Devoir #3

Question 1:

Le programme va imprimer:

"dans abc avant addition x= 10"
"dans abc après addition x= 110"
"dans abc avant addition x= 10"
"dans abc après addition x= 120"
"dans main : a=110, x= 120"

En effet, dans la fonction main, x et a sont déclarés et initialisés à 100 et 20 respectivement. Lorsque la fonction abc est appelée premièrement dans main avec comme argument x=100,la valeur de x est en ce moment 100. Ensuite dans la fonction abc, on a déclaré et initialisé une variable x=10(variable locale qui n'a rien avoir avec la variable x dans main),puis la fonction abc imprime la valeur de x qui est 10;après elle affecte a x la valeur "x+a" soit "10 + 100 = 110" et imprime la nouvelle valeur de x qui est 110 qui est retournée a la variable a dans main (la valeur initiale de a qui est 20 est écrasée et remplacée par cette nouvelle valeur). A ce niveau, dans la fonction main,les valeurs de a et x sont respectivement 110 et 100. La fonction abc est à nouveau appelée dans main mais cette fois ci avec comme argument a=110 (dernière valeur stockée dans a). Ainsi la fonction imprime la variable x (locale) qui est toujours de 10 avant la somme,puis affecte à x la somme "x+a" soit "10 +110" qui est 120 qu'elle imprime. Cette valeur de x(dans abc) est retournée à la variable x dans main initialement de 100; la nouvelle valeur de x dans main est donc 110. Enfin la fonction main imprime les valeurs de a et x qui sont respectivement 110 et 120.

Question 2:

La sortie sera:

"dans main: x= 100"
"dans abc avant addition: x=10"
"dans abc après addition: x=110"
"dans main : x=110"
"dans abc avant addition: x=110"

"dans abc après addition: x=220"
"dans main : x=220"

En effet, dans la fonction main, x est une variable locale initialisée à 100,ensuite la fonction imprime la valeur de x qui est de 100. La fonction abc est alors appelée avec comme argument x=100. Dans la fonction abc, on a déclaré et initialisé une variable x=10 (variable locale de abc qui n'a rien avoir avec la variable x dans main),puis la fonction abc imprime la valeur de x qui est 10;après elle affecte à x la valeur "x+a" soit "10 + 100 = 110" et imprime la nouvelle valeur de x qui est 110 qui est retournée a la variable x dans main (la valeur initiale de x qui était 100 est écrasée et remplacée par cette nouvelle valeur). La fonction abc est a nouveau appelée dans main mais cette fois ci avec comme argument x=110 (dernière valeur stockée dans x). Ainsi la fonction abc imprime la variable x (locale) qui est de 110 (mais pas 10 car x est déclaré dans abc avec la classe "static" donc la valeur de x n'est pas réinitialisé) avant la somme,puis affecte à x la somme "x+a" soit "110 +110" qui est 220 qu'elle imprime. Cette valeur de x(dans abc) est retournée à la variable x dans main; la nouvelle valeur de x dans main est donc 220. Enfin la fonction main imprime la valeur de x qui est 220.

Question 3:

Etape 1 : Enoncé du problème

On doit écrire une fonction qui retourne une valeur du type int et qui n'accepte pas d'argument qui renvoie le i-eme nombre de Fibonacci lorsqu'elle appelée la i-eme fois. Cependant dans la fonction main, le programme doit demander a l'utilisateur et vérifier la positon du nombre de Fibonacci qu'il souhaite déterminer.

Etape 2 : Cueillette d'informations et description des entrées et sorties

Le i-ième nombre de Fibonacci Xi est lié aux (i - 1)-ième et (i - 2)-ième nombres de Fibonacci Xi1 et Xi2 par Xi = Xi1 + Xi2. Le premier nombre de Fibonacci est 1, le deuxième est 1, le troisième est 2, le quatrième 5 et ainsi de suite.

Entrées :
Dans main : rang_du_nombre.

Sortie :
dans main : 'nombre incorrect. Veuillez entre un nombre positif'; X.
La fonction n'a pas d'entées, elle retourne uniquement une valeur dans main.

