## Explique ce qu’est un algorithme et explique les 3 preuves qui le constitue:

Un algorithme est une fonction qui permet de résoudre un problème de façon systématique.

Ses trois preuves sont :

1. La terminaison
2. La complétude
3. La correction

Voici quelques exemples : un GPS, une recette de cuisine,…

## Explique ce que sont les différents arbres (binaire, par extension) ainsi que les différentes méthodes de parcours de ce genre de structure.

Un arbre binaire est défini par ses nœuds, sa hauteur ainsi que sa profondeur.

La profondeur étant la distance entre un nœud et la racine.

Tandis que la hauteur correspond à la profondeur maximum de l’arbre.

Un noeud d’un arbre binaire ne peut avoir que 1 ou 2 enfant(s)

Il y a 4 méthodes de parcours de ces arbres:

1. Préfixe => Racine puis de gauche à droite
2. PostFixe => De Gauche à droite puis la racine
3. Infixe => La gauche jusqu'à la racine ensuite la droite
4. En Largeur => Étage par étage de haut en bas

## Définit ce que sont les arbres binaires de recherche, puis explique comment se déroule la recherche, l’insertion et la suppression d’un nœud dans ce type de structure.

C'est un arbre binaire dans lequel chaque nœud possède une clé (valeur) de tel sorte que chaque nœud du sous-arbre de gauche ait une clé plus petite ou égale au nœud considéré, et inversement pour le sous-arbre droit

Recherche :

1. Si l’arbre est vide retourner “noeud non trouvé”
2. Si la valeur recherchée est égale à la valeur du noeud courant, retourner le noeud courant
3. Si la valeur recherchée est inférieure à la valeure du noeud courant, rechercher dans le sous-arbre gauche
4. Sinon, recherche dans le sous-arbre droit

Insertion :

1. On commence par une recherche fictive du noeud à insérer
2. lorsqu’il arrive à une feuille :

Ajouter comme fils en respectant les règles de l’arbre

Si la valeur voulue est plus grande, aller à droite si plus petite, à gauche

Suppression :

Recherche du noeud en question:

1. Suppression d’une feuille
2. Suppression avec un enfant
3. Suppression avec les 2 enfants

## Explique l’algorithme de tri par tas et les règles à respecter.

Les Règles à respecter :

1. S’assurer qu’on respecte la propriété "Max-Heape" (la clé du nœud parent est plus grande que celle du nœud enfant)
2. Echanger la racine avec le dernier élément – **Swap**
3. Réduire la taille du tas de 1 – **Remove**
4. Repositionner l’élément le plus grand à la racine – **Heapify**
5. Recommencer au point 2

## Explique ce qu’est le hachage et les propriétés à respecter. Explique les concepts de collision et de salage.

Une fonction de hachage transforme une chaine de caractère en en une valeur fixe respectant les propriétés suivantes :

Propriétés :

- Longueur de signature identique

- Unidirectionnel

- Impossible à prédire

- 2 données différentes à 2 signatures différentes

Collisions : Lorsque deux entrées différentes donnent le même hash

Salage :

- Ajouter un élément de données pour modifier la signature

- Augmente la sécurité face aux attaques

- Évite les doublons

## Définit et explique ce qu’est une table de hachage. Quels sont les soucis principaux de ce genre de structure et comment les résoudre.

Structure de données utilisant une fonction de hachage pour stocker et récupérer facilement des éléments

Problèmes:

- Recherche par clé simple : Hash de la clé à index dans le tableau

- Collisions possibles (problème)

- Gestion de l’espace

Solutions:

- Par chaînage: Chaque maillon possède une clé dans une liste chaînée

-Par adressage ouvert: Table de M structure, si une collision apparaît, on recherche une case vide par une fonction de sondage

## Définit ce qu’est un graphe. Cite les opérations de base de cette structure de données et explique les différentes représentations possibles.

Un graphe est un couple G = (V, E) où :

- V est un ensemble de sommets (nœuds, points, vertex)

- E est un ensemble d’arêtes (liens, lignes) qui sont des paires de sommets

- Peut-être (Non-)connexe (formant un objet d’un seul tenant), (non-)orientés, etc

Opérations de base :

- Adjacents(G, x, y)

- Voisins(G, x)

- Ajouter\_Sommet(G, x)

- Supprimer\_Sommet(G, x)

- Ajouter\_Arete(G, x, y)

- Supprimer\_Arete(G, x, y)

- Retourner\_Valeur(G, x)

- Fixer\_Valeur(G, x, v)

Représentation des graphes :

• Listes d’adjacence: Chaque sommet est un objet qui comprend une liste des sommets adjacents

• Matrice d’adjacence: Matrice carrée où les lignes représentent les sommets de départ et les colonnes les sommets d’arrivée

• Matrice d’incidence: Matrice où les lignes représentent les sommets et les colonnes les arêtes

## Qu’est ce que le problème de « Shortest path » ? Explique comment fonctionne l’algorithme de Dijkstra.

Shortest path: Trouver le chemin le plus court entre deux sommets d’un graph en respectant différents critères : la vitesse, la distance et le coût.

