Name: Brian Sherif Nazmi Hanna Nasralla

Matr: 30001430

Problem 2:

a) Problem Definition:

Multiplying large integers "a" & "b" with n bits

Example:

a: 101001 = 41

b: 101010 = 42

```
      1010010
  +  101001
  + 101001
  -----------
  11010111010 = 1722
```

$41 + 41 + \dots + 41$ } 42 times

Process known as the naive approach which we learned in School.

Example from: geeksforgeeks.org

Assumption provided in problem:

~~multiplication~~ addition has a time of $\Theta(n)$

where n is number of bits

Bit shifting has a time of $\Theta(n)$

Total time: $(n(\Theta(n))) + (n(\Theta(n))) = 2n(\Theta(n)) =$

note: n represent the quantity or times function is repeated

$\Theta(n) \times \Theta(n) = \Theta(n^2)$

b) Given: Assume n to be of power 2.

Assuming our two large integers are "a" & "b"

where a l will contain all left most bits and a r will contain all right most bits

Same goes for "b l" and "b r"

In which case $a = al \times 2^{n/2} + ar$

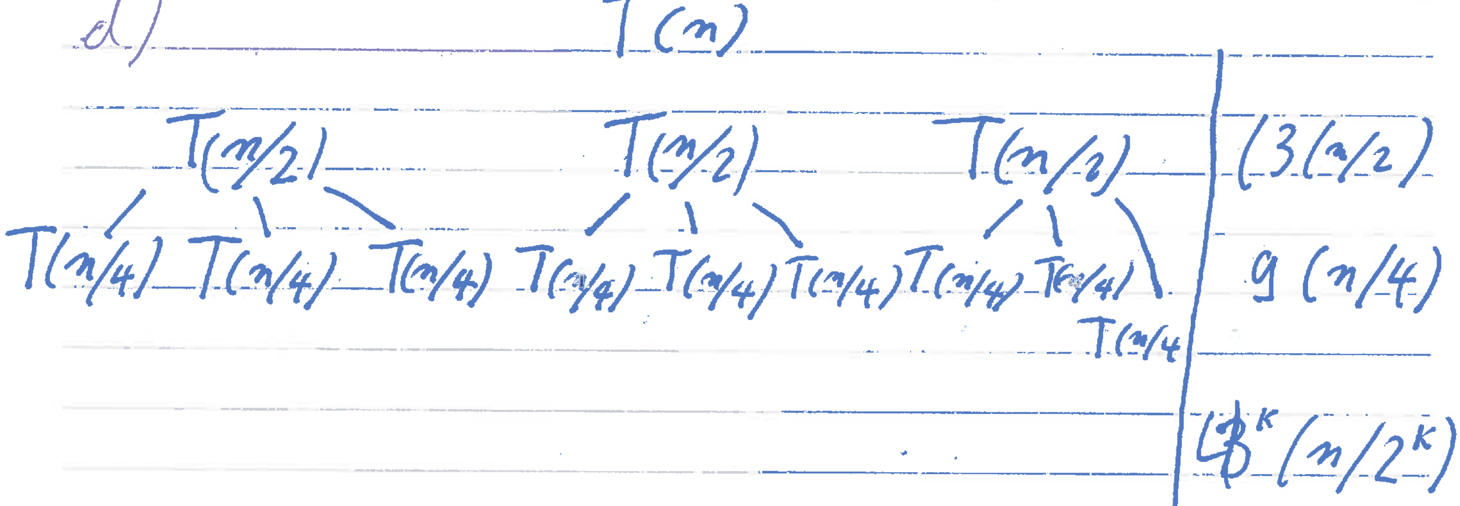
$b = bl \times 2^{n/2} + br$

$ab = (a)(b) = (al \times 2^{n/2} + ar)(bl \times 2^{n/2} + br)$

Simplify using FOIL $= 2^{n}(al \times bl) + 2^{n/2}(al \cdot br + ar \cdot bl) + (ar \cdot br)$

In this way, we have broken down the large problem into 4 multiplications of $n/2$ bits with 3 additions in between.

d)



$$3T(m/2) + \Theta(n)$$

e) using case 1 of master Theorme

$$T(n) = \Theta(n^{\log_2(3)}) = O(n^{1.585})$$

Problem 1

c) for all methods, for the same n itretwrns the same number Fibonnaci.

(~~While, all~~) While all functions produce the same result, They all take different approaches.

Example: Some functions, such as Fib-Recursive and Fib-Bottom up have a poor Space complexity and slow down as n increases.

d) After plotting all 4 functions.

We notice:

1) Recursive has the worst time complexity growing at an exponential rate.

2) closed form appears to be the fastest. as n grows.

3) but, between 1 - ~30 Matrix function is faster