



**EDUCACIÓN**  
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO  
NACIONAL DE MÉXICO

Instituto tecnológico de Iztapalapa

# INSTITUTO TECNOLÓGICO DE IZTAPALAPA.

INGENIERIA EN SISTEMAS COMPUTACIONALES.

*Propuesta para el desarrollo del proyecto del titulado:*

## **LOS CONFLICTOS DE LA PROGRAMACIÓN.**

Presenta:

GUILLOT DE LEÓN DIANA GUADALUPE.

LIRA MANCERA CARLOS EMILIANO

GUITIERRES HERNANDEZ EDUARDO ANTONIO

No de control:

161080124

161080118

161080121

Asesor interno:

ABIEL TOMAS PARRA HERNANDEZ.

Ciudad de México.

Diciembre/2020



AV. Telecomunicaciones S/N, Col. Chinampac de Juárez, C.P 09208, alcaldía de Iztapalapa, Ciudad de México. Tel. 5773-8210, e-mail: [division@iztapalapa.tecnm.mx](mailto:division@iztapalapa.tecnm.mx)  
[www.tecnm.edu.mx](http://www.tecnm.edu.mx) | [www.iztapalapa.tecnm.edu.mx](http://www.iztapalapa.tecnm.edu.mx)

## Antecedentes del problema.

### ¿Qué debo hacer primero?

Para comenzar se tiene que tomar en cuenta las investigaciones y aportes previos al tópico a desarrollar ósea la problemática ya que no podemos pasar por alto aportes que han figurado como pilares dentro del rubro de la investigación y también dentro de la programación y el pensamiento analítico referente a la informática y las ramas relacionadas a esta.

Nos podemos apoyar en tesis ya antes planteadas, textos y manuales técnicos referentes al tema de la programación como lo tomaremos más adelante, también se vale citar a alguna figura sobresaliente, pero nos estamos enfocando en uno solo, ¿será preciso poder basarnos en las experiencias de empresas y o programas? Y esto depende meramente del punto de proyección que se va a tomar en cuenta ya que en este proyecto nos enfocamos y estamos del lado de una persona o grupo adeptos a la programación como a novatos y gente recién iniciad, así como también de distintas edades.

Y justo el planteamiento requiere no solo referencias modernas ya que no podemos partir sin tener las bases prefijadas como y de donde viene el problema. También son muy válidas las opiniones que no sean directamente de los profesionales ya que el público objetivo del proyecto son parte iniciados en el ámbito de la programación y más que nadie ellos conocen y han experimentado en carne propia los obstáculos y problemas que conlleva esta rama de la informática como se abordara en los temas siguientes.



### Analogía.

Muchas personas que comienzan a programar lo encuentran muy difícil. Tal vez lo que ocurre es que están tan concentrados en el lenguaje de programación que se olvidan de lo importante...

Hay mucha literatura sobre las fases de la programación, pero a efectos prácticos, cuando te pones a hacer tus primeros programas, necesitas ser consciente de estos **cuatro pasos fundamentales**:

- PASO 1) Entender bien el problema
- PASO 2) Diseñar un método para resolverlo (algoritmo)
- PASO 3) Escribir el programa en un lenguaje concreto
- PASO 4) Depuración y pruebas

Esta sencilla idea de los 4 pasos puede asustar. Puede parecer que tanta fase significa un proceso largo y costoso. Tal vez por eso muchos estudiantes intentan saltarse el paso 1 y 2 para *acabar antes*. Pero el efecto acaba siendo el contrario... ¿no te lo crees?

Repasa la lista de los 4 pasos y observa cómo cada paso depende del anterior.





Como cada paso depende del anterior, los errores en los primeros se arrastran a los siguientes, y la situación empeora:

-Si no dedicas suficiente esfuerzo a entender bien el problema (paso 1), es probable que el programa no haga lo que tiene que hacer, y todo esfuerzo posterior será inútil.

-Una vez tengas claro el problema (paso 1), si no dedicas el tiempo necesario a buscar un buen algoritmo (paso 2), te complicarás la vida. Tal vez el programa compile, pero no funcionará a la primera, ni a la segunda, ni a la tercera... y entrarás en un largo proceso de depuración y pruebas. En el caso mejor, si consigues hacerlo funcionar, habrás perdido más tiempo del que supuestamente “ahorraste” al principio.

-Puede que el programa acabe funcionando “a martillazos”, a base de prueba y error, pero eso se nota... será un programa complicado, poco eficiente, y difícil de entender y defender. Ni siquiera estarás muy seguro de que el programa funcione con según qué datos de entrada. Lo peor es que con tanto tiempo ocupado en la desagradable tarea de “probar y probar”, el proceso de programación te parecerá difícil, pesado y engorroso. En resumen, odiarás programar...



