

Hey ! On m'a dit que tu connaissais bien l'algorithme Ford-Johnson. Tu veux bien m'expliquer ?

Bien sûr ! Voici ce que disent les papiers sources et citations.

Euh...

Je sais c'est pas très clair. Je reformule :

Le tri Ford-Johnson est donc un algorithme récursif qui possède 3 étapes

1. Faire des paires de 2 nombres et faire une nouvelle liste <1> avec le plus grand nombre de chaque paire (stocker les autres nombres dans une autre liste <2>).
2. Relancer l'algorithme sur la nouvelle liste <1> (c'est la récursion)
3. Insérer les nombres de l'autre liste <2> dans la nouvelle liste <1> (qui aura été modifiée à chaque niveau de récursion).

Donc, après avoir lancé récursivement la 1^{ere} étape (faire des paires, récupérer le gagnant et recommencer avec les gagnants) on remonte dans la récursion et on applique la 2^{eme} étape à chaque niveau (créer le malin à partir des gagnants et le perd avec les autres, puis insérer les éléments du perd dans l'ordre de la suite de Jacobsthal)

⚠ Les changements appliqués à la liste doivent se hanter à travers les niveaux de récursions

C'est toujours pas très clair. C'est possible d'avoir un exemple ?

Biensur ! Je vais même te l'expliquer de trois façon différentes (la 3^{ème} méthode est celle que je préfère)

Méthode 1 :

Voici une liste de 25 nombres générée aléatoirement

21, 2, 13, 20, 22, 10, 26, 7, 5, 3, 14, 8, 6, 24, 13, 18, 11, 16, 1, 17, 12, 23, 25, 15

Niveau / Rang de récursion 1 :

Pour commencer on fait des paires de 2 nombres :

21 2, 13 20, 22 10, 26 7, 5 3, 14 8, 6 24, 13 18, 11 16, 1 17, 12 23, 25, 15

On fait une liste avec les nombres les plus grand de chaque paire (on appellera cette liste le main).

Main : 21, 20, 22, 26, 7, 14, 8, 24, 18, 16, 17, 25,

Avec les nombres restant on fait une autre liste (on appellera cette liste le pend).

Pend : 2, 13, 10, 4, 5, 3, 6, 13, 11, 1, 12, 23, 15

Puis on relance l'algorithme sur le main et on passe au 2^{ème} niveau de récursion.

R. 2 :

Liste : 21 20, 22 26, 7 14, 8 24, 18 16, 17, 25

Main : 21, 26, 14, 24, 18, 25

Pend : 20, 22, 7, 8, 16, 17

On relance sur le main et on passe au 3^{ème} niveau de récursion.

R.3:

Liste: 21, 26, 14, 24, 18, 25

Main: 26, 24, 25

Pend: 21, 14, 18

On relance sur le main et on passe au 4^{ème} niveau de récursion.

R.4:

Liste: 26, 24, 25

Main: 26

Pend: 24, 25

Le main n'ayant plus qu'un seul élément on ne peut plus faire de paire.

On va donc remonter dans les niveaux de récursion et insérer le pend dans le main.

Mais on ne va pas insérer linéairement les nombres dans le main! On va les insérer selon la suite de Jacobsthal et en utilisant l'insertion binaire.

Mais pourquoi Jacobsthal ??

En effet, à aucun moment Jacobsthal est mentionné dans les sources que je t'ai donné.

Mais, dans le papier original, il est indiqué une technique de classement qui est la suivante :

$$\text{and } t_j = \{2^{j+1} + (-1)^j\}/3.$$

Or la suite de Jacobsthal est définie par :

$$J_n = J_{n-1} + 2J_{n-2}$$

Mais son application pour les suites récurrentes linéaires est :

$$J_n = \frac{2^n - (-1)^n}{3}$$

Maintenant si on regarde le rang suivant de cette formule on obtient :

$$J_{n+1} = \frac{2^{n+1} - (-1)^{n+1}}{3}$$

$$\Leftrightarrow J_{n+1} = \frac{2^{n+1} - (-1)^n(-1)}{3}$$

$$\Leftrightarrow J_{n+1} = \frac{2^{n+1} + (-1)^n}{3}$$

Dans $t_{n+1} = J_{n+1}$, la formule donnée par Ford-Johnson est la suite de Jacobsthal au rang $n+1$.

Mais comment l'utiliser ??

Comme nous commençons au rang $n+1$ de Jacobsthal
les 2 premiers éléments sont $1, 1$ (et non $0, 1$)
Le début de la suite donne donc :

$1, 1, 3, 5, 11, 21, \dots$

On ne va pas insérer 2 fois 1 (1er élément) donc on supprime le premier.

OKAY, mais il manque des nombres, non ?

Exact ! Il faut rajouter les autres indices. Pour ce faire, entre chaque nombre Jacobsthal on rajoute les indices qui se trouvent entre le Jacobsthal actuel et le précédent en décroissant la Jacobsthal actuel.

Cela donne :

$1, 3, 2$

$1, 3, 2, 5, 4$

$1, 3, 2, 5, 4, 11, 10, 3, 8, 7, 6$

Ookay, mais j'ai pas de 11^{eme} élément à insérer,
j'en l'ai que 10.

Pas de panique il y a une solution!

A chaque nouveau nombre de Jacobsthal on regarde
si il est inférieur ou égal à la size de pend.
Si c'est le cas, alors tout est bon. Si non, on
remplace le nouveau nombre de Jacobsthal par
la size du pend.

Ce qui donne :

1, 3, 2, 5, 4, 10, 9, 8, 7, 6

OKAY! J'ai compris! Maintenant je peux insérer?

