```
class Carte:
    """Initialise Couleur (entre 1 à 4), et Valeur (entre 1 à 13)"""
    def __init__(self, c, v):
        # ajout des clauses de gardes sur
        # les couleurs et les valeurs
        assert 1 <= c <= 4, 'erreur de couleur'
        assert 1 <= v <= 13, 'erreur de valeur'
        self.Couleur = c
        self.Valeur = v

    """Renvoie le nom de la Carte As, 2, ... 10,
       Valet, Dame, Roi"""
    def getNom(self):
        if ( self.Valeur > 1 and self.Valeur < 11):
            return str( self.Valeur)
        elif self.Valeur == 11:
            return "Valet"
        elif self.Valeur == 12:
            return "Dame"
        elif self.Valeur == 13:
            return "Roi"
        else:
            return "As"

    """Renvoie la couleur de la Carte (parmi pique, coeur, carreau, trefle"""
    def getCouleur(self):
        return ['pique', 'coeur', 'carreau', 'trefle' ][self.Couleur - 1]

class PaquetDeCarte:
    def __init__(self):
        self.contenu = []

    """Remplit le paquet de cartes"""
    def remplir(self):
        # c'est un tableau défini par compréhension
        self.contenu = [Carte(couleur, valeur) for couleur in range(1, 5) for valeur
 in range(1, 14)]

    """Renvoie la Carte qui se trouve à la position donnée"""
    def getCarteAt(self, pos):
        # clause de garde pour ne pas demander une
        # position plus grande que le tableau de cartes
        n = len(self.contenu)
        assert 0 <= pos < n, 'index out of range'

        return self.contenu[pos]


unPaquet = PaquetDeCarte()
unPaquet.remplir()
uneCarte = unPaquet.getCarteAt(20)
print(uneCarte.getNom(), "de", uneCarte.getCouleur())
assert str(uneCarte.getNom()) + " de " + str(uneCarte.getCouleur()) == "8 de coeur"
```