Etape 3 : Cas de test et algorithme

Cas de test :

1-Si l'utilisateur entre -8, le programme doit affiché un message d'erreur et redemander a celui ci d'entrer un nombre valide. Si il entre de nouveau 0, alors le programme doit ré-affiché un message d'erreur et redemander d'entrer un nombre valide.

2- Si l'utilisateur entre 1, le programme doit afficher le message suivant : 'Le 1-ieme nombre de Fibonacci est : 1'.

3- Si l'utilisateur entre 2, le programme doit afficher le message suivant : 'Le 1-ieme nombre de Fibonacci est : 1'.

4- Si l'utilisateur entre 8, le programme doit afficher le message suivant : 'Le 1-ieme nombre de Fibonacci est : 21'.

Algorithme :

Fonction getNextFibonacciNumber qui n'accepte aucune valeur et retourne un entier

Déclaration et initialisation du rang des éléments de la séquence à 1

Déclaration et initialisation du premier élément Xi-2 comme entier statique a

Déclaration et initialisation du premier élément Xi-1 comme entier statique b

Si x est inférieur strict à 3
Affecter 1 à X

Imprimer le rang et la valeur de X
Sinon
Affecter a+b à X
Imprimer le rang et la valeur de X
Incrémenter rang
Affecter b à a
Affecter X à b
retourner X comme valeur
Fonction 'main'

Répéter
Imprimer 'Le quellieme nombre de Fibonacci voulez vous?'
lire dans c
Si (c est inférieur ou égal a 0)
imprimer 'Nombre incorrect. Veuillez un nombre positif'

Sinon
Pour i allant de 0 a c avec un pas de 1
X= getNextFibonacciNumber

imprimer Le c-eme nombre de Fibonacci est', X
PendantQue (c est inférieur ou égal a 0)

Etape 4 : Implémentation

fichier joint C

Etape 5 : Vérification

Test1 :

```
Le Quelieme nombre de Fibonacci voulez-vous: -8
Nombre incorrect. Veuillez entrer un nombre positif!

Le Quelieme nombre de Fibonacci voulez-vous: 0
Nombre incorrect. Veuillez entrer un nombre positif!

Le Quelieme nombre de Fibonacci voulez-vous:
```

Test2 :

```
Le Quelieme nombre de Fibonacci voulez-vous: 1
Le 1-ieme nombre de Fibonacci est: 1
Program ended with exit code: 0
```

Test3 :

```
Le Quelieme nombre de Fibonacci voulez-vous: 2
Le 2-ieme nombre de Fibonacci est: 1
Program ended with exit code: 0
```

Test4 :

```
Le Quelieme nombre de Fibonacci voulez-vous: 8
Le 8-ieme nombre de Fibonacci est: 21
Program ended with exit code: 0
```

Question 4 :

Etape 1 : Enoncé du problème

On doit écrire un programme qui invite l'utilisateur à entrer un entier positif N et affiche l'ensemble de chaînes binaires {0, 1}.

Etape 2 : Cueillette d'informations et description des Entrées et Sorties

L'ensemble des chaines binaires sont composés des éléments {0,1}. L'utilisateur entre un nombre N, ce nombre représentera le nombre de colonnes d'un tableau bidimensionnelle qui stockera toutes les combinaisons possible. Le nombre de lignes de ce tableau est déterminée par la formule 2^N ou N est le nombre entré par l'utilisateur.

Entrées et Sorties :

Entrée : N

Sortie : Tableau bidimensionnel/'Nombre incorrect. Veuillez ressaisir svp"

Etape 3 : Cas de test et algorithme

Cas de test :

Tests	Entrées	Sorties
1	-4	Nombre incorrect. Veuillez ressaisir svp!
2	0	Nombre incorrect. Veuillez ressaisir svp!
3	3	0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1
4	4	0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0 0 1 0 1 0 1 1 0 0 1 1 1 1 0 0 0 1 0 0 1 1 0 1 0 1 0 1 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 1

Algorithme :

Répéter
Imprimer 'Saisir un entier positif'
Lire dans N

Si (N est strictement inférieur a 1)
Imprimer 'Nombre incorrect. Veuillez ressaisir svp!'
Sinon
Affecter 2^N à la variable nombre_total_de_lignes
Affecter nombre_total_de_lignes à la variable d

