GNG 1106 Homework 3

Question 1:

The output of the first program will be as follows:

```
     in abc  before  increasing: x= 10
     in abc  after  increasing : x= 110
     in abc  before  increasing: x= 10
     in abc  after  increasing : x= 120
in main:   a=110, x=120
```

The first and third lines are printed because x=10 in the function abc before any action is taken. Second line is printed because the input integer (int a) is hundred, hence the math operation a=abc(x). Therefore the mathematical operation in the abc function is x=10+100 which equals 110. The abc function is executed again because of the operation x=abc(a). The input (a) from the main function is 110 from the previous line, therefore the math operation in the abc function is x=110+10, which equals 120, therefore 120 is printed in the fourth line of the output code. Last line just proves what happened and what has been described above, that is that a=110 and x=120. The abc function is executed twice because it is needed twice. Once for each value: x and a.

Question 2:

The second block of code will print the following:

```
in main: x= 100
     in abc  before  increasing: x= 10
     in abc  after  increasing : x= 110
in main: x= 110
     in abc  before  increasing: x= 110
     in abc  after  increasing : x= 220
in main: x= 220
```

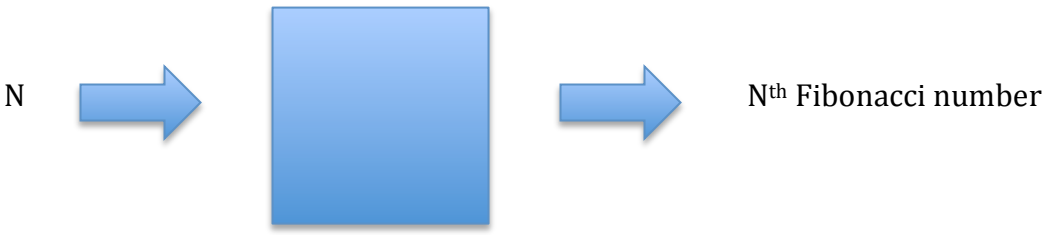
Main function first assigns the value of 100 to x and then prints it (first line of printed output). After that it assigns the value of output of function abc with the input of x (100). Although the value of 100 is assigned to a integer x in main function, the input of 100 is actually assigned to a in abc function and since x is declared to be 10 in abc function and then x=x+a, it will mean that x=10+100, which equals 110. Then in main function, the program prints that output value of x in main function as 110. Since x is a static value in abc, the third x in main function will equal the output of abc function of x (which is in abc a), which is 110. Since 110 was the last value x had in the abc function, 110 is printed as the value of x in abc; therefore the mathematical operation is x=x+a, which equals x=110+110, output as x is 220, therefore last x in main is printed as 220.

Question 3:

Problem:
Write a function with the following prototype:

int getNextFibonacciNumber(void);

