Méthode 1 : construire un programme informatique

1/ Poser le problème

1. Lire le sujet, attentivement et entièrement ;
2. Un sujet est rarement bien formulé. Il faut en extraire ce qui important et le résumer en 1 ou 2 phrases interrogatives.
3. Un sujet est censé être difficile. Il ne faut pas chercher la solution dès la première lecture (néanmoins, si une idée de solution vous vient, il faut la noter sur un brouillon).
4. Poser le problème :
    a. Définir et paramétrer les données du problème :
        i. Lister les entrées (types, intervalle, quantité)
        ii. D'où viennent-elles ? (mémoire ou utilisateur)
        iii. Leur ordre est-il important ?
        iv. Paramétrer : associer une variable à ces données
    b. Décrire le résultat :
        i. Est-ce une interaction avec l'utilisateur ?
        ii. Est-ce une modification de la mémoire ?
        iii. Quelle(s) propriété(s) attend-on du résultat ? (à écrire mathématiquement après paramétrage)

2/ Recherche de solution

2.1/ *Cas trivial (évident)*

- La solution découle de la propriété du résultat à respecter.
    - Mettre en évidence des inconnues à calculer
    - Mettre en place les calculs pour déterminer ces inconnues

2.2/ *Cas non trivial*

1. Commencer par un exemple des données choisi arbitrairement
2. Résoudre entièrement cet exemple à la main
    a. Le plus clairement et proprement possible (en utilisant une écriture mathématique, un dessin, un tableau, etc.)
3. Mettre en évidence l'algorithme naïf, c'est-à-dire les opérations successives que fait votre cerveau pour mener à bien cette résolution. [attention, une opération de base à la fois]
4. Si l'algorithme naïf n'apparait pas, recommencer avec d'autres exemples [garder ces exemples pour la suite]
    a. des exemples plus simples ;
    b. des exemples plus compliqués, qui intègrent des points de vue différents ;
    c. des cas particuliers, à la limite de l'acceptable ;
    d. des cas inacceptables.
5. Une fois l'algorithme naïf trouvé, il faut l'améliorer :
    a. Repérer les opérations inutiles
    b. Regrouper les opérations
    c. Ordonner les opérations.
6. Enfin, écrire le programme.
7. Tester le bon résultat du programme sur les exemples utilisé ci-dessus.