TD_5 – Booléens et autres histoires

1 Cours – type Booléen

Une variable de type *booléen* (`boolean`) peut prendre deux valeurs : `true` (vrai) ou `false` (faux).
Le type booléen est notamment le type des opérateurs de comparaison (`=`, `<`, `>`, ...). Ainsi :
- `5 = 3 + 2` vaut `true` ;
- `7 > 9` vaut `false` ;
- `'A' <> 'B'` vaut `true`.

2 Crible d'Eratosthène

2.1 Principe

Ce crible permet de déterminer les nombres premiers inférieurs à un entier n préalablement fixé.

On commence par écrire les entiers de 1 à n dans un tableau. Il va s'agir de barrer les nombres qui ne sont pas premiers. On répète la procédure suivante pour tous les nombres p non barrés, en commençant par 2 :

1. Afficher p car on sait que c'est un nombre premier ;
2. Barrer tous les multiples de p sauf p.

On remarquera qu'on a choisi d'afficher les nombres premiers trouvés au fur et à mesure. Alternativement, on pourrait ne rien afficher pendant la phase de détermination, et faire l'affichage de tous les nombres non barrés en une seule fois à la fin.

2.2 Programmation

Implémenter cet algorithme en Turbo Pascal. On utilisera un tableau de booléens.

3 Nombres parfaits

Un nombre entier $n \geq 2$ est dit *parfait* s'il est égal à la somme de tous ses diviseurs stricts, 1 compris.

Ecrire une fonction qui teste si son argument est un nombre parfait. Le prototype de cette fonction sera :

```
function parfait(n: integer): boolean;
```

4 Suite de Syracuse

On appelle *suite de Syracuse* la suite u_n définie par un u_0 choisi arbitrairement et par la relation de récurrence suivante :

$$u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n \text{ est pair,} \\ 3u_n + 1 & \text{sinon.} \end{cases}$$

La *conjecture de Syracuse* (non démontrée) prévoit que l'on atteint *forcément* la valeur 1 après un certain nombre d'étapes : pour tout u_0, il existe un rang n_0 tel que $u_{n_0} = 1$.

Ecrire un programme qui demande u_0 à l'utilisateur, puis calcule les termes de la suite.