Fase	Objetivo	Herramientas útiles	Habilidades Adquirir	a Consejos
<b>1) Analizar el problema</b>	Tener una idea clara de <b>qué</b> hay que hacer	Lectura / escucha atenta. Petición de aclaraciones si es necesario.	Lectura/escucha comprensiva.	Cuando leas el enunciado no presupongas nada.  Relájate y concéntrate en leer y entender lo que significa <i>exactamente</i> .  Normalmente no basta con leer por encima y tener una idea aproximada de lo que te piden. Lee el detalle. Entiéndelo completamente.  Preguntas a hacerte: ¿Cuál es la entrada? ¿qué hay que sacar? ¿qué hay que



				hacer? ¿Cómo paso de las entradas a los resultados?
<b>2) Diseñar un método para resolver el problema (algoritmo)</b>	<p>Tener una idea clara de <b>cómo</b> va a hacer nuestro programa lo que tiene que hacer.</p> <p>Es decir, obtener un <i>método</i> (<b>algoritmo</b>) que haga lo que hay que hacer, de la forma más sencilla y eficiente posible.</p>	Pseudocódigo, diagrama de flujo.	<p>Ser capaz de descubrir el método más sencillo y eficiente posible que resuelva el problema.</p> <p>Conocer las piezas del "puzzle" con las que contamos para componer un algoritmo (variables, condiciones, bucles,...) y saber componer una solución con ellas.</p>	<p>Tú sabes resolver problemas, lo haces continuamente. La única diferencia cuando programas es que tienes que dividir el método de resolución en pasos pequeños, en instrucciones simples que pueda entender un ordenador.</p> <p>Preguntas a hacerte: ¿cómo se soluciona? ¿Qué pasos hay que dar? ¿Cuál es la forma más sencilla y eficiente?</p>



				<p>Mejora y simplifica tu algoritmo y te ahorrarás trabajo después.</p> <p>Si no te sale "de cabeza", dibújalo. El diagrama de flujo y el pseudocódigo se inventaron para ayudar al programador.</p>
<p><b>3) Escribir el problema en un lenguaje de programación</b></p>	<p><b>Traducir</b> el algoritmo a un lenguaje de programación</p> <p>El programa tiene que ser lo más sencillo y legible posible.</p>	<p>Entornos de desarrollo, Compiladores.</p> <p>Guías de estilo.</p>	<p>Conocimiento de un lenguaje de programación,</p> <p>Conocimiento de las reglas que hacen más legible un programa.</p>	<p>Intenta que el algoritmo (paso 2) esté completamente claro y simplificado antes de comenzar a escribir código.</p> <p>Cuando conozcas las instrucciones del lenguaje, este paso tiene que ser breve y limitarse a ser una traducción de tu algoritmo.</p>



			<p>En programación “avanzada”, conocimiento profundo de las particularidades del lenguaje para hacer un uso eficiente.</p>	<p>El trabajo debe centrarse más en los pasos 1 y 2.</p> <p>El algoritmo es lo importante, es “la idea que quieres expresar”. El lenguaje de programación que uses es sólo un vehículo más de expresión de tu algoritmo, como lo son los diagramas de flujo o el pseudocódigo.</p> <p>Conoce el lenguaje: Sintaxis, estructuras de datos, instrucciones ...</p> <p><u>Domina</u> las instrucciones condicionales.</p>
--	--	--	--	---





				<p>Estudia y aplica la guía de estilo.</p> <p>Haz tu programa agradable de leer y entender, con un uso adecuado del estilo y de los comentarios.</p> <p>Usa las estructuras y las instrucciones más adecuadas. Normalmente la forma más sencilla y legible de hacerlo será la mejor.</p>
<b>4) Depuración y pruebas</b>	Asegurar el buen funcionamiento del programa	Herramientas y métodos de depuración del entorno de desarrollo	Buena selección de juegos de prueba.	Intenta poner a prueba tu programa con valores normales y también con



			<p>Si se encuentran errores, habilidad deductiva para enfocar la búsqueda hacia la causa.</p>	<p>valores que creas que puedan dar problemas.</p> <p>Aprende a usar el depurador.</p> <p>Intenta como reto que en esta fase todo funcione a la primera. Pasa el esfuerzo a la fase 1 y 2.</p> <p>Importante: La depuración y pruebas no debe de ser "una forma de programar" –por prueba y error-, sino la forma de comprobar que va bien.</p>
--	--	--	---	---



### Referencia literaria.

Dentro de los escenarios relacionados con las ciencias computacionales, las actividades de aprendizaje asociadas a la programación de computadoras han sido reconocidas con alto grado de dificultad, según los antecedentes revelados en el presente artículo de revisión. Con esta situación, que al parecer es bastante común en el ámbito global, las causas que generan dicha problemática se relacionan con determinadas características que suceden dentro del aula de clase. Ciertas habilidades cognitivas son relevantes al momento del aprendizaje de los fundamentos de programación, tales como la capacidad de abstracción, una buena aptitud lógico-matemática y la facilidad para la resolución de problemas de orden algorítmico. En adición, factores de motivación son necesarios al momento de enfrentar las temáticas asociadas a los fundamentos de programación dentro de los escenarios de práctica. El contenido de esta revisión involucra experiencias en diferentes zonas del planeta, cuyo interés apunta a develar los orígenes del problema. Finalmente se elabora una reflexión en la búsqueda de posibles soluciones, y donde se abre el espacio de actuación a una nueva orientación basado en el núcleo de la Esencia de Semat.