Presque! Il te manque juste 2 info.

1. On ne compare pas l'élément à insérer à tout les
élément de main. Comme on sait que l'élément
est forcément plus petit que sa paire qui se
trouve dans le main et que ce dernier est trié,
l'élément doit forcément s'insérer avant sa paire et
ne jamais être comparé à sa paire.

L'élément a donc sa paire comme borne supérieure
(par défaut la borne inférieur est 0) (les borne sont
les indices des numéros dans la liste).

Cela crée donc une zone de comparaison autorisée.

2. On n'insère pas linéairement dans le main (on ne compare pas l'élément à insérer avec tous les éléments du main). On utilise l'insertion binaire.

On compare l'élément à insérer avec l'élément qui se trouve au milieu de la zone de comparaison autorisé.

Si l'élément est inférieur au milieu alors la nouvelle borne supérieur est le milieu.

Si non, la nouvelle borne inférieur est le milieu

Puis on recommence avec la nouvelle zone de comparaison jusqu'à qu'il n'y ai plus que 1 indice de différence entre les 2 bornes ou que les bornes aient le même indice.

A quand même ! Ça fait pas mal d'info !

Oui je sais, mais tient bon ! On reprend à l'insertion du rang 4

On a:

Liste: 26, 24, 25

Main: 26
a₀

Pend: 24, 25
b₀ b₁

donc la liste d'insertion est :

1, 2

J'ai placé sous chaque nombre un indice en vert foncé. Ils vont te permettre de suivre les mouvements des paires plus facilement. Les grands éléments des paires sont représentés par la lettre a et les petits éléments ou les éléments non paireés par la lettre b. Le chiffre en indice indique le numéro de la paire.

Main: 26
 a_0

donc la liste d'insertion est:

Pend: 24, 25
 b_0 b_1

1, 2

On commence par insérer le 1^{er} élément b_0 .
Comme il est borné par a_0 il se trouve forcément avant.

Main: 24, 26
 b_0 a_0 a_1 (fictif)

Ensuite on insère b_1 . Comme c'est un élément non borné il n'a pas de borne. La borne supérieur devient alors un a_1 fictif.

La borne inférieur est donc b_0 et sa borne supérieur a_1 .

L'indice médian est donc a_0 . On compare le nombre à insérer au nombre à l'indice de a_0 .

$$25 < 26$$

La borne supérieur devient donc a_0 .

La différence entre les bornes étant égal à 1 on stoppe la recherche binaire et on insère le nombre à l'indice < borne inférieur > + 1 donc $b_0 + 1$

On obtient donc :

Main: 24, 25, 26

La liste est trié, on peut donc passer au niveau de récursion précédent.

R.3: Voici où on en avait laissé le niveau 3:

Liste: $\underline{21, 26}, \underline{14, 24}, \underline{18, 25}$

Main: $26, 24, 25$
 $a_0 \quad a_1 \quad a_2$

Pend: $21, 14, 18$
 $b_0 \quad b_1 \quad b_2$

Mais attention! Dans le niveau de récursion précédent (R.4) l'ordre du main a été modifié, il faut donc réorganiser l'ordre du main du niveau 3 par rapport à la nouvelle liste produite au niveau 4.

Au R.4 nous avions fini avec Main: 24, 25, 26.

Ce sera donc notre nouveau main:

Main: $24, 25, 26$
 $a_1 \quad a_2 \quad a_0$ Mais maintenant les nombres perdant qui se trouvent dans le pend ne sont plus dans le même ordre que leur pair, il faut donc les réorganiser en fonction de l'ordre du main

On obtient donc:

Main: $24, 25, 26$
 $a_1 \quad a_2 \quad a_0$ Maintenant que les listes sont ordonnées correctement on réinitialise les indices des pairs (les a et le b doivent toujours être triés au sein de leur liste correspondante)

Pend: $14, 18, 21$
 $b_1 \quad b_2 \quad b_0$

Pend: $14, 18, 21$
 $b_0 \quad b_1 \quad b_2$

On peut maintenant insérer le pend dans la main.

La liste d'insertion est: 1, 3, 2 $\rightarrow b_0, b_2, b_1$

Main: 14, 18, 21, 24, 25, 26
 $b_0, b_1, b_2, a_0, a_1, a_2$

Pend: 14, 18, 21
 b_0, b_1, b_2

Le main est trié, on passe au rang précédent.

R 2:

Réorganisation des paires:

Main: 21, 26, 14, 24, 18, 25

14, 18, 21, 24, 25, 26

Pend: 20, 22, 7, 8, 16, 17
 $a_0, a_1, a_2, a_3, a_4, a_5$
 $b_0, b_1, b_2, b_3, b_4, b_5$

7, 16, 20, 8, 17, 22
 $a_2, a_4, a_0, a_3, a_5, a_1$
 $b_2, b_4, b_0, b_3, b_5, b_1$

Réinitialisation
des indices:

14, 18, 21, 24, 25, 26

Insertion:

$a_0, a_1, a_2, a_3, a_4, a_5$
7, 16, 20, 8, 17, 22
 $b_0, b_1, b_2, b_3, b_4, b_5$

7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26
 $b_0, b_3, a_0, b_1, b_4, a_1, b_2, a_2, b_5, a_3, a_4, a_5$

On passe au niveau précédent.

