Assignment - 2

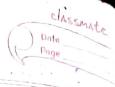
Shashank Bagda 92100133020

1 what is an Algorithm?

An algorithm is a step-by-step procedure on a set of roules for solving a specific problem or accomplishing a paroticular task. It is well-defined sequence of instrouctions that takes some input, peroforms a series of operations and produces the desired output. It can be implemented in various programming languages and used to solve rounge of problems.

- 2) Explain the need form an algorithm?
- a) Efficiency: They are designed to optimize the use of computational mesources such as
 - time & memory.

 b) Reproducibility They can be followed by anyone to achieve the same negalt
 - c) Scalability As problems grow in complexity and size, algorithms provide a way to
 - handle langers inputs and still produce results in a reasonable amount of time.
 - analyzed and tested for correctness
 ensuring that they produce the correct
 - output for all valid inputs.



1: Initialize sum = 0 2; For in marge 0 to 9 else go to 5 3: sum = sum + age(i) 4; End loop s: average = sum /10 6 : neturn avenige No of primitive operation = 11 Here areall Asymptotic Complexity is (1) 4) 1: total amount =0 2: cumpency =0 3; total amount = total amount + (2,0 x,50) 4: total amount = total amount + (1, 5 * 35) 5: total amount = total amount + (2.5 + 10) G: total amount = total amount + (1.0 + 15) 7 : amount neturned = connency - total amount 8; total items = 2.0+1,5+2,5+1.0 9: return amount returned q total items No of promitive operation = 14 Here overall Asymptotic complexity is O(1) 5) 1: factorial =1 2: in from 1 to N alse 5 3; fuctorial - factorial * 1 4: End s; neturn fuctorial. No of primitive operation = 3N+2 overall Asymptotic complexity = O(N)

	1: Sum = 0
6)	2; i in pange 1 to 100 else s
	3: sum = sum +1
	y: End
	5: return sum
	No of primitive operation = 202
	overall Asymptotic complexity = 0(1)
	OVOVE
	1: largest = number
9)	2; For i in ronge 2 to N
	3; current > langest, set langest = current
	4; End
	s: neturn largest
	No of primitive operation: 4N+2
	Asymptotic complexity = O(N)
	<u>A</u>
٧)	1: N <= 1 petron Alse
	2: Foro i in range 2 to IN
	3: N is div by 1, return fulle
	4. Endidone some sit in the
	5: petren trove
	primitive operation = 4 \(\tau \)
	Asymptotic complexity = 0 (VN)
	true at the order could be a to the
	Ach mal the first and a second the
	The state of the s



2: Foro i in roange 2 to 49 3: Fib(i) = Fib(1-1) + Fib(1-2) 4: Enl

a) 1: fib(0) = 1, fib(1) = 1

5: netron lib

Asymptotic complexity = o(N)

10) 1; binary = " "

2: while is >0, else s 3. remainder = n 1,2 4! binary = renainder + binary

5: n=n12

6: end

7: seturn binary

The algorithm of the given question is as follows

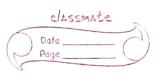
1. The algorithm starts with the first two element of the Fibonacci sequence [1]

prinitive operation = loge(1)+5 Asymptotic complexity = o(log N)

primitive operation = 98

2. It then iteratively calculates the next Fibonacii numbers by adding the last two

numbers in the sequence of append it to the sequence



	3. The algorithm performs 3 primitive operations in
	each iteration of the loop. 4. The lap rouns for 'n-2' iterations to generate
	the first so fibonacci numbers.
	5. The overall asymptotic complexity of this
	grown linearly with the input size
\	
12)	1. The algorithm uses a while loop to repeatedly
	divide the decimal number 1 by 2 and
	concatenate the remainders to the left side of
	the binary representation
	2. The loop runs until n becomes o
	3. In each iteration, the algorithm perstorms 3
-	prolimitive operations
	4. The overall asymptotic complexity of this algorithm
	is O(logn) because the number of iterations
	required to convert the decimal numbers to
	binary is logaroithmic in the size of the
	input numbers 'n'