**TP01 – SAE 2 – Complexité et heuristiques**

**Question 1 : Redonnez la définition de complexité d’un algorithme (dans le pire des cas).**

Estimation du nombre d’opérations élémentaires qui sont effectuées dans le pire des cas en fonction la taille de l’entrée (N) pour résoudre un problème.

**Question 2 : En vous aidant d’Internet, donnez une définition simple de la notion de complexité d’un problème**

C’est la complexité minimale de la complexité des algos qui résolvent le problème.

**Question 3 : En vous aidant d’Internet, donnez une définition simple de la notion de classe de complexité et en particulier des classes T, P et NPc.**

Une classe de complexité permet de différentier les niveaux de difficultés des problèmes en les classant par niveau : T, P, NPc.

T : problème solvable en temps constant quelque soit la taille de l’entrée

P : problème solvable en un temps polynomial

NPc : problème non solvable en un temps polynomial permettant de résoudre rapidement n’importe quelle instance du problème.

**Question 4 : Cherchez des exemples de problèmes pour chacune de ces trois classes de complexités.**

T : Renvoyer 1ere valeur d’un tableau

P : tri de tableau / plus court chemin …

NP : voyageur commerce / coloration …

**Question 5 : Quel est l’intérêt de connaître la complexité d’un problème avant de coder un programme pour le résoudre ? Expliquez le comportement à adopter en fonction de la classe de complexité du problème.**

Permet de savoir s’il est possible de résoudre le problème

NPc : impossible de résoudre

**Question 6 : Donnez une définition simple d’une heuristique.**

Un algorithme qui produit rapidement une solution réalisable mais pas forcément optimale.

**Question 7 : Donnez le nom et le principe de l’heuristique de coloration de graphe que vous connaissez.**

Coloration : DSATUR

……..

**Question 8 : Selon vous, qu’est-ce qu’une bonne heuristique et qu’est-ce qu’une mauvaise heuristique ?**

Bonne heuristique : elle donne une réponse souvent de bonne qualité , rapide et pas gourmande en mémoire

**Question 9 : Expliquez brièvement le principe de chacune de ces trois types d’heuristiques.**

Durant votre formation, vous allez être confronté principalement à trois types d’heuristiques :

**• Des heuristiques gloutonnes**

Cela consiste à toujours prendre la solution optinmale à chaque étape

On construit la solution étape par étape et à chaque étape on fait le meilleur choix local sans garantie au niveau globale. On ne revient jamais sur les choix passés. Signé l’épée de sasuke

* DSATUR

**• Des heuristiques de recherche locale**

A partir d’une solution de base , on fait des petites modifications pour trouver 1 meilleure solution dans le voisinage

**• Des heuristiques génétiques**

Inspiré des principes de solution naturelle.

Faire jusqu’à la fin du monde

A partir d’une population de solutions générées aléatoirement.   
Croisement des parents pour produire les enfants.  
Mutation de quelques individus

Sélection de la génération suivants

**Question 10 : Pour chacun de ces types d’heuristique, imaginez plusieurs exemples pour résoudre le problème de coloration d’un graphe.**

**Locale :**

* REQI simulé ou chocolat bruyant

**Génétique :**

Faire plein de coloration, reproduire les meilleurs entre eux, à l’infini

* Problème des fourmis

**Question 11 : Choisissez avec votre enseignant un problème de la classe NPc (autre que le voyageur de commerce et la coloration) et adaptez vos différentes idées d’heuristiques à ce nouveau problème.**

Problème du sac à dos :

**Glouton :**

* Objet par valeur décroissant
* Objet par ratio valeur/poids décroissant
* Pas optimale car pas forcément tomber juste en poids sur le sac

**Recherche locale :**

* Sortir un élément du sac et essayer de le remplacer si meilleure valeur

**Algo génétique :**

* On remplit plein de sac, on les reproduits et on prend la meilleure
* Peu prendre beaucoup de temps car solution au hasard s

Question 12 : Proposez des idées d’heuristiques pour le problème du voyageur de commerce

* Depuis le point de départ, prendre le point le plus près d’où on est jusqu’à les avoir tous (plus proche voisin)