Maths dans SNT

- travail sur la base 2, la base 10 et la base 16 (numération de position…)
- travail sur les puissances de 2, de 10
- comprendre un algorithme
- lire un graphe
- programmer un lancer de dé (et plus si affinité)
- notion de *événement presque certain*

INTERNET

1. Des câbles pour l'Internet

Rappels : volume du cylindre / ordre de grandeur / puissances de 10

1. Connaissant la longueur et le diamètre (d'après la vidéo : https://www.youtube.com/watch?v=Cb7ibgRivwU) estimer le volume du câble enroulé sur le bateau.
2. Comparer ce volume à un volume connu : celui de votre habitation, de la cantine, du CDI...
3. La flotte de la société française *Orange* compte six navires câbliers, dont deux portent le nom de mathématiciens. Pour chacun d'entre eux, à l'aide des brochures techniques (préciser la source) donner le volume de câble qu'il peut transporter.
4. En cinq lignes maximum, dire qui étaient ces mathématiciens (préciser les sources).

Pour compléter :

- une carte des câbles sous marins : https://submarine-cable-map-2014.telegeography.com
- un reportage d'Europe 1 : https://www.europe1.fr/technologies/...
- base de données des câbles : http://www.sigcables.com
- site de la divsion marin d'*Orange* : https://marine.orange.com/fr

2. Protocole IP

2.1 Adresses IPv4 et IPv6

travail sur la base 2, la base 16
1 octet = 8 bits
travail sur les puissances de 10

adresse IPv4

En IPv4, les adresses des machines sont codées sur 4 octets, un octet est un groupe de 8 bits (*BInary digiT* : chiffre binaire). Les adresses sont codées sous la forme d'une succession de 0 et de 1 puis chaque octet est traduit en base dix et est séparé du suivant par un point.

un objet connecté peut avoir plusieurs interfaces, chacune avec son adresse IP...

1. Déterminer le nombre d'adresses IPv4 possibles (théoriques).
2. **Conversion en base 2**
   en base 10, $\overline{abcd}_{10}$ est le nombre représentant $a\cdot10^3+b\cdot10^2+c\cdot10^1+d\cdot10^0$ (a, b, c et d étant des entiers compris entre 0 et 9).
   Décomposer les nombres suivants : • 132 • 9843 • 902
   en base 2, $\overline{abcd}_2$ est le nombre représentant $a\cdot2^3+b\cdot2^2+c\cdot2^1+d\cdot2^0$ (a, b, c et d étant des entiers égaux à 0 ou à 1).
   a) Convertir en base 10 les nombres suivants : • 1001_2 • 10101_2 • 100_2
   b) Convertir en base 2 les nombres suivants : • 45_{10} • 100_{10} • 37_{10}
   c) Une adresse IPv4 s'écrit en base 10 sous forme de quatre entiers séparés par des points (par exemple : 0.1.2.3). Quelles sont les entiers utilisables pour former l'adresse ?
   d) Déterminer les adresses IPv4 suivantes existent sous le format a.b.c.d ; puis retrouver les sites associés : ①
   - 1101 1000.11 1010.1101 0001.1110 0011
   - 1011 0000.10 0010.1001 1011.1 0111
   - 101 0110.100 0001.10 0111.1101

Numworks, module python `bin(45)`, `hex(45)`, `int("0b101101")`

①`host -4 google.fr`

- 101 0111.110 0010.1000 1001.110 0101

Adresse IPv6

Les adresses IPv6 sont codées sur 16 octets. On utilise la base hexadécimale qui comporte donc seize symboles pour écrire les nombres : les chiffres de 0 à 9, puis les lettres de A à F.