R. I.

Main: 2, 1, 20, 22, 26, 7, 14, 8, 24, 18, 16, 17, 25,
 a_0 , a_1 , a_2 , a_3 , a_4 , a_5 , a_6 , a_7 , a_8 , a_9 , a_{10} , a_{11}

Pend: 2, 19, 10, 4, 5, 3, 6, 13, 11, 1, 12, 23, 15
 b_0 , b_1 , b_2 , b_3 , b_4 , b_5 , b_6 , b_7 , b_8 , b_9 , b_{10} , b_{11} , b_{12}

Nouveau main:

7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26,
 a_4 , a_6 , a_5 , a_9 , a_{10} , a_8 , a_1 , a_0 , a_2 , a_7 , a_{11} , a_3

Nouveau pend:

5, 6, 3, 1, 12, 11, 13, 2, 10, 13, 23, 4, 15
 b_4 , b_6 , b_5 , b_3 , b_{10} , b_8 , b_1 , b_0 , b_9 , b_7 , b_{11} , b_2 , b_{12}

Ordre d'insertion: 1, 3, 2, 5, 4, 11, 10, 9, 8, 7, 6, 13, 12

Nouvelle liste après insertion:

1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
 b_3 , b_0 , b_5 , b_6 , b_4 , b_6 , a_4 , a_6 , b_9 , b_3 , b_{10} , b_7 , a_5 , b_{12} , a_9 , a_{10} , a_8 , b_1 , a_1 , a_0 , a_2 , b_{11} , a_7 , a_{11} , a_3

La liste est maintenant trié!

Je la refais avec la montée des niveaux de récursion à côté de la redesccente, ça sera un peu plus clair.

Liste initial :

21, 2, 19, 20, 22, 10, 26, 4, 7, 5, 3, 14, 8, 6, 24, 13, 18, 11, 16, 1, 17, 12, 23, 25, 15

Liste finale :

1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
 $b_3 b_0 b_5 b_3 b_4 b_6 a_4 a_6 b_7 b_8 b_{10} b_7 b_5 b_{12} a_9 a_{10} a_8 b_1 a_1 a_0 a_2 b_{11} a_7 a_{11} a_3$

R. 1 : descente de la récursion

Remonté de la récursion

Main :

21, 20, 22, 26, 7, 14, 8, 24, 18, 16, 17, 25,

Pend :

2, 19, 10, 4, 5, 3, 6, 13, 11, 1, 12, 23, 15

R. 2 :

Liste :

21, 20, 22, 26, 7, 14, 8, 24, 18, 16, 17, 25,

Main : 21, 26, 14, 24, 18, 25

Pend : 20, 22, 7, 8, 16, 17

R. 3 :

Liste : 21, 26, 14, 24, 18, 25

Main : 26, 24, 25

Pend : 21, 14, 18

R. 4 :

Liste : 26, 24, 25

Main : 26

Pend : 24, 25

7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26,
 $a_4 a_6 a_5 a_9 a_{10} a_8 a_1 a_0 a_2 a_7 a_{11} a_3$

5, 6, 3, 1, 12, 11, 13, 2, 10, 13, 23, 4, 15
 $b_4 b_6 b_5 b_9 b_{10} b_8 b_1 b_0 b_2 b_7 b_3 b_{11} b_{12}$

7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26
 $b_0 b_3 a_0 b_1 b_9 a_1 b_2 a_2 b_5 a_8 a_4 a_5$

14, 18, 21, 24, 25, 26
 $a_0 a_1 a_2 a_5 a_4 a_6$

7, 16, 20, 8, 17, 22
 $b_0 b_1 b_2 b_3 b_9 b_5$

Liste : 14, 18, 21, 24, 25, 26
 $b_0 b_1 b_2 a_0 a_1 a_2$

Main : 24, 25, 26
 $a_0 a_1 a_2$

Pend : 14, 18, 21
 $b_0 b_1 b_2$

Liste : 24, 25, 26

Main : 26

Pend : 24, 25
 $b_0 b_1$

Méthode 2:

Maintenant je recommence mais cette fois ci je vais faire un arbre.

On prend la liste et on fait des paire:

2 1, 2 1, 3 2, 0 2, 2 2, 1 0, 2 6, 4 7, 5 3, 1 4, 8 6, 2 4, 1 3, 1 8, 1 1, 1 6, 1 1, 1 7, 1 2, 2 3, 2 5, 1 5

Avec les gagnants on va rajouter un niveau à l'arbre.
Chaque niveau de l'arbre correspond à un niveau de récursion.

Cela nous donne:

Niveaux de récursion: R



Mais attention ! comme dans la méthode précédente les changements fait à la liste doivent être appliqués à tous les niveaux de récursions.

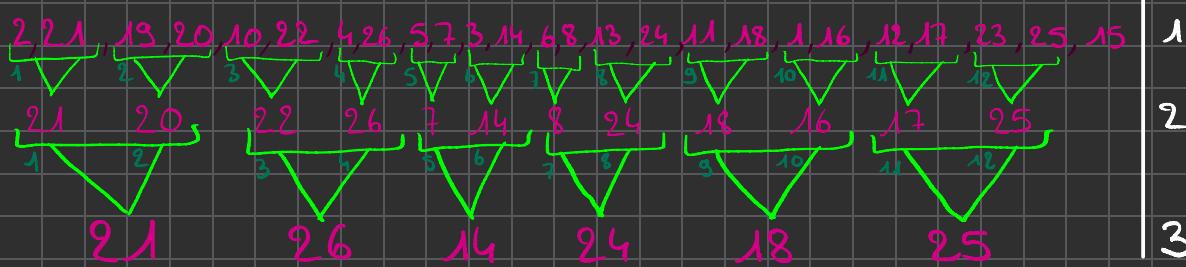
On va donc permuter les branches des niveaux supérieurs en fonction de l'élément le plus grand:



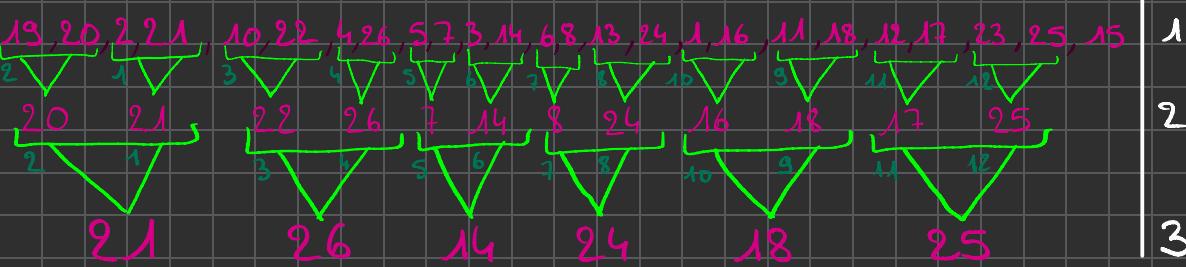
Ensuite on recommence: on fait des paires:



On crée un nouveaux niveau avec les plus grands éléments des paires:



Et on permute les branches en fonction des plus grands éléments:



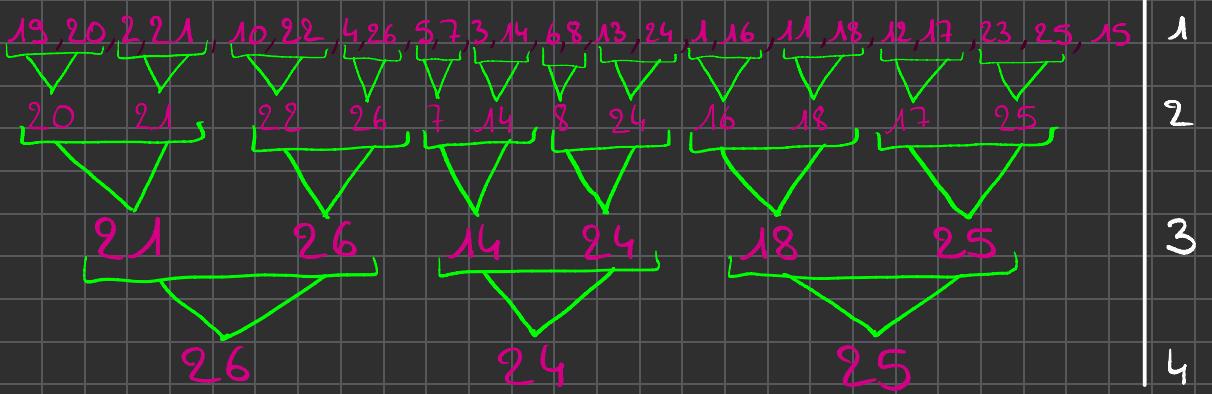
(Psst, pour voir les permutations regarde les petits chiffres en vert foncé)

Et on recommence!

On fait des paires:



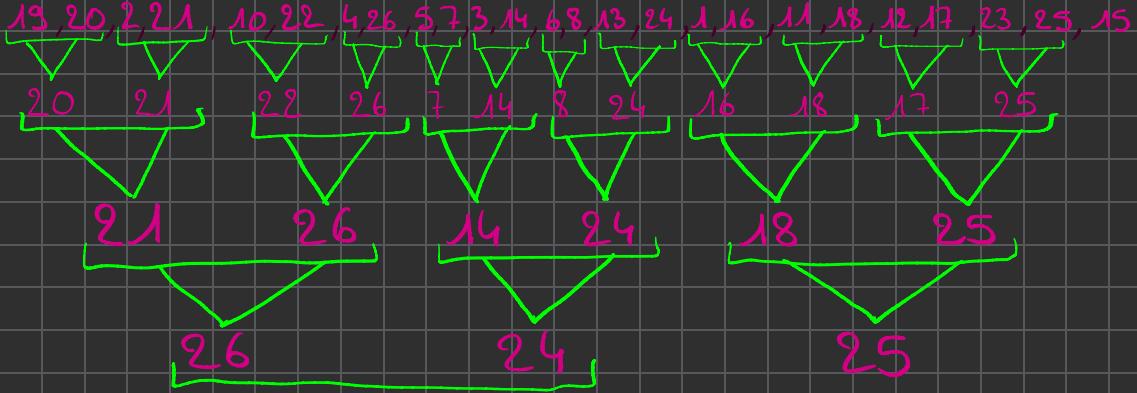
On extrait les gagnants:



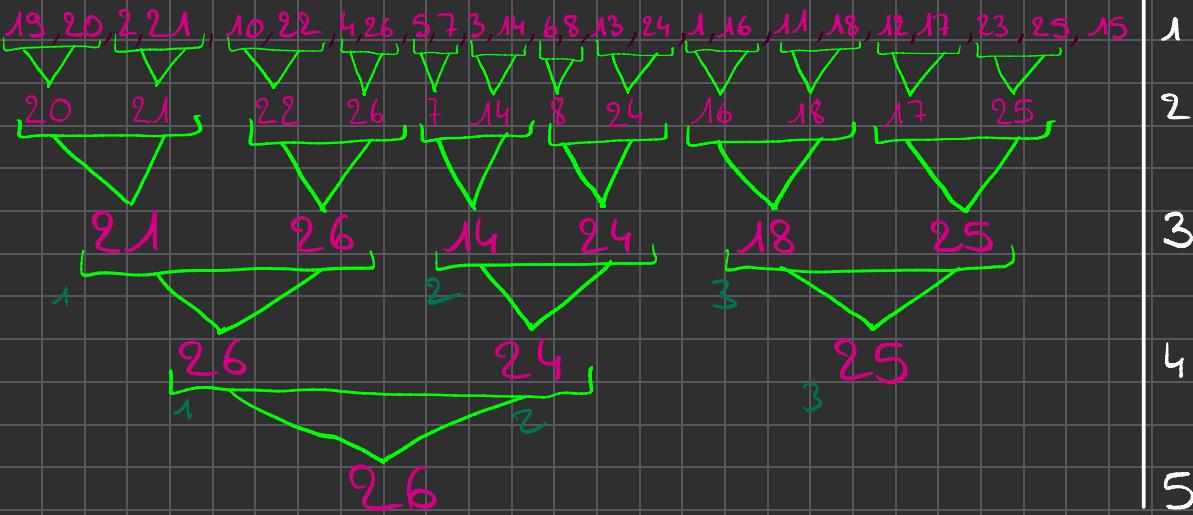
Ici il n'y a pas de permutation à faire car les éléments de chaque paire du niveau 3 sont déjà triés entre eux.

On passe donc au niveau suivant!

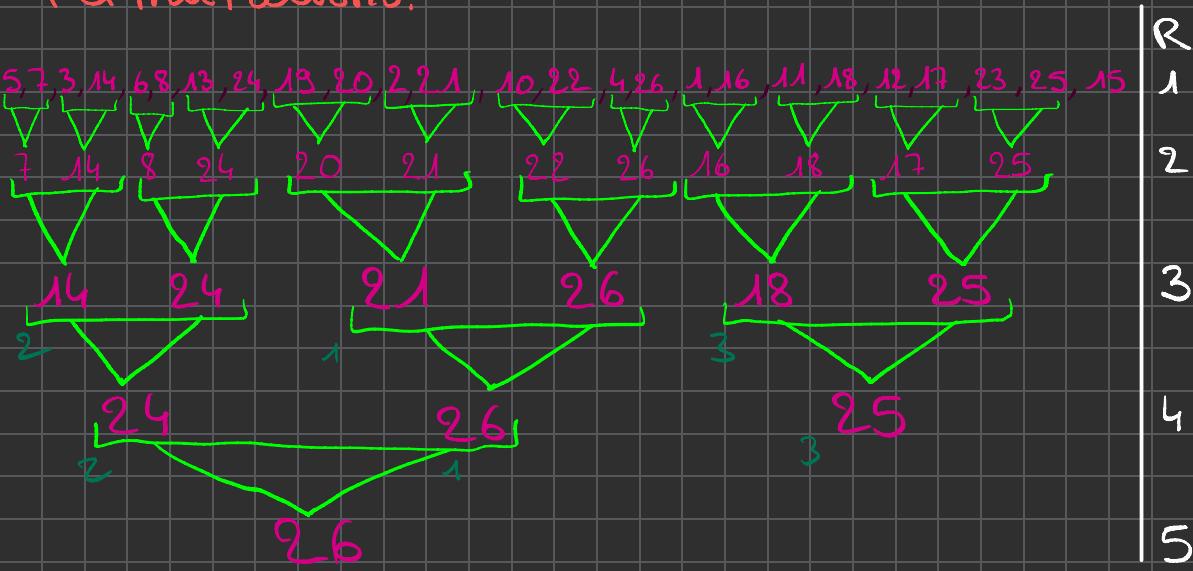
Paînes:



Gagnants:



Permutations:



Il n'y a plus qu'un seul éléments, on peut donc plus faire de paire.

On va donc passer à la deuxième étape en partant du dernier niveau de récursion (5ème niveau de l'arbre).

Avec les plus grands éléments des paires de la liste de nombres de ce niveau on crée la liste main.

Et avec les plus petits et l'élément non paire potentiel on fait la liste pend.

R . 5

Le nombre 26 étant seul dans la liste, il n'y a rien à faire sur ce rang de récursion

R . 4 26 24 25

Il a déjà été établie dans l'étape précédente que 26 est le seul gagnant. Il formera donc la main, et les autres éléments le pend

Main: 26

Pend: 24 25

Nous allons donc insérer chaque élément du Pend dans la main en fonction de la suite de Jacobsthal

Main: 26 | donc la liste d'insertion est:

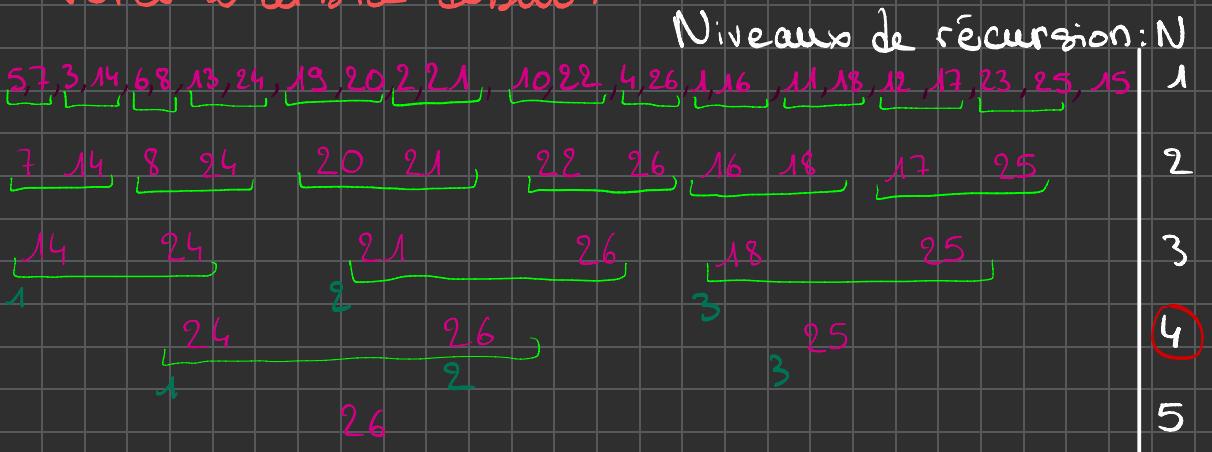
Pend: 24 25 | 1 , 2

Ce qui donne après insertion :

