Engineering Method.

1. **Identification of the problem:**

1.1. Identification of needs and symptoms:

. The program must be an innovative problem solution

. The solution must bring entertaining user experience

. The solution must be well-Thought and correctly use the knowledge previously obtained.

. The software must use bring an intuitive graphical interface.

. The software must bring to the user a video game that will make him think about how to win it.

. The game must be interesting and be a defiant puzzle.

1.2. Definition of the problem:

The situation corresponding to the problem to be addressed arises from the need to use what we have learned in other situations than the curricular one, thus we can implement our programming skills in a context which not only provides a square program but also a dynamic and interactive program, As a result of this we chose as a proposal a mental agility game invented by us which will be able to give the user a glimpse of entertainment and space to reason; in this way we solve how from a program we can use algorithms to provide innovative answers to new ideas.

1.3. Functional requirements.

1. **The program must be able to show the shortest possible path to the player**
2. **The program must be able to show the player the last point in which his character was**
3. **The program must be able to show the time that the user has been in the current game**
4. **The program must be able to show the player the costs of using each platform for 5 seconds, and hide them again 3 times**
5. **The program must be able to move the player's character**
6. **The program must be able to count the player's points**
7. **The program must be able to save the player's score**
8. **The program must be able to show the best scores**

2. **Compilation of information**

   Some terms that should be considered:

   - **Graphs: An effective way to solve problems related to element's set.**
   - **Binary search Tree: Is a node-based binary tree data structure which has the following properties:**

     The left subtree of a node contains only nodes with keys lesser than the node's key.
     The right subtree of a node contains only nodes with keys greater than the node's key.
     The left and right subtree each must also be a binary search tree.
   - **Random functions: Which going to make possible generate the game.**
   - **Linked List: An efficacy way to link elements.**
   - **Holes: An obstacle in the game, making the player restart their score.**
   - **Score: A way to bring player a classification**
   - **AVL Tree: Is a self-balancing binary search tree in which the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property.**
   - **Binary Tree: Is a tree data structure in which each node has at most two children, which are referred to as the left child and the right child.**
   - **Tree data structure: Is a collection of nodes, where each node is a data structure consisting of a value and a list of references to nodes. The start of the tree is the "root node" and the reference nodes are the "children"**

3. **Search of creative solutions.**

   Bearing in mind the number of possibilities to implement the required methods, whose uses are going to bring the problem solution, we must choose a way to create the game.

   a. **The proposal to be made by our group lies in the realization of a simulation like a video game. The choice was made to implement the concepts seen in varied and entertaining fields, providing a moment of mental puzzles and relief from the rigorous presentations of conventional software programs, which allows user interaction with a puzzle; This puzzle will use the tools taught in the programming course generating a field which will have certain restrictions which will allow the game to progress:**

- The field has a traced path which will not be visible to the player, this path will be generated randomly and will use graphs to find the effective path that will take you to the goal.

- When passing through a grid of the field this will crack preventing the player from passing through that place again.

- If the player goes through a cracked path, it will fall into the void restarting the game.

- There will be obstacles around the field that will prevent the player from passing and he will have to look for alternatives to cross the field and at the exit, passing only once through each grid.

- The player may surrender if he does not feel capable of completing the puzzle, by surrendering the game will show him the most efficient way using graphs.

- Failing that, after completing the puzzle, the game will show the player which path they took and if it was the most effective,

- The player in turn is a score based on the number of movements it took to reach the goal, which will be saved in binary trees.

b. The objective of the game is to start from a starting point and reach the end by crossing squares through movements in three directions (forward, left or right), one per turn, so the player must choose where to go taking all this into account when At the end of the game, the players with the lowest scores will be those who will have the first places in a leaderboard. All this is the basis of the game since the board and the player have certain properties and restrictions explained below:

- The player starts from a pre-established starting point

- There is only one end point, and it will also be preset

- The value of each grid, determining whether it will have a numerical value or will be a hole, is given randomly.

- The player can only go through each square once and will not be able to return to a previously trodden square.

- At the beginning of the game, the player will be able to view the entire board with its holes and squares with their respective numerical values for 10 seconds.

- Once the 10 seconds have passed, the entire board will be covered, and the game will begin where the player must start his movements blindly.

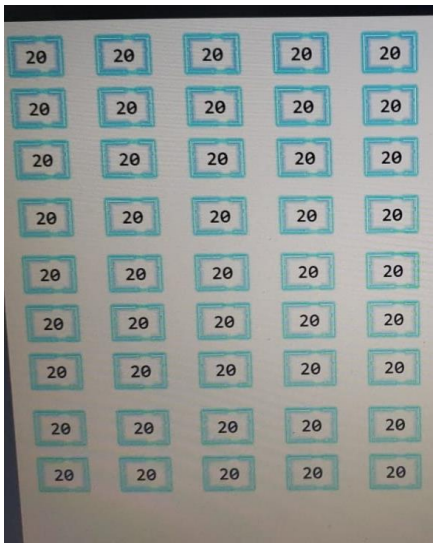
- The player has three jokers which will allow him to see the entire board for 5 seconds, after which the board will be covered again.

- Each square that is not a hole will give its numerical value to the player once stepped on by the player and this value will be added to his score.

- Each hole will send the player to the starting point by doing two things: first they will reset their number of points, however, the player will have the opportunity to see what their past score was in order to know if they will take that path again or look for another, second it will mark the last position the player had before falling into the hole to give the player some slight help.

-At the end of the game the user will be able to see his score and the time it took him to complete the game. Likewise, the game will teach you which was the shortest path, and it will also teach you which path with the lowest score you can take (best path).

-The player will have the opportunity to surrender if he does not feel capable of completing the game and the game will teach him the best way and the shortest way that is not necessarily the best always since he can have a greater number of total points.



4. **Transition from formulation of ideas to preliminary designs.**

   The "a" idea give us a few problems at the moment of think in a effective way to entertain the user, discarding de "a" idea we start to thought about implement an own videogame, with rules invented by us, so careful reviewing the "b" idea lead us to following:

   . Uses a board game will work

   . Uses a value for every square in the board added something interesting

   . Bring player a videogame created from zero, sounds more entertaining.

   So, in a nutshell:

   Using graphical interfaces and the knowledge obtained in the course, we will reach an interesting way to implement a new videogame who bring the user a puzzle to solve.

5. **Evaluation and selection of the best solution.**

   Criterion:

   - **A. Board game.**
     1. **An ice stages.**
     2. **A board of squares.**
   - **B. Graphs.**
     1. **Weights**
     2. **Simple**
   - **C. Kind of game.**
     1. **Simulation**

2. **Playable.**

D. **Wildcards.**

1. **5**
2. **3**

	A	B	C	D
1		x		
2	x		x	x

Selection.

Once picked the way to implement the game, the better option to approach our chooses to a solution is the “b” alternative.

- https://landingcode.blogspot.com/2017/06/pokemon-en-javascript-usando.html
- https://www.geeksforgeeks.org/snake-ladder-problem-2/