SYSTÈMES DE VOTE

1 Scrutin à un tour

1.
```python
def creer_dico_resultats(votes):
    D = {}
    for bulletin in votes:
        if bulletin in D:
            D[bulletin] += 1
        else:
            D[bulletin] = 1
    return D
```

2.
```python
def cle_de_valeur_max(D):
    max = 0
    for cle in D:
        valeur = D[cle]
        if valeur > max:
            max = valeur
            cle_max = cle
    return cle_max
```

3.
```python
def scrutin_un_tour(votes):
    D = creer_dico_resultats(votes)
    return cle_de_valeur_max(D)
```

2 Scrutin à deux tours

1. C'est Bob qui gagne au second tour contre Alice.
2.
```python
def candidats_second_tour(D):
    p = cle_de_valeur_max(D)
    max = 0
    for cle in D:
        if cle != p and D[cle] > max:
            max = valeur
            s = cle
    return p,s
```

3.
```python
def vote_second_tour(bulletin,p,s):
    for c in bulletin:
        if c==p or c==s:
            return c
```

4.
```python
def scrutin_deux_tours(votes):
    votes_premier_tour = [bulletin[0] for bulletin in votes]
    resultats_premier_tour = creer_dico_resultats(votes_premier_tour)
    p,s = candidats_second_tour(resultats_premier_tour)
    votes_second_tour = [vote_second_tour(bulletin,p,s) for bulletin in




    return scrutin_un_tour(votes_second_tour)
```

5. Pour le scrutin à un tour, en supposant que les listes de préférences soient celles données dans la partie 3, si le 4^e votant vote Bob plutôt qu'Alice, et le 6^e Bob plutôt que Charlie, alors c'est Bob qui est élu au scrutin à un tour, candidat préféré à Alice pour ces deux votants.
   Pour le scrutin à deux tours, si au premier tour les votants 1 et 3 votent Charlie plutôt qu'Alice, alors Bob perd les élections et Charlie gagne. Pour ces deux votants, c'est bien Charlie qu'ils préfèrent à Bob. Ces deux scrutins sont donc sujets au dilemme du vote utile.

3 Dictature aléatoire

1.
```python
from random import *

def dictature_aleatoire(T):
    n = len(T)
    k = randint(0,n-1)
    return T[k]
```

2. Non, on a toujours intérêt à voter pour son candidat préféré pour qu'en probabilité il ait plus de chance d'être choisi.

4 Scrutin par élimination

1. C'est Dave qui est élu.
2.
```python
def liste_sans(L,a):
    R=[]
```

```
    for x in l:
        if x!=a:
            R.append(x)
    return R
```

3.

```
def enlever_candidat(T,p):
    R=[]
    for L in T:
        L2 = liste_sans(L,p)
        R.append(L2)
    return R
```

4.

```
def cle_de_valeur_min(D):
    min = float('inf')
    for cle in D:
        if D[cle] < min or (D[cle] == min and cle < cle_min) :
            min = D[cle]
            cle_min = cle
    return cle_min
```

5.

```
def dernier(votes):
    votes_tour_courant = [bulletin[0] for bulletin in votes]
    resultats = creer_dico_resultats(votes_tour_courant)
    for bulletin in votes:
        for c in bulletin:
            if c not in resultats:
                resultats[c] = 0
    return cle_de_valeur_min(resultats)
```

6.

```
def scrutin_elimination(votes):
    R=votes
    while len(R[0]) > 1:
        p = dernier(R)
        R = enlever_candidat(R,p)
    return R[0][0]
```

7. Si le votant 1 vote Charlie plutôt que Alice et le votant 2 Charlie plutôt que Bob, alors c'est Charlie qui est élu, et il est placé avant Dave dans leur liste de préférence. Ce système de scrutin est donc sujet au vote utile.

5 Scrutin de Condorcet

1. Oui, Charlie est le vainqueur de Condorcet.
2. Par exemple : Alice > Bob > Charlie, Bob > Charlie > Alice, Charlie > Alice > Bob.
3.

```
def duel(votes,c1,c2):
    votes_duel = [vote_second_tour(bulletin,c1,c2) for bulletin in votes]
    return scrutin_un_tour(votes_duel)

def liste_candidats(votes):
    L = []
    for bulletin in votes:
        for candidat in bulletin:
            if candidat not in L:
                L.append(candidat)
    return L

def est_vainqueur_condorcet(votes,L,c):
    for x in L:
        if x!=c and duel(votes,x,c)==x:
            return False
    return True


def scrutin_condorcet(votes):
    L = liste_candidats(votes)
    for c in L:
        if est_vainqueur_condorcet(votes,L,c):
            return c
    return None
```

Fin