Coral Del Mar:

# Research tittle:

This will be defined as we scope into a specific algorithm.

## Research Theme

Fundamentals of Algorithmics with Java

## Notes:

English

This Research Project entails the study of concepts in the field of algorithmics with the use of various problems, algorithmic solutions and finally their implementatation in the Java Programming language. As a final objective in this research we will study a type of problem and make and implement its algorithmic solution. **Note that the set of problems for this research is a the subset of algorithms that can be solved by a computer.**

Spanish

Esta investigación conllevara el estudio de conceptos algorítmicos con el uso de varios problemas, las soluciones algorítmicas y finalmente su implementación dentro del lenguaje Java. Como objetivo final se implementara un algoritmo de gran complejidad. **Cuando digo algoritmo me refiero a algoritmo de computadoras.**

# Topics:

* Algoritmics : The science that studies algorithms.
* Variety of computer solvable problems.

Start by Reading This :

Definiciones :

#### Algorithmics:

“Algorithmics is more than a branch of computer science. It is the core of

computer science, and, in all fairness, can be said to be relevant to most of

science, business, and technology.” [Har92, p. 6] - *Algorithmics: the Spirit of Computing*

#### Importance of Algorithms:

“A person well-trained in computer science knows how to deal with algorithms:

how to construct them, manipulate them, understand them, analyze them.

This knowledge is preparation for much more than writing good computer

programs; it is a general-purpose mental tool that will be a definite aid to

the understanding of other subjects, whether they be chemistry, linguistics,

or music, etc. The reason for this may be understood in the following way:

It has often been said that a person does not really understand something

until after teaching it to someone else. Actually, a person does not *really*

understand something until after teaching it to a *computer*, i.e., expressing

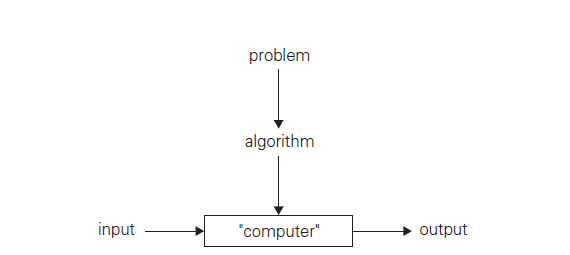
it as an algorithm . . . An attempt to formalize things as algorithms leads to

a much deeper understanding than “- Donald Knuth

#### Algorithm

“An ***algorithm*** is a sequence of unambiguous instructions for solving a problem, i.e., for obtaining a required output for any legitimate input in a finite amount of time.” - Introduction to the Design and Analysis of Algorithms 3rd Edition, Anany Levitin

* The nonambiguity requirement for each step of an algorithm cannot be compromised.
* The range of inputs for which an algorithm works has to be specified carefully.
* The same algorithm can be represented in several different ways.
* There may exist several algorithms for solving the same problem.

’

#### Computer Algorithm

Is an algorithm that can be solved by a computer via computations.

#### Problems with computer algorithms:

We will not duel deep into these because of they require some background in Computer Theory.

* The haling problem: given an algorithm and an input; Does the algorithm halts?
* The correctness problem: if the algorithm halts; Does it return the right answer?

Watch only the FIRST 5 MINUTES ONLY video :  
http://d396qusza40orc.cloudfront.net/algo1/recoded\_videos%2Fintronew%20%5Be0d4ad02%5D%20.mp4

My Suggestions:  
Study these definitions and we will begin analysing and solving problems.  
  
Problem 1:

#### Definion:

Given two nonnegave integers m and n, not--‐both--‐zero, their **greatest** **common** **divisor** is defined to be the largest integer that divides m and n (meaning that the remainder is 0 in both divisions)

#### Example:

Consider m = 60 and n = 24.

Their common divisors are: 1, 2, 3, 4, 6, and 12.

Therefore, the greatest common divisor of m and n is **12**; or, **gcd(60,** **24)** **=** **12**  
**nota: gcd() = greates common divisor**

#### Simple algorithm for computing gcd(m, n)

//Computes gcd*(m, n)* by Euclid’s algorithm

//Input: Two nonnegative, not-both-zero integers *m* and *n*

//Output: Greatest common divisor of *m* and *n*

***Step 1:*** Get the mínimum value of *n* and *m*, let’s call it “*t*”

***Step 2:*** Divide *n* by the value of *t* and if the reminder of the division is *0* then n is divisible by *t*

***Step 3:*** Divide *m* by the value of *t* and if the reminder of the division is *0* then *m* is divisible by *t*

***Step 4:*** If one of the numbers is not divisible by *t* subtract *1* from *t* and go to ***Step 2***