Objets 3D et impression

Sujet n°9 — Corrigé

Question 1 *Distance entre deux sommets (fichier `Sommet.py`)*

```
    def distance(self, s):
        return math.sqrt((s.x - self.x)**2 + (s.y - self.y)**2 + (s.z - self.z)**2)
```

`Sommet(0, 0, 0).distance(Sommet(3, 4, 0))` vaut 5,0. (Le fichier fourni contient déjà une version de cette méthode, écrite avec `** 0.5`.)

Question 2 *Sommets adjacents (fichier `Objet3D.py`)*

Deux sommets sont adjacents s'ils sont consécutifs dans une face, en comptant le côté qui relie le dernier sommet au premier ; on compare les coordonnées, dans les deux sens.

```
    def sommets_adjacents(self, s1, s2):
        p1 = (s1.x, s1.y, s1.z)
        p2 = (s2.x, s2.y, s2.z)
        for face in self.faces:
            n = len(face.sommets)
            for k in range(n):
                a = face.sommets[k]
                b = face.sommets[(k + 1) % n]
                pa = (a.x, a.y, a.z)
                pb = (b.x, b.y, b.z)
                if (pa == p1 and pb == p2) or (pa == p2 and pb == p1):
                    return True
        return False
```

Cube fourni : arête la plus longue 3,0, volume englobant 27 mm^3.

Question 3 *Temps d'impression (fichier `Imprimante3D.py`)*

```
    def estimation_impression(self, objet):
        volume_impression = objet.volume_cube_englobant() * self.remplissage
        return volume_impression / self.vitesse_extrusion
```

Rhombicuboctaèdre : arête la plus longue $2\sqrt{2}$, volume englobant $\approx$ 22,63 mm^3 ; avec `Imprimante3D(20, 1.2)`, l'estimation vaut environ 377,1 s. Ces paramètres ne respectent pas la description (un taux de remplissage devrait être compris entre 0 et 1).

Question 4 *Un agrandissement sans effet*

Analyse : `ajouter_face` stocke dans chaque face les *objets* `Sommet` eux-mêmes. `transformer` crée de nouveaux sommets et les place dans `self.sommets`, mais les faces référencent toujours les anciens : l'affichage, qui parcourt les faces, ne change pas. **Correction** : modifier les sommets existants.

```
    def transformer(self, rapport):
        for sommet in self.sommets:
            sommet.x = sommet.x * rapport
            sommet.y = sommet.y * rapport
            sommet.z = sommet.z * rapport
```

Après `cube.transformer(2)`, l'arête la plus longue vaut 6,0.