Solveur

Variables :

```
jeu2 de type game
i de type entier
n de type pointeur d'entier
D de type direction
dir de type tableau de directions
```

Entrées :

```
jeu de type cgame
option de type option
nom_fichier de type mot
```

Code :

```
fonction solveur (jeu, option, nom_fichier) :
    jeu2 = copie de jeu
    retourner solveur_aux (jeu2, option, i, n, nom_fichier)
fin fonction

fonction solveur_aux (jeu2, option, i, n, nom_fichier) :
    si i == nb_cases (jeu2) alors
        si is_game_over (jeu2) alors (pas besoin car on fait en sorte d'arriver ici seulemer
            resultat (jeu2, option, n, nom_fichier)
        fin si
        retourner rien
    fin si
    si i-ème pièce de jeu2 appartient à {Leaf, Corner, Tee} alors
        dir = {N, S, E, W}
    sinon si i-ème pièce de jeu2 appartient à {Segment} alors
        dir = {N, E}
```

```
    sinon alors
        dir = {N}
    fin si
    pour D appartient à dir faire
        orienter la i-ème pièce de jeu2 dans la direction D
        si c'est une bonne direction* alors
            solveur_aux(jeu2, option, i+1, n, nom_fichier)
        fin si
    fin pour
    retourner rien
fin fonction
```

Bonne direction :

Si ce n'est pas wrapping :

1. Pièce vise une/des case(s) existante(s) (donc pas le bord)
2. Pièces voisines Ouest et Sud (si elles existent) doivent :
    - soit viser la pièce courante et à ce moment là la pièce courante doit aussi viser la voisine
    - soit ne pas viser la pièce courante et à ce moment là la pièce courante ne doit pas viser la voisine

Si c'est wrapping :

1. Pièces voisines Ouest et Sud (si elles existent) doivent faire la même chose que sans wrapping
2. Si on touche un bord Est ou Nord, alors il faut aussi que les pièces voisines correspondantes :
    - soit visent la pièce courante et alors il faut que celle-ci vise la voisine
    - soit ne visent pas la pièce courante et alors celle-ci ne doit pas viser la voisine