1. Déterminer le nombre d'adresses IPv6 possibles (théoriques).
2. Les couleurs sont souvent codées en hexadécimal : les deux premiers chiffres représente le rouge, les deux suivant le vert et les deux derniers le bleu.
   exemple : 1BAFFE signifie
   1B pour rouge, soit $1 \cdot 16^1 + 11 \cdot 16^0 = 27$
   AF pour vert, soit $10 \cdot 16^1 + 15 \cdot 16^0 = 175$
   FE pour vert, soit $15 \cdot 16^1 + 14 \cdot 16^0 = 254$
   donc
   Donner les quantité de rouge, vert, bleu de chacune des couleurs suivante, puis trouver la couleur : 01CAFE ; BACFAC ; C0C010 ; C0C0DA.

2.2 Magie

3. Routage

Le protocole TCP/IP « découpe » les données en paquets et les envoie au destinataire via une route définie par les serveurs.

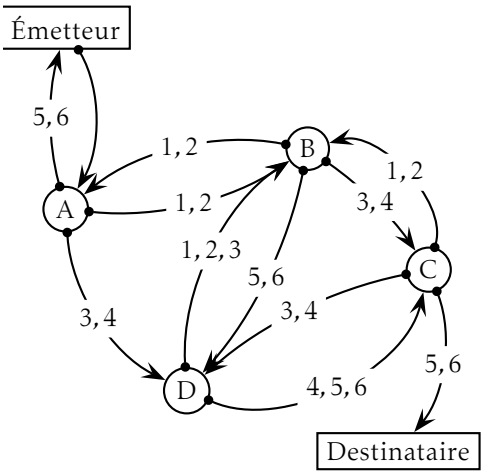
Problème : Supposons un réseaux de 4 serveurs. Quelle devrait être la durée de vie minimale des paquets pour être *quasiment* certains qu'ils arrivent tous à destination ? (attention, le TTL est codé sur un octet.)

https://www.frameip.com/entete-ip/ et https://fr.wikipedia.org/wiki/Time_to_Live, la durée de vie (TTL = Time To Live) serait souvent de 64

Recherche d'une solution :

1. Quelle contrainte impose le fait que le TTL soit codé sur un octet ?
2. Lancer le dé, suivre la route du paquet, à chaque étape augmenter la durée de vie nécessaire de 1 (au début la durée de vie du paquet est égale à 0).
   Mettre en commun les résultats, puis interpréter les données récoltées.

Programmer lancer de dé à 6 faces, statistiques sur l'ensemble des données de la classe, algorithme



Exemple :

Au début le paquet est chez *l'émetteur*.

face du dé	le paquet arrive en	vie nécessaire
n'importe	A	1
5	émetteur	2
5	A	3
2	B	4
1	A	5
1	B	6
4	C	7
2	...	...

```python
#-*- coding : utf8 -*-
# python 3
#
# <...> indique qu'il faut écrire quelque chose,
# éventuellement sur plusieurs lignes !

from random import *

def res():
    """ renvoi le nombre d'étapes pour parcourir le réseau
    de façon aléatoire.
    """
    position = "e" # émeteur
    etape = 0
    while position != "d": #destinataire
        choix = randrange(1,7) # face du dé
        if position == "e":
            position = "A"
        elif position == "A":
            if choix in [1,2]:
                position = "B"
            elif choix in [3,4]:
                position = "D"
            else:
                position = "e"
        elif position == "B":
            if choix in <...>:
                position = "A"
            elif choix in <...>:
                position = "C"
            else:
                position = "D"
        elif position == "C":
            if <...>:
                position = "B"
            elif <...>:
                position = "D"
            else:
                <...>
        else: #position == "D"
            if choix in [1,2,3]:
                <...>
        etape = etape + 1
        print(etape,":",position)
    return etape

def vie(n):
    """ durée de vie nécéssaire pour que le destinataire reçoive
    le message avec 95% de réussite sur un test de n étapes
    """
    nb_etape = []
    for _ in range(n):
        nb_etape.append(res())
    nb_etape.sort()
```

```
55      return nb_etape[int(.95*n)]
```