```python
# TPNIM.py

import random as rd


def voisins(char:str)->list :
    """
    Permet de renvoyer l'étiquetage des sommets atteints par
char
    ================================================
    entrée :
    char : une chaine de caractère contenant l'étiquetage d'un
sommet
    --------------------------------
    sortie :
    L : une liste des étiquetage des sommets atteints par char
    """
    a,b=char.split(';')
    a,b=int(a),int(b)
    L=[]
    for k in range(max(1,a-3),a):
        L=[str(k)+';'+str(1-b)]+L # sinon L.append(str(k)
+';'+str(1-b)) mais
    return L                      # ils sont rajoutés dans le
même sens que l'exemple ainsi.


def
const_voisins(S:list,A:list[list],char:str,Lvoisins:list[str])-
>tuple[list,list[list]]:
    """
    Permet de rajouter le voisinage de char dans le couple
(A,S)
    ================================================
    entrée :
    S : dictionnaire des sommets
    A : liste des arêtes
    char : étiquetage du sommet dont on rajoute son voisinage
    Lvoisins=liste des étiquetages des sommets atteignables
par char
    --------------------------------
    sortie :
    S : dictionnaire mis à jour
    A : listes des arêtes mise à jour
    """
    for k in Lvoisins:
        if k not in S:
            S[k]=len(S)
        A.append([S[char],S[k]])
    return S,A


def graphe(N:int)->tuple[list,list[list]]:
    """
```

```
046|     Permet de construire le couple (S,A) pour une partir de
Nim avec N jetons
047|     ================================================
048|     entrée :
049|     N ; entier, nombre de jetons de la partie
050|     --------------------------------
051|     sortie :
052|     S : dictionnaire des sommets
053|     A : liste des arêtes
054|     """
055|     t=0
056|     M=[str(N)+';0']
057|     char=str(N)+';0'
058|     S={}
059|     S[char]=0
060|     A=[]
061|     Lvoisins=voisins(char)
062|     S,A=const_voisins(S,A,char,Lvoisins)
063|     for k in Lvoisins:
064|         if k not in M:
065|             M.append(k)
066|     t=t+1
067|     while t<len(S):
068|         char=M[t]
069|         Lvoisins=voisins(char)
070|         S,A=const_voisins(S,A,char,Lvoisins)
071|         for k in Lvoisins:
072|             if k not in M:
073|                 M.append(k)
074|         t=t+1
075|     return S,A
076|
077| def joueurA(char:str)->str:
078|     """
079|     Permet de simuler un joueur jouant avec la stratégie
énoncée dans la question 4
080|     ================================================
081|     entrée :
082|     char : chaine de caractères, correspond à l'étiquetage
d'où en est la partie
083|     --------------------------------
084|     sortie :
085|     une chaine de caractère correspondant à l'étiquetage d'où
en est la partie après que le joueur est joué sa stratégie
086|     """
087|     a,b=char.split(';')
088|     a,b=int(a),int(b)
089|     if a%4==1:
090|         a=rd.choice([a-k for k in range(1,4) if a-k>=1])
091|     else :
092|         a=a-a%4+1
093|         return str(a)+";"+str(1-b)
094|     return str(a)+";"+str(1-b)
095|
096|
```

```
097| def joueurB(char:str)->str:
098|     """
099|     Permet de simuler un joueur jouant avec la stratégie
énoncée dans la question 5
100|     ================================================
101|     entrée :
102|     char : chaine de caractères, correspond à l'étiquetage
d'où en est la partie
103|     --------------------------------
104|     sortie :
105|     une chaine de caractère correspondant à l'étiquetage d'où
en est la partie après que le joueur est joué sa stratégie
106|     """
107|     a,b=char.split(';')
108|     a,b=int(a),int(b)
109|     if a in [2,3,4]:
110|         return "1;"+str(1-b)
111|     a=rd.choice([a-k for k in range(1,4) if a-k>=1])
112|     return str(a)+";"+str(1-b)
113|
114|
115| def partie(Ljoueur:list,N:int)->int:
116|     """
117|     Simule une partie du jeu de Nim commençant avec N jetons
118|     ================================================
119|     entrée :
120|     Ljoueur : liste de fonctions, L[k] correspond à la
fonction donnant la stratégie du joueur k
121|     N : entier, nombre de jetons
122|     --------------------------------
123|     sortie :
124|     entier, numéro du vainqueur de la partie
125|     """
126|     char=str(N)+";"+str(0)
127|     while char not in ["1;0","1;1"]:
128|         a,b=char.split(';')
129|         a,b=int(a),int(b)
130|         char=Ljoueur[b](char)
131|     return 1-int(char.split(';')[1])
132|
133|
134| ##
135| L=[0,0]
136| for k in range(100000):
137|     L[partie([joueurB,joueurA],20)]+=1
138| print(L[1]/100000)
139|
140|
```