carré parfait, faux sinon """

def carreParfait(n):
    dec=sqrt(n)-int(sqrt(n))
    return (dec<0.0000001)



""" Fonction probaCarresParfaits(;flottant)
h : entier, le numéro de l'heure
m : entier, le nombre de minutes
nbcarres : entier, le nombre de carres parfaits entre 2230 et 2359 puis entre
0 et 730
renvoie un flottant, donnant la proportion de carres parfaits  """
def probaCarresParfaits():
    h=22
    m=30
    nbcarres=0
    while h!=7:
        if carreParfait(h*100+m):
            nbcarres+=1
        m+=1
        if m==60:
            h+=1
            m=0
            if h==24:
                h=0
    return nbcarres/860
```

Retour au sommaire