1) Ecrire un algorithme qui demande à l'utilisateur un nombre compris entre 1 et 3 jusqu'à ce que la réponse convienne.

```
Variable N en Entier
Debut
N ← 0
Ecrire "Entrez un nombre entre 1 et 3"
TantQue N < 1 ou N > 3
  Lire N
    Si N < 1 ou N > 3 Alors
      Ecrire "Saisie erronée. Recommencez"
    FinSi
  FinTantQue
Fin
```

2) Ecrire un algorithme qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : « Plus petit ! », et inversement, « Plus grand ! » si le nombre est inférieur à 10.

```
Variable N en Entier
Debut
N ← 0
Ecrire "Entrez un nombre entre 10 et 20"
TantQue N < 10 ou N > 20
  Lire N
  Si N < 10 Alors
    Ecrire "Plus grand !"
  SinonSi N > 20 Alors
    Ecrire "Plus petit !"
  FinSi
FinTantQue
Fin
```

3) Ecrire un algorithme qui demande un nombre de départ, et qui ensuite affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

```
Variables N, i en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
Stop ← N+10
Ecrire "Les 10 nombres suivants sont : "
TantQue N < Stop
   N ← N+1
   Ecrire N
FinTantQue
Fin
```

Ou bien

```
Variables N, i en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
i ← 0
Ecrire "Les 10 nombres suivants sont : "
TantQue i < 10
   i ← i + 1
   Ecrire N + i
FinTantQue
Fin
```

4) Ecrire un algorithme qui demande un nombre de départ, et qui ensuite écrit la table de multiplication de ce nombre, présentée comme suit (cas où l'utilisateur entre le nombre 7) :

```
Table de 7 :
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
…
7 x 10 = 70
```

```
Variables N, i en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
Ecrire "La table de multiplication de ce nombre est : "
Pour i ← 1 à 10
  Ecrire N, " x ", i, " = ", n*i
i Suivant
Fin
```

5) Ecrire un algorithme qui demande un nombre de départ, et qui calcule sa factorielle.

NB : la factorielle de 8, notée 8 !, vaut
1 x 2 x 3 x 4 x 5 x 6 x 7 x 8

```
Variables N, i, F en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
F ← 1
Pour i ← 2 à N
  F ← F * i
i Suivant
Ecrire "La factorielle est : ", F
Fin
```

6) Écrire un algorithme qui permette de connaître ses chances de gagner au tiercé, quarté, quinté et autres impôts volontaires.

On demande à l'utilisateur le nombre de chevaux partants, et le nombre de chevaux joués. Les deux messages affichés devront être :

```
Dans l'ordre : une chance sur X de gagner
Dans le désordre : une chance sur Y de gagner
```

X et Y nous sont donnés par la formule suivante, si n est le nombre de chevaux partants et p le nombre de chevaux joués (on rappelle que le signe ! signifie "factorielle", comme dans l'exercice 5.6 ci-dessus) :

```
X = n ! / (n - p) !
Y = n ! / (p ! * (n - p) !)
```

NB : cet algorithme peut être écrit d'une manière simple, mais relativement peu performante. Ses performances peuvent être singulièrement augmentées par une petite astuce. Vous commencerez par écrire la manière la plus simple, puis vous identifierez le problème, et écrirez une deuxième version permettant de le résoudre.

Spontanément, on est tenté d'écrire l'algorithme suivant :

```
Variables N, P, i, Numé, Déno1, Déno2 en Entier
Debut Ecrire "Entrez le nombre de chevaux partants : "
Lire N
Ecrire "Entrez le nombre de chevaux joués : "
Lire P
Numé ← 1
Pour i ← 2 à N
  Numé ← Numé * i
i Suivant
Déno1 ← 1
Pour i ← 2 à N-P
  Déno1 ← Déno1 * i
i Suivant
Déno2 ← 1
Pour i ← 2 à P
  Déno2 ← Déno2 * i
i Suivant
Ecrire "Dans l'ordre, une chance sur ", Numé / Déno1
Ecrire "Dans le désordre, une sur ", Numé / (Déno1 * Déno2)
Fin
```

Cette version, formellement juste, comporte tout de même deux faiblesses.

La première, et la plus grave, concerne la manière dont elle calcule le résultat final. Celui-ci est le quotient d'un nombre par un autre ; or, ces nombres auront rapidement tendance à être très grands. En calculant, comme on le fait ici, d'abord le numérateur, puis ensuite le dénominateur, on prend le risque de demander à la machine de stocker des nombres trop grands pour qu'elle soit capable de les coder

(cf. le préambule). C'est d'autant plus bête que rien ne nous oblige à procéder ainsi : on n'est pas obligé de passer par la division de deux très grands nombres pour obtenir le résultat voulu.

La deuxième remarque est qu'on a programmé ici trois boucles successives. Or, en y regardant bien, on peut voir qu'après simplification de la formule, ces trois boucles comportent le même nombre de tours ! (si vous ne me croyez pas, écrivez un exemple de calcul et biffez les nombres identiques au numérateur et au dénominateur). Ce triple calcul (ces trois boucles) peut donc être ramené(es) à un(e) seul(e). Et voilà le travail, qui est non seulement bien plus court, mais aussi plus performant :

```
Variables N, P, i, A, B en Numérique
Debut
Ecrire "Entrez le nombre de chevaux partants : "
Lire N
Ecrire "Entrez le nombre de chevaux joués : "
Lire P
A ← 1
B ← 1
Pour i ← 1 à P
  A ← A * (i + N - P)
  B ← B * i
i Suivant
Ecrire "Dans l'ordre, une chance sur ", A
Ecrire "Dans le désordre, une chance sur ", A / B
Fin
```