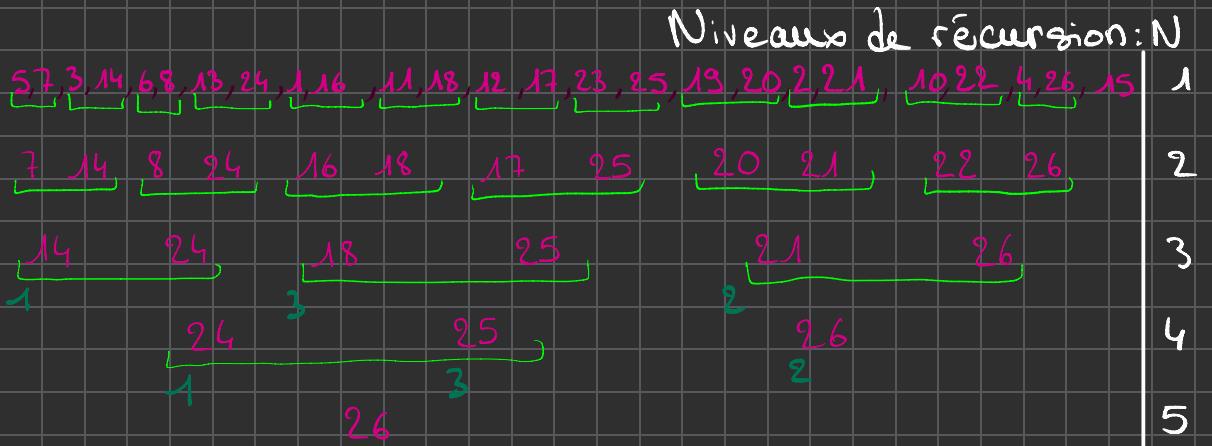
Main: 24 , 25 , 26

Mais attention ! Je t'ai dit que les changements appliqués à la liste doivent subsister à travers les niveaux de récursions. Il faut donc permuter les branches entière de l'arbre

Voici l'arbre début :



Mais on a permute des éléments du rang 4.
L'arbre devient donc :



Puis on passe au rang 3.

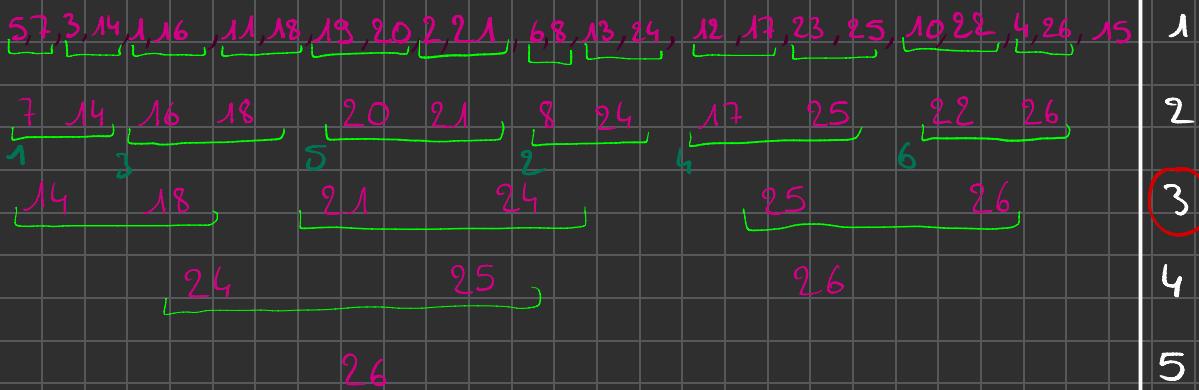
R.3: 14 24 18 25, 21 26

Main: 24 25 26 Liste d'insertion:

Pend: 14 18 21 | 1, 3, 2

Main: 14, 18, 21, 24, 25, 26

Niveaux de récursion: N



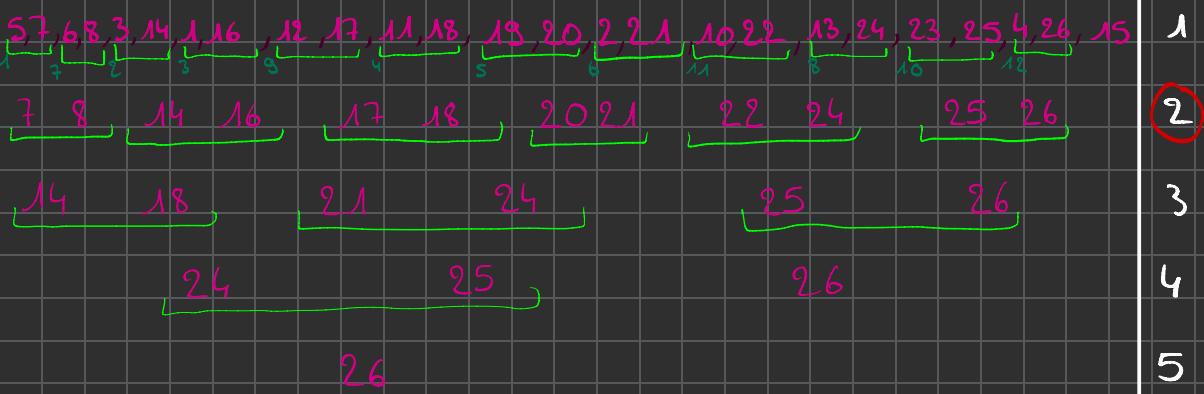
R. 2: 7, 14, 16, 18, 20, 21, 8, 24, 17, 25, 22, 26

