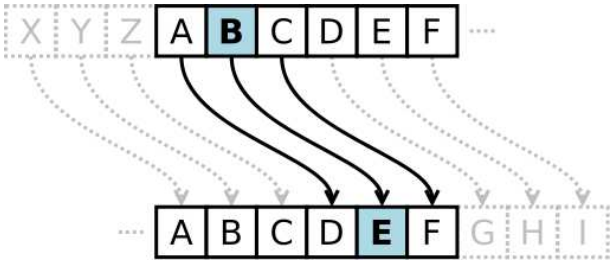
❑ **Exercice 1** : *nombre de chiffres d'un entier*

1. Ecrire une fonction `nombre_chiffres` en C qui prend en argument un entier supposé positif et renvoie le nombre de chiffres de cet entier dans son écriture en base 10. Par exemple `nombre_chiffres(1912)` renvoie 4.
2. Dessiner le graphe de flot de contrôle de cette fonction.
3. Proposer des tests afin de valider le comportement de cette fonction.

❑ **Exercice 2** : *chiffrement de César*

L'une des plus ancienne méthodes de chiffrement est le code de César qui consiste simplement à décaler chaque lettre d'une distance fixe dans l'alphabet. Par exemple si cette distance (appelée clé de chiffrement) est 3, la lettre A devient D, B devient E, et ainsi de suite. Pour les dernières lettres on reprend au début de l'alphabet. Par exemple toujours avec une distance de 3, X devient A, Y devient B et Z devient C. Ce fonctionnement est illustré ci-dessous :



1. Ecrire une fonction `chiffre_lettre` qui prend en argument un caractère `car` et un entier `cle` et renvoie un le caractère `car` chiffré avec le décalage `cle` tel que décrit ci-dessus.
   Par exemple `chiffre_lettre('A',3)` doit renvoyer `'D'`.
2. Modifier la fonction précédente afin que seules les lettres majuscules soient chiffrées, si un autre caractère est envoyée à la fonction `chiffre_lettre` alors ce même caractère est renvoyé par la fonction.
   Par exemple `chiffre_lettre('!',3)` doit renvoyer `'!'`.

❑ **Exercice 3** : *pangramme*

Un *pangramme* est un phrase comportant toutes les lettres de l'alphabet, l'un des exemples les plus connus est la phrase « *portez ce vieux whisky au juge blond qui fume* ».

1. Ecrire une fonction `est_pangramme` qui prend en argument une chaine de caractères écrite en majuscules et renvoie `true` s'il s'agit d'un pangramme et `false` sinon.
2. Proposer un jeu de test afin de valider le comportement de cette fonction.

❑ **Exercice 4** : *nombres narcissiques*

Un entier naturel n s'écrivant avec p chiffres en base 10 est dit *narcissique* lorsqu'il est égal à la somme des puissances $p$ième de ses chiffres. Par exemples, :

— 153 est un nombre narcissique car $1^3 + 5^3 + 3^3 = 153$ $(1 + 125 + 27)$,

— 255 n'est pas un nombre narcissique car $2^3 + 5^3 + 5^3 \neq 250$,

— 1 634 est un nombre narcissique car $1^4 + 6^4 + 3^4 + 4^4 = 1634$.

1. Ecrire la spécification d'une fonction testant si un nombre est narcissique.
2. Ecrire en C cette fonction.
3. En utilisant votre fonction donner la liste des nombres narcissiques compris entre 99 et 10 000.

❑ **Exercice 5** : *moyenne olympique*

La *moyenne olympique* de n notes s'obtient en retirant les deux notes extrêmes (la plus élevée et la plus basse) et en calculant la moyenne des notes restantes. Par exemple :

— la moyenne olympique des notes $12, 10, 13, 14, 16$ est 13,

— la moyenne olympique des notes $16, 16, 16, 16$ est 16,

— la moyenne olympque des notes $5, 1, 4, 3, 2, 6$ est 3.5.

1. Ecrire une fonction `moyenne_olympique` qui prend en argument un tableau de notes et sa taille qu'on supposera supérieure ou égale à 3 et renvoie la moyenne olympique des notes de ce tableau.

❑ **Exercice 6** : *Second maximum*

1. Proposer un algorithme permettant de calculer le deuxième plus grand élément d'un tableau d'entiers. On suppose que le tableau contient toujours plus de deux éléments. Par exemple pour le tableau `[2, 10, 5, 17, 9]`, l'algorithme renvoie 10.
2. Prouver la correction de cet algorithme
3. Proposer une implémentation en langage C et fournir un jeu de tests.