

TP TEST EN BOÎTE NOIRE

Tous les livrables sont à retrouver sur le git:

<https://etulab.univ-amu.fr/m19025211/m-1-info-fsi-tp-template-mededji.git>

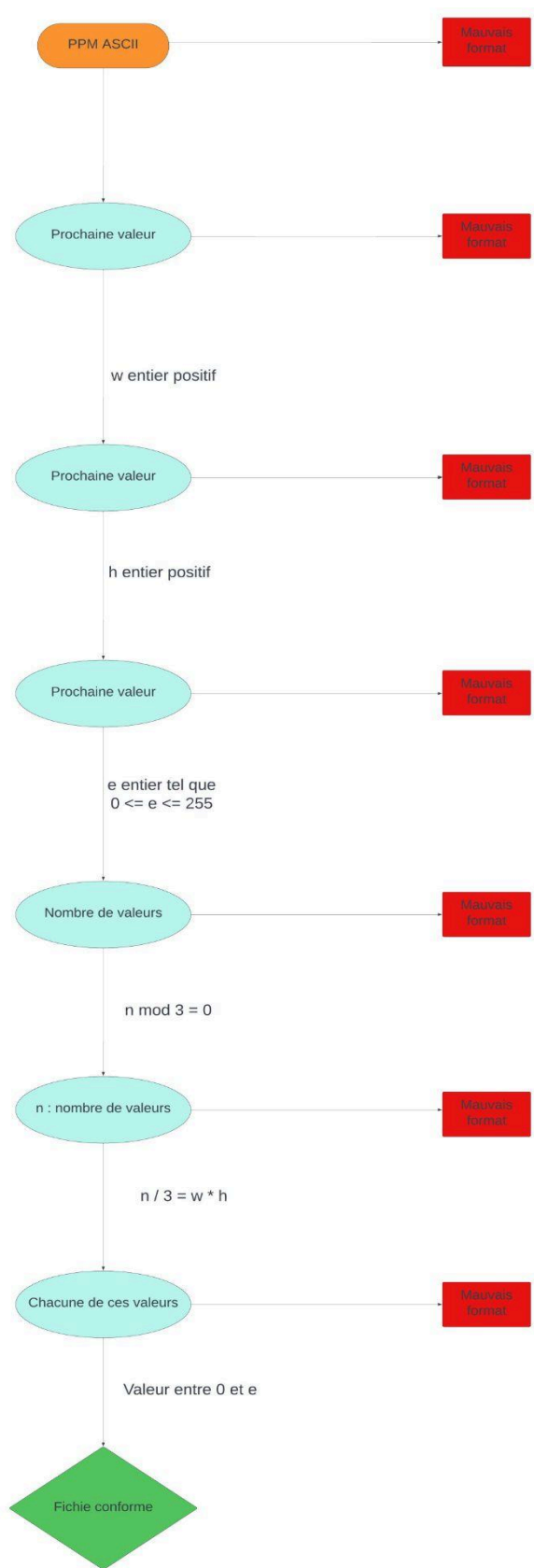
1. Arbre de décision représentant les cas de test



