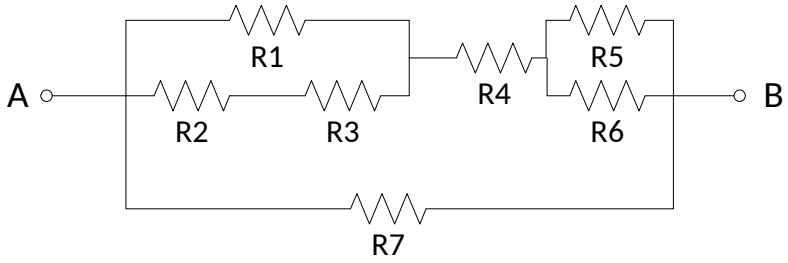
TP N° 02 - Module R107

Le but de ce TP est de se familiariser avec les fonctions python.

1. Introduction :

1. Programmer et tester les fonctions de l'exercice 1 du TD n° 03.
2. Créer ensuite 2 fonctions permettant de calculer la résistance équivalente à deux résistances associées **en série**, puis la résistance équivalente à deux résistances associées **en parallèle**.
3. Calculer alors la résistance équivalente au montage suivant avec R1=R2=R3...R7=100Ω, puis R1=100Ω, R2=200Ω, R3=300Ω, ..., R7=700Ω :



2. Crible d'ÉRATOSTHÈNE :

RAPPELS : Un nombre premier est un nombre entier positif (supérieur à 1) qui se trouve être divisible uniquement par lui-même (et par 1). Par exemple : 2, 3, 5, 7, 11, 13 ... Donc 4 n'est pas un nombre premier car il est divisible par 2 (2x2=4).

Pour vérifier si un nombre N est un nombre premier, on pourrait essayer de diviser celui-ci par tous les nombres entiers <N (et si on ne trouve pas de diviseurs, on dit qu'il est premier), mais cela prendrait trop de temps pour trouver tous les nombres premiers inférieurs à 1000 ou à 10000.

ÉRATOSTHÈNE a établi une méthode beaucoup plus efficace et plus rapide pour trouver tous les nombres premiers inférieurs à une borne donnée :

On écrit tous les nombres jusqu'à MAX (dans une liste par exemple : [2, 3, 4, ... ,MAX]).
On balaye ensuite la liste de gauche à droite.
Si un nombre n'est pas « barré », c'est un nombre premier. Dans ce cas, on « barre » tous ses multiples jusqu'à MAX.
Puis on teste le suivant, etc. jusqu'à MAX.
Les nombres restants sont premiers ! Il ne reste plus qu'à les afficher.

1. Crée une liste contenant les valeurs de 2 jusqu'à MAX inclus.
2. Écrire ensuite une fonction qui réalise le traitement de la liste conformément à la méthode du crible d'ÉRATOSTHÈNE. On représentera un nombre barré par le fait de remplacer sa valeur par zéro dans la liste.

   Voici un exemple pour les nombres de 2 à 20 :

```
2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 (liste de départ)
2 3 0 5 0 7 0 9  0 11  0 13  0 15  0 17  0 19  0 (passe pour p=2)
2 3 0 5 0 7 0 0  0 11  0 13  0  0  0 17  0 19  0 (passe pour p=3)
2 3 0 5 0 7 0 0  0 11  0 13  0  0  0 17  0 19  0 (passe pour p=5)
...
```

3. Écrire une fonction qui permette d'afficher tous les éléments non nuls d'une liste (ce qui reviendra à afficher tous les nombres premiers inférieurs à MAX d'une liste traitée).
4. Comment transformer cette fonction pour n'obtenir que les nombres premiers dans la liste résultat ?
5. Écrire alors un programme qui détermine et affiche tous les nombres premiers inférieurs à 5000. Combien il y en a-t-il ?

3. Décomposition en facteurs premiers :

On cherche maintenant à factoriser un nombre entier en utilisant les nombres premiers que l'on vient de déterminer.

1. Écrire une fonction qui détermine les diviseurs d'un nombre entier en testant pour chaque nombre premier si le reste de la division est nul ou pas.
2. Comment trouver ensuite les puissances de chaque diviseurs ? Donner alors le résultat sous forme de liste de tuples. Par exemple :

```
4030 = 2 * 5 * 13 * 31 => [(2,1), (5,1), (13,1), (31,1)]
4032 = 2**6 * 3**2 * 7 => [(2,6), (3,2), (7,1)]
```

3. Déterminer alors la décomposition des 4 nombres suivants :

```
541204020

541204180

14666966713929330331864225183815839453352960265945 5

571919178299702033669046870804219275856214101777287588042408660158597510
254769770641613240494120978125533485065820417895457123784950793870860224
584642587200817186197185965877491339672508099106217605102057198328150685
465409213979895263552382910118764561187552388823668310376498484490763835
474355716793855993347959663301751310541529298079669259695243644387837891
018477330601209737140610891273512002120332774393587337228796856110314217
130265714190349521398257264975865911094996480752439984399639336362543586
205653378068627066794353228631635745260866608704636208421472801944114216
326572738984638035495391989303792543760597328158060915271085090982249423
017797814887578675621310415072726796351909869101349838404992528325035975
469731561178769021529286047432929112011192615522812041120209204811557345
544499567713469709445290160997787874567099784780423468414395018613620124
228874359624293991994676124193873054543454453735805249234118370015985809
962353076197710834076526302717723771236720498794657988811737109644188865
919994667807763344611689040326930162901752800048973553491 74
```