TP7 : `input`, `find` et nombres complexes

1 Quelques nouvelles commandes

1.1 `input`

La commande `input` permet à l'utilisateur de rentrer des valeurs en argument dans un programme sans avoir à définir une fonction.

Par exemple, `x = input("rentrer x")` affiche à l'écran "rentrer x" et laisse l'utilisateur rentrer une valeur, qui est affectée à la variable x.

Exercice 1. Écrire un programme Scilab qui demande à l'utilisateur son âge et affecte la réponse à la variable a.

1.2 Conditions sur des matrices

La commande `find` prend en argument une condition portant sur une matrice. Elle renvoie l'ensemble des indices des cases qui satisfont la condition.

Exercice 2. Que font les programmes suivants ?

1. `A = grand(1,9,"bin",4,1/2) ;`
2. `A==3`
3. `sum(A==3)`
4. `find(A==3)`
5. `B = A(find(A>1))`

Exercice 3. Écrire un programme Scilab qui simule 10 valeurs d'une variable aléatoire $X \hookrightarrow \mathcal{U}(\llbracket 0, 15 \rrbracket)$ et qui renvoie la somme des valeurs supérieures à 6.

2 Nombres complexes

Scilab permet de manipuler les nombres complexes, qui sont stockés sous forme de paires de nombres. On peut les définir en utilisant la notation algébrique : le nombre complexe $z = a + ib$ avec a et b deux réels, s'écrit :

```
z = a + % i * b
```

L'addition et la multiplication sont notées : $+$ et $*$, comme pour les réels.
Les fonctions suivantes sont prédéfinies (mais ne sont pas au programme) : `abs` (module), `real` (partie réelle), `imag` (partie imaginaire).

Exercice 4. En s'aidant des commandes précédentes,

1. Mettre $(1+i)^5$ sous forme algébrique,
2. Simplifier $\dfrac{3+4i}{2-i}$ en le mettant sous forme algébrique, puis calculer son module.

Exercice 5. Écrire un programme Scilab qui prend en argument un nombre complexe, renvoie "oui" si il est imaginaire pur, et "non" sinon.