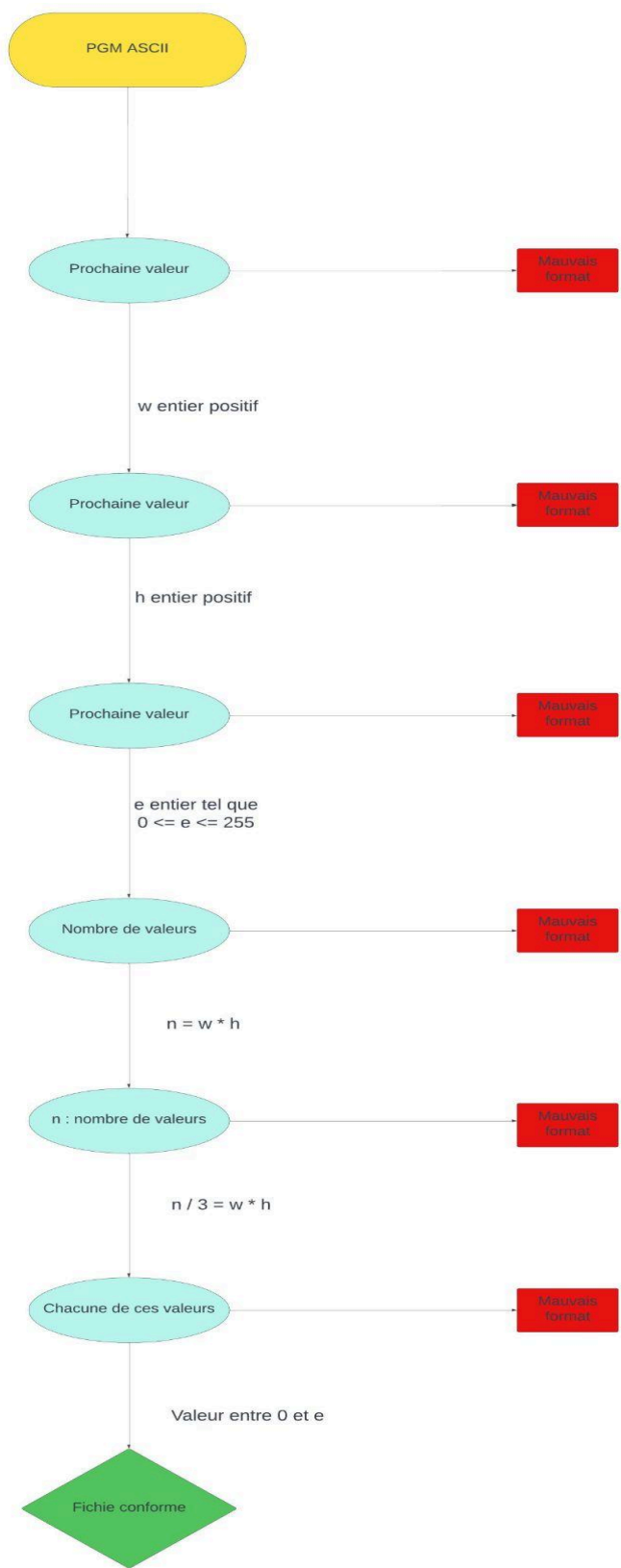
Le reste de l'arbre de décision va être découpé en 6 sous parties suivant les différents types d'images en raison de sa longueur.

A noter que, à chaque noeud de cet arbre (dès qu'on lit le fichier), on a une possible branche de test pour une ligne de commentaire débutant par #. Ces lignes ne sont pas prises en compte par l'application. On a donc pas besoin de l'inclure dans notre arbre de décision.