Main: 14 18 21 24 25 26

Pend: 7 16 20 8 17 22

Main: 7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26

Niveaux de récursion: N



R. 1:

5, 6, 3, 1, 12, 11, 13, 2, 10, 13, 23, 4, 15

Main: 7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26

Pend: 5, 6, 3, 1, 12, 11, 13, 2, 10, 13, 23, 4, 15

Main:

1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26

Et voilà c'est trié!

M.3

Pour cette méthode au lieu de subdiviser récursivement la liste je vais, à chaque niveau de récursion agrandir la taille des éléments des paires.

Dans les méthodes précédents les paires étaient toujours composées de deux éléments de 1 nombres.

Ici, à chaque niveau de récursion, la taille de chaque élément par paire va être multiplié par 2.

Dans la première partie de la récursion (la comparaison et l'échange des éléments de chaque paire), cela va nous permettre de ne jamais avoir à subdiviser la liste initiale.

L'autre avantage majeur de cette méthode c'est que nous n'avons pas à appliquer les changements fait sur un niveau de récursion sur tout les autres niveaux (comme on va bouger des blocs entier les changement s'appliqueront automatiquement).

Reprendons donc la liste initiale.

21, 2, 19, 20, 22, 10, 26, 4, 7, 5, 3, 14, 8, 6, 24, 13, 18, 11, 16, 1, 17, 12, 23, 25, 15

R. 1:

On fait des pairets:

21 2 19 20 22 10 26 4 7 5 3 14 8 6 24 13 18 11 16 1 17 12 23, 25, 15

Et on échange les éléments des au sein des pairets si nécessaire.

21 2 19 20 22 10 26 4 7 5 3 14 8 6 24 13 18 11 16 1 17 12 23, 25, 15
~~22~~ ~~1~~ ~~18~~ ~~20~~ ~~10~~ ~~22~~ ~~4~~ ~~26~~ ~~5~~ ~~7~~ ~~3~~ ~~14~~ ~~6~~ ~~8~~ ~~13~~, ~~24~~ ~~11~~, ~~18~~ ~~1~~, ~~16~~ ~~12~~, ~~17~~, ~~23~~, ~~25~~, ~~15~~

R. 2:

On multiplie la taille des éléments par 2. Nous avons donc maintenant des éléments de 2 nombres (et donc des pairets de 4 nombres).

éléments

2 21 19 20 10 22 4 26 5 7 3 14 6 8 13, 24 11, 18 1, 16 12, 17 23, 25, 15

pairets

Maintenant on compare les derniers nombres de chaque éléments (les plus grands) au sein de chaque pairet et on échange si nécessaire.

2 21 19 20 10 22 4 26 5 7 3 14 6 8 13, 24 11, 18 1, 16 12, 17 23, 25, 15
~~22~~ ~~1~~ ~~18~~ ~~20~~ ~~10~~ ~~22~~ ~~4~~ ~~26~~ ~~5~~ ~~7~~ ~~3~~ ~~14~~ ~~6~~ ~~8~~ ~~13~~, ~~24~~ ~~11~~, ~~16~~ ~~18~~ ~~1~~, ~~17~~ ~~12~~, ~~13~~, ~~23~~, ~~25~~, ~~15~~
car $21 > 20$ car $18 > 16$

R.3:

On passe à 4 nombres par éléments (paire de 8 nombres)

13 20, 221 1022 4, 26, 57, 3, 14, 6, 8, 13, 24, 1, 16, 11, 18, 12, 17, 23, 25, 15

On compare le dernier nombre de chaque éléments et on permute les éléments si nécessaire.

A ce niveau il n'y a pas de permutation à faire
En effet :

$$21 < 26 ; 14 < 24 ; 18 < 25$$

R.4:

Éléments de 8 nombres.

13 20, 221 1022 4, 26, 57, 3, 14, 6, 8, 13, 24, 1, 16, 11, 18, 12, 17, 23, 25, 15

On compare et échange si nécessaire

13 20, 221 1022 4, 26, 57, 3, 14, 6, 8, 13, 24, 1, 16, 11, 18, 12, 17, 23, 25, 15

car $26 > 24$

57, 3, 14, 6, 8, 13, 24, 13, 20, 221 1022 4, 26, 1, 16, 11, 18, 12, 17, 23, 25, 15

R.5:

Élement de 16 nombres.

57, 3, 14, 6, 8, 13, 24, 13, 20, 221 1022 4, 26, 1, 16, 11, 18, 12, 17, 23, 25, 15

A ce rang on ne peut plus faire de paire car la taille des éléments excède la moitié de la taille de la liste.

On va donc pour passer à l'insertion et remonter les niveaux de récursion.

Pour faire le main, on prend chaque élément gagnant des paires (on prend le bloc entier d'élément).

