Initiation à l'algorithmique

TD 3

Exercice 3.1

On se propose de représenter sous forme algébrique un nombre complexe z en utilisant le type *complexe* :

```
complexe = structure
             a: réel;
             b: réel;
           finStructure
```

avec a partie réelle d'un nombre complexe, b partie imaginaire. On a donc $z = a + ib$.

1. Écrivez les fonctions d'**addition** de deux nombres complexes et de **multiplication** d'un nombre complexe par un réel.
2. Écrivez les fonctions de calcul du **module** d'un nombre complexe z ($|z| = \sqrt{a^2 + b^2}$) et de l'**argument** d'un nombre complexe z ($arg(z) = \arccos \frac{a}{|z|}$).

Exercice 3.2

Nous voulons maintenant représenter un nombre complexe sous sa forme exponentielle $z = re^{i\theta}$, avec r module de z et θ argument. Proposez un nouveau type *complexeExp* permettant cette représentation.

1. Écrivez une fonction de conversion d'un nombre complexe de sa forme algébrique vers sa forme exponentielle.
2. Écrivez une fonction de multiplication d'un nombre complexe représenté sous forme algébrique avec un autre nombre complexe représenté sous forme exponentielle. Le résultat sera donné sous forme exponentielle.

Exercice 3.3

On cherche à représenter un point dans l'espace. Proposez une structure de données *point* adéquate.

1. Écrivez une fonction échangeant la valeur de deux points passés en paramètres.
2. Écrivez une fonction calculant la distance entre deux points.
3. Écrivez une fonction prenant trois points a, b, c en paramètres et stockant dans c le point tel que le vecteur $\overrightarrow{oc}$ soit égal au vecteur $\overrightarrow{ab}$.
4. Écrivez une fonction prenant trois points a, b, c en paramètres et stockant dans c le point tel que le vecteur $\overrightarrow{oc}$ soit égal au vecteur $\overrightarrow{ab}$ si a est plus proche de l'origine que b, $\overrightarrow{ba}$ sinon.

Exercice 3.4

On s'intéresse ici à des segments de droites en 3D. Ils ne prennent pas en compte la direction, et sont définis par leur deux points extrémités. Proposez un type pour représenter un segment de droite.

1. Écrivez une fonction prenant en paramètre deux segments de droite qui teste si les deux segments sont égaux ou non.
2. Écrivez une fonction renvoyant le segment de droite le plus court parmi les deux segments en paramètres.

Exercice 3.5

Soit la fonction suivante :

```
fonction deuxPiecesFace (val n:entier):booléen;
  var i: entier;
  var f, g: lancé;
  début
    pour i allant de 1 à n faire
      f = résultat_lancer_pièce();
      g = résultat_lancer_pièce();
      si (f == face et g == face) alors
        retourner(VRAI)
      finsi
    finpour
    retourner(FAUX)
  fin
finFonction
```

Déterminez sa complexité dans le pire des cas et sa complexité dans le meilleur des cas.