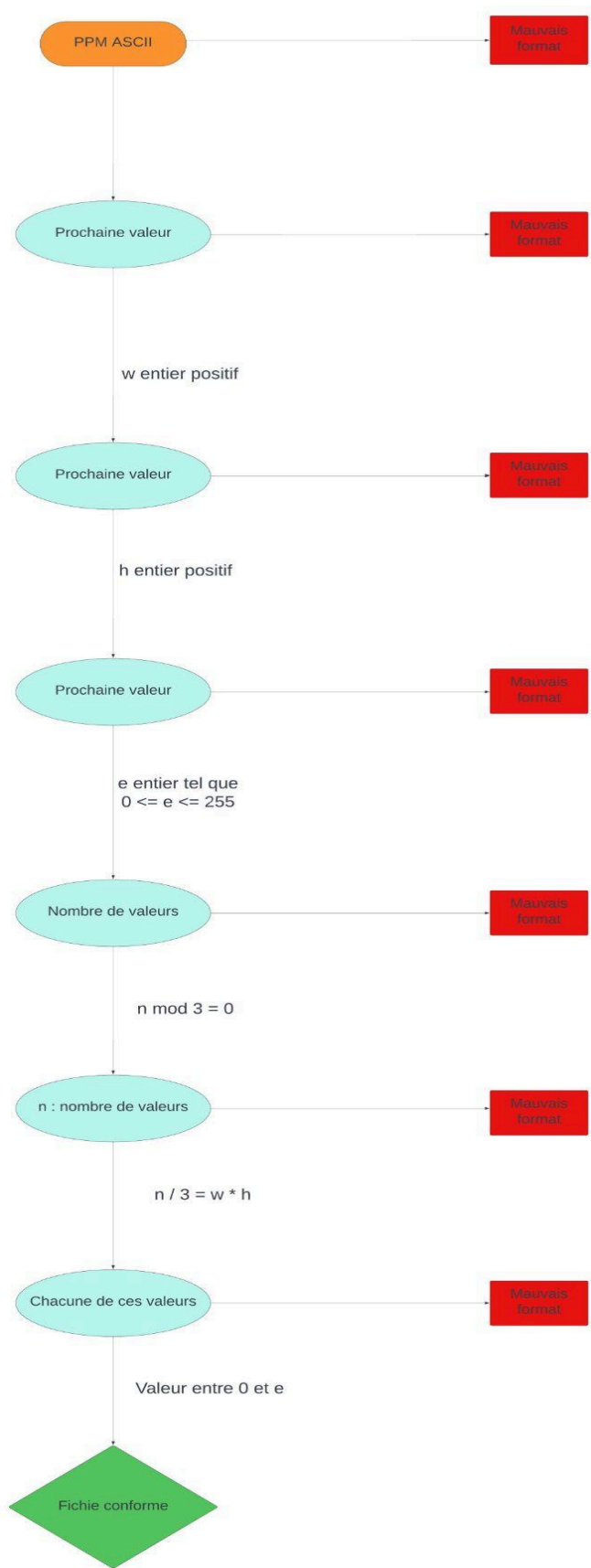
- PPM ASCII



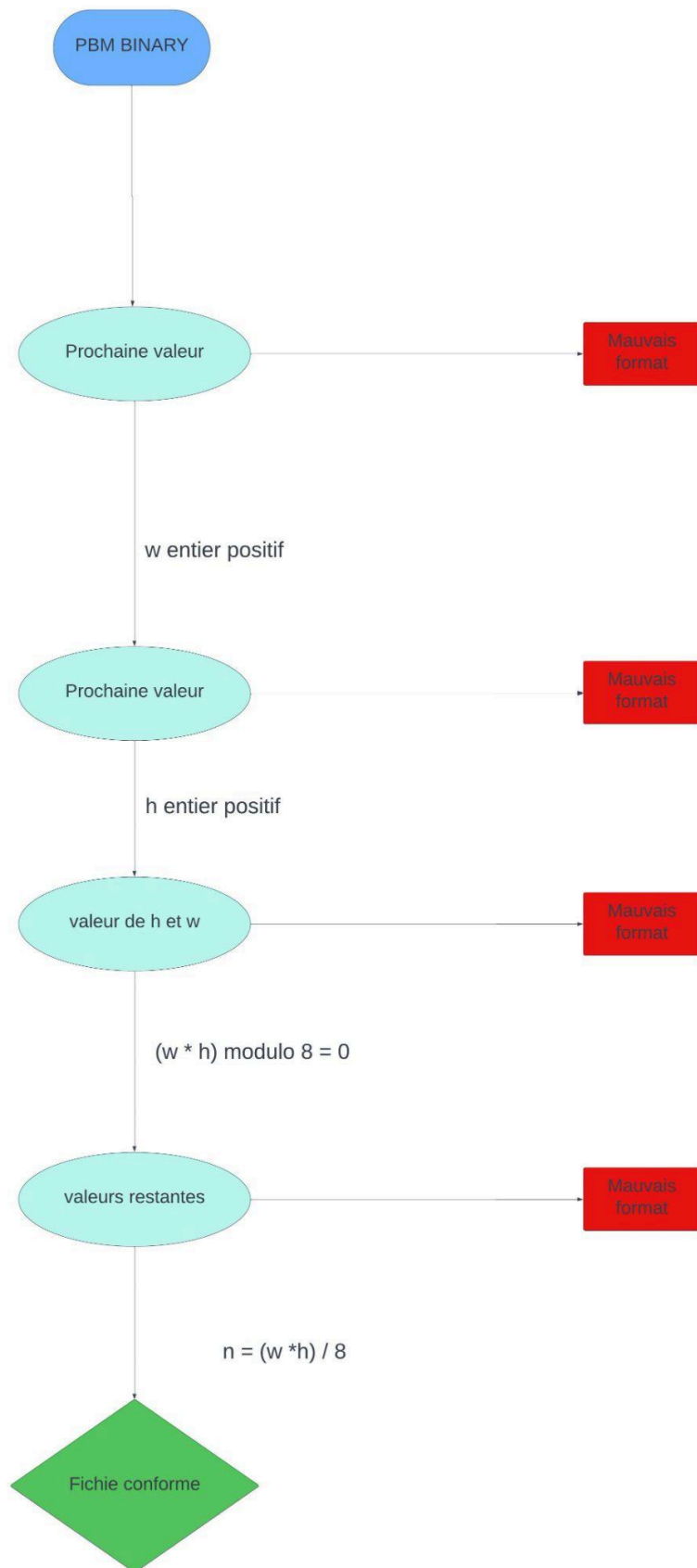
PGM ASCII



● PPM ASCII



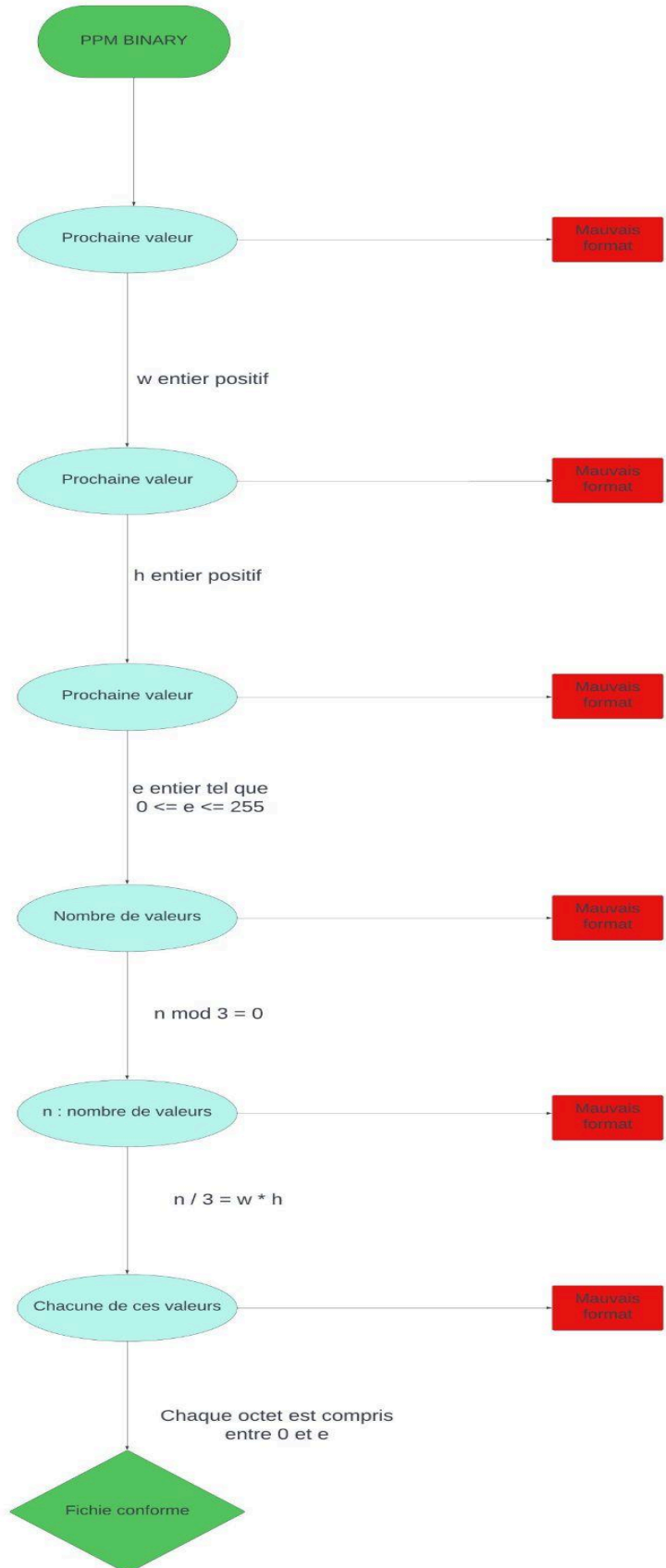
- PBM BINARY



- PGM BINARY



- PPM BINARY



2. Cas de test

Les cas de tests représentent les 12 images qui sont dans le répertoire image du projet. Comme voulu dans le tp, les fichiers se finissant par Y sont les fichiers qui ont réellement un bon format (images tests de 00 à 05) et les fichiers se terminant par N sont les fichiers tronqués ayant donc un mauvais format (images tests de 06 à 11).

On se retrouve donc avec deux images (une image au bon format et une image au mauvais format) pour chaque type de fichiers.

Les 3 images en ASCII ont été récupérées sur la page wikipédia:

<https://en.wikipedia.org/wiki/Netpbm>.

Les 3 autres images en Binaire ont été pris sur la page :

<https://people.sc.fsu.edu/~jburkardt/data/>

Et pour créer les mauvais formats j'ai juste inséré des mauvais caractères dans les images au bon format.