Fonctionnement :

• Construction d’un sous-graphe :

- Distance de chaque sommet avec celui de départ = ∞

- Choix du sommet à distance minimale hors du sous-graphe

- Mise-à-jour des distances des sommets voisins

- Plus de sommets / Sélection du sommet d’arrivée

## Qu’est ce que le problème de Spanning tree ? Explique comment fonctionne l’algorithme de Kruskal.

Spanning Tree : Algorithme de recherche d'arbre recouvrant de poids minimum

Construction d’une liste avec les arêtes par poids croissants : - Sélection des arêtes par ordre croissant en vérifiant qu’aucun cycle n’est créé

## Définit ce qu’est la compression de données et explique les différentes méthodes qui permettent de le faire.

Processus permettant de réduire la taille des données pour économiser de l’espace de stockage et améliorer les performances de transmission

Avec pertes: Utilisée pour les données sonores ou visuelles où la perte d’informations est imperceptible pour l’œil humain

Sans pertes: Aucune perte des données d’origine, réécriture de façon concise.

(Exemple LZW, ou Huffman)

## 

## Qu’est-ce que l’algorithme de LZW, comment fonctionne-t-il ?

• Très souvent utilisé avec les .GIF, les .PDF et les .TIFF

• Regroupement de symboles en chaînes

• Convertir chaînes en codes

• Codes prennent moins de places que les chaînes

Fonctionnement :

* Choix fréquents d’entrées possibles : 4096
* 0 à 255 bloqués pour les caractères ASCII
* Identification des séquences répétées (exemple BA= 256)

## Qu’est-ce que le codage de Huffman statique, comment fonctionne-t-il ?

* Attribution de code en fonction de la fréquence d’apparition
* Génération d’un arbre binaire avec branches étiquetées 0 ou 1
* L’arbre et le message compressé doivent être envoyés pour être décodé

## Qu’est ce que le chiffrement et quels sont ses objectifs principaux ? Explique brièvement les différentes techniques de base et donne des exemples.

C’est un procédé cryptographique qui consiste à rendre les données incompréhensibles à toute personne ne possédant pas la clé.

Divers moyen de procéder :

* Substitution (césar)
* Transposition
* Bloc (AES)
* Flux (RC4)

## Expliquer ce qu’est le chiffrement symétrique, quels sont ses problèmes courants et des solutions à ces problèmes. Expliquer brièvement les vulnérabilités de ce type de chiffrement.

Méthode chiffrement permettant de chiffrer et déchiffrer à l’aide d’une clé unique partagé entre les différents acteurs

Problèmes : échange, distribution ou rotation des clés.

Solutions :

* Chiffrement des clés
* PKI
* Gestion de clé centralisée

Vulnérabilités :

* Brut force
* Attaque au dictionnaire
* Attaque par analyse différentielle
* Attaque par bruteforce en temps réel

## Explique ce qu’est le chiffrement asymétrique. Explique comment fonctionne le RSA et cite quelques attaques réalisées sur les RSA.

Le chiffrement asymétrique utilise un couple de clé : une clé publique et une clé privée.

La clé privée sert à déchiffrer tandis que la clé publique sert à chiffrer.

RSA Fonctionnement :

Basé sur la difficulté de factoriser de grands nombre entiers en facteurs premiers

Clé publique (n,e) à chiffrement : messageclaire

Clé privée (d) à déchiffrement : messagechiffré

Difficile de calculer d sur base de (n, e) dans un temps raisonnable

Générer une clé RSA :

• Prendre deux nombre premiers

• Calculer le module de chiffrement

• Calculer l’indice d’Euler

• Choisir le nombre premier

• Calculer le nombre entier

Attaques :

• Force brute sur p et q

• Time attack

• Attaque de Wiener

## Qu’est ce qu’un tas ? Cite ses avantages.

Un tas c’est un arbre ordonné qui suit la loi du max heap, c’est à dire que l’élément qui à le plus de poids sera toujours au dessus.