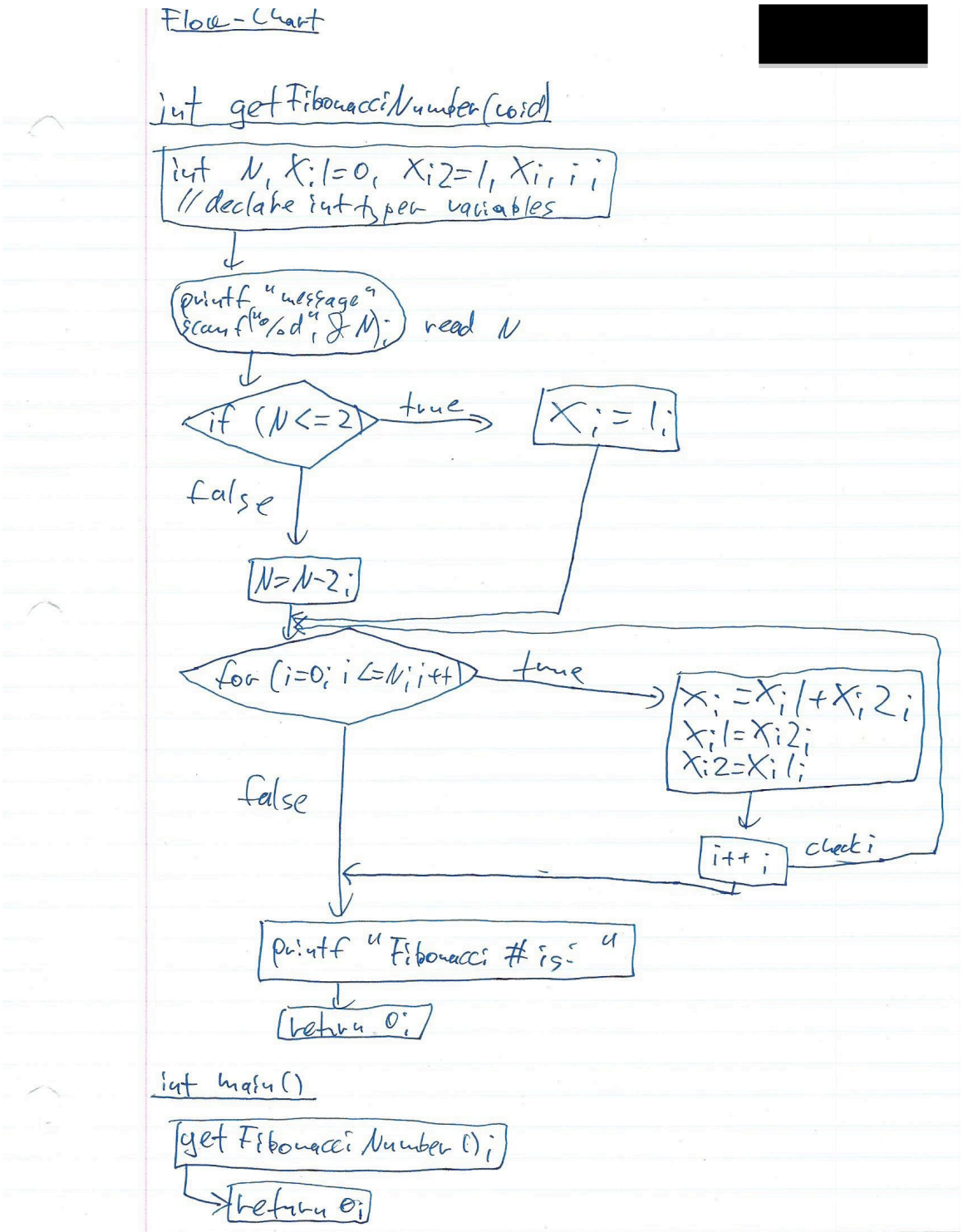
The function, when called for the i^{th} time, returns the i^{th} Fibonacci number.
I/O Description:
Input: Output:



Analysis:

The sequence 1; 1; 2; 3; 5; 8; 11; ….. is called the Fibonnacci sequence and each number in the sequence is called a Fibonacci number. That is, the i^{th} Fibonacci number X_i is related to the $(i-1)^{th}$ and $(i-2)^{th}$ Fibonacci numbers X_{i-1} and X_{i-2} by $X_i=X_{i-1}+X_{i-2}$ and the first two Fibonacci numbers X_1 and X_2 are both 1.

Flow-chart:


Flow-Chart
int getFibonacciNumber(void)
int N, Xi1=0, Xi2=1, Xi, i;
// declare int type variables
printf "message"
scanf "%d", &N;
read N
if (N<=2)
true
Xi = 1;
false
N=N-2;
for (i=0; i<=N; i++)
true
Xi = Xi1 + Xi2;
Xi1 = Xi2;
Xi2 = Xi1;
i++;
check i
false
printf "Fibonacci # is: "
return 0;
int main()
getFibonacciNumber();
return 0;

Source code for step 3 is attached in a separate .c file.