Pour faire le perd, on prend chaque élément perdant des paires. On prend aussi l'élément potentiellement non peinté (il peut y avoir un nombre différent d'élément) triés au sein d'une même paire il suffit de prendre le dernier élément de chaque paire. Si il n'y a plus assez de nombre pour faire un élément ces derniers sont considérés comme reste, laissés dans le main et seront ignorés pour la comparaison.

Reprendons au rang 5

5, 7, 3, 14, 6, 8, 13, 24, 19, 20, 2, 21, 10, 22, 4, 26, 1, 16, 11, 18, 12, 17, 23, 25, 15

Ici nous avons qu'un seul élément (les 9 nombres qui restent à la fin de la liste ne sont pas assez pour former un élément, il sont donc ignorés)

les éléments de cette liste sont donc déjà triés.
On considère les éléments d'une liste triés quand leur plus grand nombre (celui le plus à droite dans l'élément) sont triés entre eux.

Cette liste est donc considérée comme triée à ce rang de récursion.

On divise la taille de l'élément par 2 pour revenir au rang de récursion précédent.

R. 4 : 8 nombres par éléments

57, 3, 14, 6, 8, 13, 24, 19, 20, 2, 21, 10, 22, 4, 26, 1, 16, 11, 18, 12, 17, 23, 25, 15, reste

On crée donc le main avec le plus grand élément de chaque paire et le reste. Comme les éléments sont triés au sein d'une même paire il suffit de prendre la 2^e élément de chaque paire.

Et le pend avec le plus petit élément de chaque paire et l'élément non paire.

Main: 19, 20, 2, 21, 10, 22, 4, 26, 15

Pend: 57, 3, 14, 6, 8, 13, 24, 1, 16, 11, 18, 12, 17, 23, 25

Puis on insère les éléments du pend dans le main

Main:

57, 3, 14, 6, 8, 13, 24, 1, 16, 11, 18, 12, 17, 23, 25, 19, 20, 2, 21, 10, 22, 4, 26, 15

les plus grands nombres des éléments sont triés ($24 < 25 & 26$) donc la liste est triée à ce rang de récursion.

On divise la taille de l'élément par 2 pour passer au rang de récursion précédent.

R.3: 4 nombres par éléments

57, 3, 14, 6, 8, 13, 24 116, 11, 18 12, 17, 23, 25 19, 20, 2, 21 10, 22, 4, 26, 15

Main: 6, 8, 13, 24 12, 17, 23, 25 10, 22, 4, 26, 15

Pend: 57, 3, 14 116, 11, 18 19, 20, 2, 21,

On insère:

Main:

57, 3, 14, 116, 11, 18 19, 20, 2, 21 6, 8, 13, 24 12, 17, 23, 25 10, 22, 4, 26, 15

R.2: 2 nombres par éléments

3, 14 116 11, 18 19, 20 2, 21 6, 8, 13, 24 12, 17, 23, 25 10, 22, 4, 26, 15

Main: 3, 14 11, 18 2, 21 13, 24 23, 25 4, 26, 15

Pend: 57 116 19, 20 6, 8 12, 17, 10, 22,

On insère:

Main:

57 6, 8 3, 14 116 12, 17 11, 18 19, 20 2, 21 10, 22 13, 24 23, 25 4, 26, 15

R. 1.1 nombres par éléments

5 7 6 8 3 14 1 16, 12 13, 11 18, 19, 20 2 21, 10 22, 13, 24, 23 25 4 26, 15

Main 7, 8, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26

Pend : 5, 6, 3, 1, 12, 11, 19, 2, 10, 13, 23, 4, 15

On insère :

Main :

1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26

Et voilà la liste est trié.

Petit tips :

Attends 2 secondes ! Tu m'as dit que je ne devais pas comparer l'élément avec sa borne. Mais du coup comment tu sais où t'arrêter dans l'insertion ?

Avec les indices ! Si le nombre de Jacobsthal est 4, l'indice de l'élément que tu insères c'est $4 - 1 = 3$ et de l'indice de la paire dans la main c'est aussi 4.

Euh ouais mais quand tu insères des éléments dans le main sa décale les indices des paires dans le main, non ?

Oui ! Tu as raison ! Je vais t'expliquer comment résoudre ce problème.

Il va falloir rajouter un décalage aux indices du main.

A chaque fois que tu vas insérer un nombre dans le main tu sauvegarde l'indice.

Pour rappel tu suis la liste d'insertion fais grâce à la suite de Jacobsthal pour savoir quel indice du pend à insérer.

Voici une règle très simple :

Si l'indice de l'élément à insérer est supérieur à l'indice précédent (celui de la liste d'insertion)

Alors le décalage à ajouter est égal au nombre d'indice que tu as sauvégarde
Sauvegarde le décalage (il faudra l'ajouter à chaque indice après) et supprime tout les indices sauvégarde.

Si non, tu dois comparer tous les indices sauvegardés à l'indice supposé du main et le rajouter momentanément au décalage.

Pseudo code

M. 1:
{ list F(list &main)

- Make pairs of 2 elements and extract in a new list
(we will call it pend) the smaller element of the pair

- Si main.size() >= 2

↳ F(main) → (main is modified as it is a reference)

- Insert pend into main → (as main members have been moved they are not aligned with their pair in the pend, so you need to find a solution to either reorganize the pend or keep track of main members indexes)
↳ struct with int* with pair index

}

A venir

1. Pseudo code des méthodes 2 et 3.
2. Un github avec les 3 méthodes implementées + un Makefile / Bash de lancer pour tester le projet.
3. Une explication de comment vérifier que son algorithme est bon (formule du nombre maximum de comparaison).