De l'ADN vers le polypeptide:

Commençons par construire l'algorithme de notre programme.

Ne pas copier les lignes commençant par ### .

```
### demander la saisie des bases de l'adn
### soit la variable "adn"
### "input" sert à ouvrir une petite fenêtre pour entrer une valeur

adn=input ("adn non transcrit")

###  avant tout, il faut vérifier si la longueur de l'adn est un multiple de 3
### "len" donne la longueur de la chaine
### "%3" vérifie que la longueur est un multiple de 3
### "raise Exception" renvoie le message d'erreur si "%3" est faux

if (len(adn) %3):
    raise Exception('La séquence doit comporter un nombre de bases multiple de 3 !')

### mettre l'adn en majuscules
adn=adn.upper()
print ("adn: "+adn)

### vérifier si les bases ne sont pas A,T,G ou C

for base in adn:
    if not base in "ATGC":
        raise Exception ("erreur de saisie des bases d'ADN, relancer le programme")
print ("brin adn correct")

### conversion ADN non transcrit en ARNm
### soit "table" une variable
### "maketrans" permet d'échanger les lettres du premier groupe par celle du second

table = str.maketrans('TAGC', 'UAGC')
ARNm= adn.translate(table)
print("ARNm:"+ARNm)

### utilistion ensuite des fonctions utilisant le tableau des codons
### la fonction donnant la sequence d'acides aminés est à définir au début:
### ajouter au début, en première ligne:

def translate_seq(ARNm):

### puis coller ligne 2 le contenu du fichier "code_gen"
### "code_gen" contient le programme de conversion

### Ajouter ceci à la fin, pour afficher le résultat:

print ("ac.am. à 3 lettres: "+translate_seq(ARNm))
```