Numéro d'inscription

Numéro de table

Né(e) le

Nom :

Prénom :

Emplacement QR Code

Filière : TSI

Session : 2024

Épreuve de : INFORMATIQUE

Consignes

- *Remplir soigneusement l'en-tête de chaque feuille avant de commencer à composer*
- *Rédiger avec un stylo non effaçable bleu ou noir*
- *Ne rien écrire dans les marges (gauche et droite)*
- *Numéroter chaque page (cadre en bas à droite)*
- *Placer les feuilles A3 ouvertes, dans le même sens et dans l'ordre*

Q21. Fonction `degre_satur`

```
def degre_satur(LA,s,C):
    "renvoie le degré de saturation du sommet s"
    deja_vue = d_y  # dictionnaire pour se souvenir
                    des couleurs déjà rencontrée
    res=0
    for j in LA[s]:
        if C[j] != -1 and C[j] not in deja_vue:
            res = res+1
            deja_vue[C[j]] = True
    return res
```

Q22. Fonction `liste_satur`

```
def liste_satur(LA,C):
    "renvoie la liste des sommets non colorés de degré de saturation max"
    lst_s, sat_max = [], 0
    for s in LA:
        sat = degre_satur(LA,s,C)
        if C[s]==-1 and sat > sat_max:
            sat_max = sat
            lst_s = [s]
        elif sat = sat_max and C[s]==-1
            lst_s.append(s)

    return lst_s
```

NE RIEN ÉCRIRE DANS CE CADRE

Q23. Fonction `pas_fini`

```
def pas_fini(C):
    """renvoie True si la coloration n'est pas finie,
    False sinon"""
    return -1 in C
```

Q24. Compléter la fonction selon les instructions de l'énoncé

```
def colorer4(LA):
    n = len(LA)    # nombre de sommets du graphe
    D = degre(LA)  # liste des degrés des sommets du graphe
    C = [-1]*n     # initialisation de la liste des couleurs
    while pas_fini(C) :
        # liste des sommets non colorés de degré de saturation maximal
        Ls = liste_satur(LA,C)
        smax = Ls[0]
        # en cas d'égalité, recherche du sommet de degré maximal
        if len(Ls) > 1:
            d_max = 0
            for s in Ls:
                if D[s] > d_max:
                    d_max = D[s]
                    smax = s

        # coloration du sommet prioritaire
        colore_sommet(C,smax,LA)

    return C
```

Q25. Relations entre n_c et le nombre minimum de couleurs, entre n_c et le degré maximum.

Un sommet de degré d est entouré de d voisins, il faut donc d+1 couleurs pour les colorer.
Il faut aussi au moins autant de couleur que le cardinal de la plus grande.
Ainsi n_c est majoré par le degré maximum +1.

Q26. Fonction `est_clique`

```
def est_clique(LA,K):
    """renvoie True si K est une clique, False sinon"""
    for i in range(len(K)):
        for j in range(i, len(K)):
            if not voisins(K[i], K[j], LA):
                return False
    return True
```

Q27. Compléter la fonction selon les instructions de l'énoncé

```
from itertools import combinations
def minoration_nb_couleurs(LA):
    n = len(LA)                    # nombre de sommets du graphe
    S = [ k for k in range(n) ] # liste des sommets du graphe
    i = n-1
    test = True
    while test:
        for K in combinations(S,i):
            if est_clique(LA,K):
                test = False

        i=i-1
    return len(K)
```