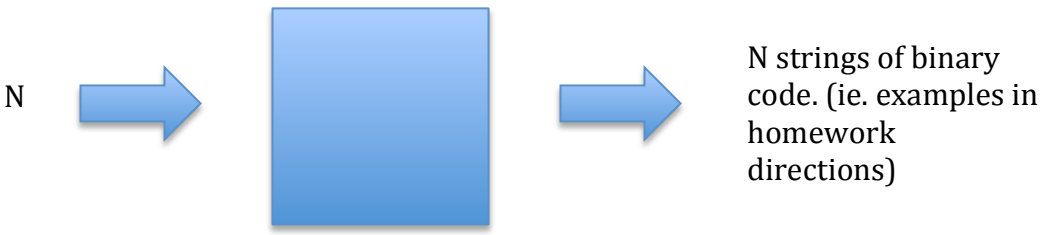
Sample outputs for the above mentioned code:

```
Which Fibonacci number would you like to know?
4
The 2th Fibonacci number is 3
[redacted]MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]/Documents/School/GNG_1106/Assignments/A3/fibonacci
Which Fibonacci number would you like to know?
5
The 3th Fibonacci number is 5
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]Documents/School/GNG_1106/Assignments/A3/fibonacci
Which Fibonacci number would you like to know?
6
The 4th Fibonacci number is 8
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]Documents/School/GNG_1106/Assignments/A3/fibonacci
Which Fibonacci number would you like to know?
7
The 5th Fibonacci number is 13
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]/Documents/School/GNG_1106/Assignments/A3/fibonacci
Which Fibonacci number would you like to know?
8
The 6th Fibonacci number is 21
[redacted]-MacBook-Pro:~ [redacted]$
```

One of the errors accompanied with this lab was Trying to figure out a way to make the first two Fibonacci numbers equal 1 and not to affect the order printed in the third line of each output. This has been figured out by subtracting 2 from N and adding the if/else loop with if prerequisite being that N is less or equal to 2, the value of 1 is assigned to the Fibonacci integer Xi.

Question 4:

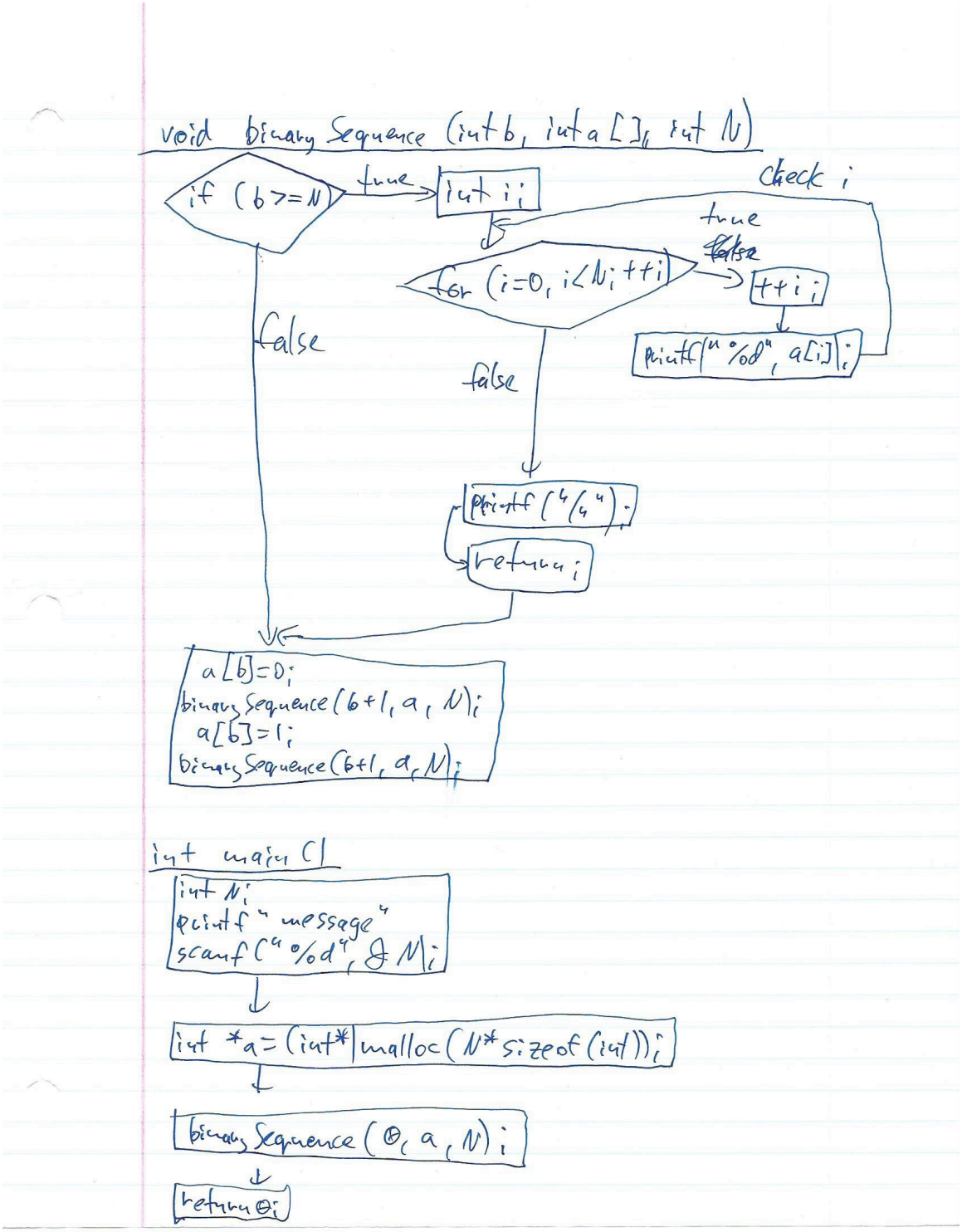
Problem:
Write a program that prompts the user to enter a positive integer N and prints the set of all binary ({0;1}-valued) strings.
I/O Description:
Input: Output:



Analysis:

Described in the Homework 3 instructions found here:
https://www.dropbox.com/sh/wterbuyxm76wwrz/AADMK49qUplFZq2io-vqCvjja/assignments%20and%20labs/assignments/homework2014_3.pdf?dl=0

Flow-chart:



Source code for step 3 is attached in a separate .c file.

Sample outputs for the above mentioned code:

```
Please enter a positive integer
1
0
1
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]/Documents/School/GNG_1106/Assignments/A3/strings
Please enter a positive integer
2
0 0
0 1
1 0
1 1
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]/Documents/School/GNG_1106/Assignments/A3/strings
Please enter a positive integer
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
```

```
1 0 1
1 1 0
1 1 1
[redacted]-MacBook-Pro:~ [redacted]$ !!
/Users/[redacted]Documents/School/GNG_1106/Assignments/A3/strings
Please enter a positive integer
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
[redacted]-MacBook-Pro:~[redacted]$ !!
```

Most errors accompanied by this lab were the ways of trying to solve the problem. First of all, an observation has been made with the example outputs from the homework instructions. The zeros and ones in the far right (last) column change every 2^0 times. In the second furthermost they change once every 2^1 times. Every column farther from the last column has an increased value of how often the values

switch, always by 2^{N+1} the N represents the exponent on 2 from the previous column (ie. 0 on the lsat one). Second approach was a bit more simple approach, trying to use nested for loops, which print the strings but only if the number of them is known (ie. if user put in 120, the program would hardly execute), yet it still was very hard to execute, finally after enough avoiding of arrays has been made, the creator of this code has decided to successfully use them to tackle this problem using arrays, memory allocation and pointers.