Déclarer tab [nombre_total_de_lignes][colonne]

Pour colonne allant de 0 à N-1
Pour j allant de 0 a d/2
Affecter 0 à tab [j][colonne]

Pour k allant de d/2 a N-1
Affecter 1 à tab [k][colonne]

Si(colonne est strictement supérieur a 0)
Affecter 0 a j

Pour l allant de d à nombre_total_de_lignes
Affecter tab[j][colonne] à tab[l][colonne]
Incrémenter j

Affecter d/2 à d

Pour i allant de 0 à nombre_total_de_lignes
Pour j allant de 0 à N-1
Imprimer tab[i][j]

Retour à la ligne;

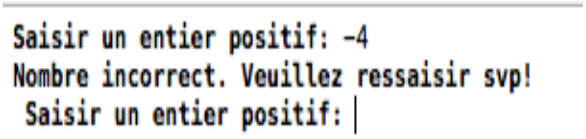
PendantQue(N est strictement inférieur à 1)

Etape 4 :Implémentation

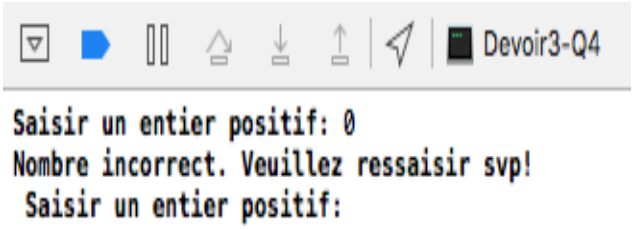
Fichier c

Etape 5 : Vérification

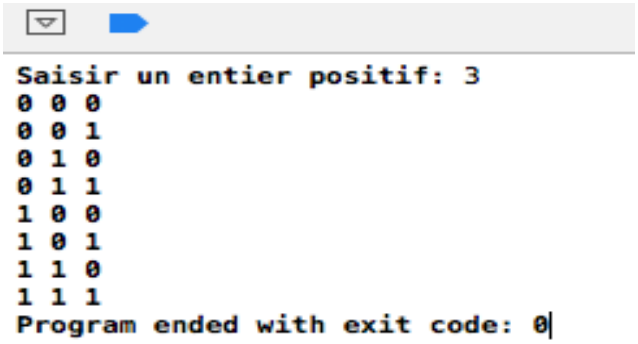
test1 :



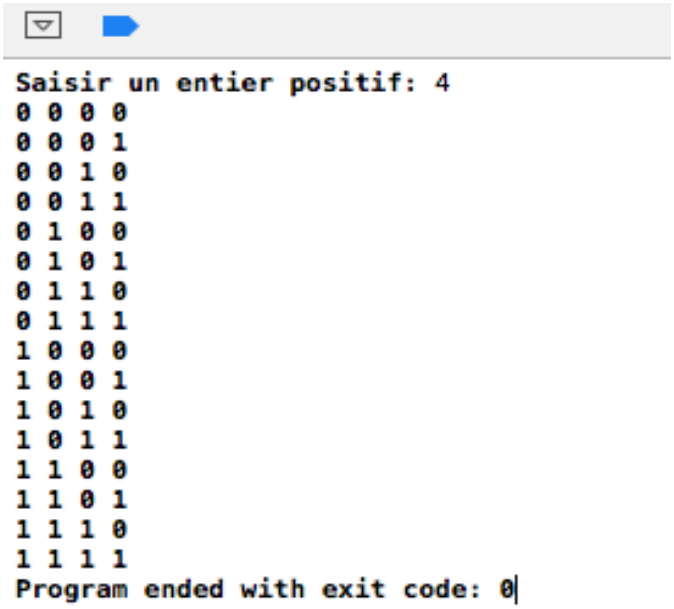
Test2 :



Test3 :



Test4 :


Saisir un entier positif: 4
0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
1 0 1 0
1 0 1 1
1 1 0 0
1 1 0 1
1 1 1 0
1 1 1 1
Program ended with exit code: 0