Liste des items représentant les tests et leur fichier de test associés

- Tester si une image PBM ASCII **réellement** au bon format est **effectivement** au bon format: **test00_Y.pbm**
- Tester si une image PGM ASCII **réellement** au bon format est **effectivement** au bon format: **test01_Y.pgm**
- Tester si une image PPM ASCII **réellement** au bon format est **effectivement** au bon format: **test02_Y.ppm**
- Tester si une image PBM BINARY **réellement** au bon format est **effectivement** au bon format: **test03_Y.pbm**
- Tester si une image PGM BINARY **réellement** au bon format est **effectivement** au bon format: **test04_Y.pgm**
- Tester si une image PPM BINARY **réellement** au bon format est **effectivement** au bon format: **test05_Y.ppm**
- Tester si une image PBM ASCII au **mauvais format** est **effectivement** au mauvais format: **test06_N.pbm**

- Tester si une image PGM ASCII au **mauvais format** est **effectivement** au mauvais format: **test07_N.pgm**
- Tester si une image PPM ASCII au **mauvais format** est **effectivement** au mauvais format: **test08_N.ppm**
- Tester si une image PBM BINARY au **mauvais format** est **effectivement** au mauvais format: **test09_N.pbm**
- Tester si une image PGM BINARY au **mauvais format** est **effectivement** au mauvais format: **test10_N.pgm**
- Tester si une image PPM BINARY au **mauvais format** est **effectivement** au mauvais format: **test11_N.ppm**

Résultat des tests

Après avoir effectué tous les tests on remarque que seule l'implémentation numéro 7 de l'application est conforme aux spécifications des images en ASCII. Mais malheureusement, elle n'est pas conforme à la spécification des images en Binaire puisqu'elle répond "true" à des images binaires au mauvais format . Ce qui constitue un problème.

On peut conclure qu'aucune implémentation de l'application répond à la totalité de la spécification.