Insuasti, J. (2016) Problemas de enseñanza y aprendizaje de los fundamentos de programación. Revista

educación y desarrollo social, 10 (2), 234-246. DOI: [org/10/18359/reds.1701](https://doi.org/10.18359/reds.1701)

Dentro de las competencias de cualquier ingeniero solicitadas por la industria, sin importar su área de especialidad, es la codificación en un lenguaje de programación. Un ingeniero debe ser capaz de codificar sus ideas ya sea para hacer experimentos y simulaciones de sus propuestas de solución, así como crear soluciones de software. El objetivo es determinar los obstáculos y clasificarlos para poder generar estrategias que faciliten el desarrollo de la competencia del desarrollo de software. El proyecto se realizó con estudiantes de segundo semestre de las ingenierías en electromecánica e ingeniería en sistemas computacionales. Se solicitó que los estudiantes describieran las diferentes problemáticas con las que se enfrentaban y cómo fueron resolviéndolas. Estas descripciones se utilizaron para determinar y clasificar los

obstáculos que se les presentaron. Es interesante descubrir que, aunque los estudiantes pertenecían a programas de ingeniería diferentes las problemáticas y obstáculos que tuvieron fueron muy similares.



AV. Telecomunicaciones S/N, Col. Chinampac de Juárez, C.P 09208, alcaldía de Iztapalapa, Ciudad de México. Tel. 5773-8210, e-mail: [division@iztapalapa.tecnm.mx](mailto:division@iztapalapa.tecnm.mx)  
[www.tecnm.edu.mx](http://www.tecnm.edu.mx) | [www.iztapalapa.tecnm.edu.mx](http://www.iztapalapa.tecnm.edu.mx)



### Habilidades para el desarrollo

La programación es parte esencial e integral de cualquier programa de ingeniería. Los estudiantes de ingeniería en sus últimos semestres tienen que enfrentar tareas de solución de problemas empleando esta competencia. El contar con buenas habilidades de desarrollo les ayudará a dar solución a estos problemas fácilmente. Es importante que los estudiantes de ingeniería y tecnología aprendan programación básica en sus primeros años de su preparación universitaria. Aprender programación no es como adquirir otro conocimiento. No es un proceso algorítmico, es decir, no es como el cálculo diferencial donde se aprende el procedimiento o fórmula se aplica repetidas veces. No es de memorización, es decir, no es como aprenderse una lista de fechas importantes y repetirlas. Para aprender a programar no basta con aprender las palabras reservadas de un lenguaje para poder aplicarlo.

Aprender a programar consiste en plasmar, mediante un lenguaje de programación, la forma de solucionar un problema. Cada problema se soluciona de manera distinta y cada programador lo resuelve de una forma diferente.

Jorge Iván Fuentes-Rosado & Melquisedec Moo-Medina

Academia de Sistemas y Animación, Instituto Tecnológico Superior Progreso, Yucatán, México. [jfuentes@itsprogreso.edu.mx](mailto:jfuentes@itsprogreso.edu.mx), [mmoo@itsprogreso.edu.mx](mailto:mmoo@itsprogreso.edu.mx)



### Hallazgos acerca del problema

En el mundo de la academia de las Ciencias Computacionales, es común observar a la programación como un arte donde la creatividad y el ingenio son factores clave del éxito. Dicho arte contempla conocimientos en lenguajes de programación y herramientas asociadas, así como habilidades cognitivas orientadas a la solución de problemas.

Desde esta óptica, un enfoque común en la enseñanza de los fundamentos de programación es abordar primero lo básico del lenguaje (o de los lenguajes) de programación, para luego guiar a los estudiantes a través de estrategias donde se contemple la totalidad del proceso de programación de computadoras. La enseñanza de los fundamentos de programación, es un punto clave en la formación de profesionales en Ciencias Computacionales. Pero existe evidencia que sugiere cómo aprender a programar no es una tarea fácil; así lo expresan Soloway y Spohrer en su libro. Estos autores afirman que la creación y el control de ambientes y soluciones computacionales a través de la programación, son cosas que para un individuo pueden ser difíciles de realizar (1989, p. 2).



## Planteamiento del problema.

*¿Por qué la programación suele ser muy complicada para la mayoría de las personas?*

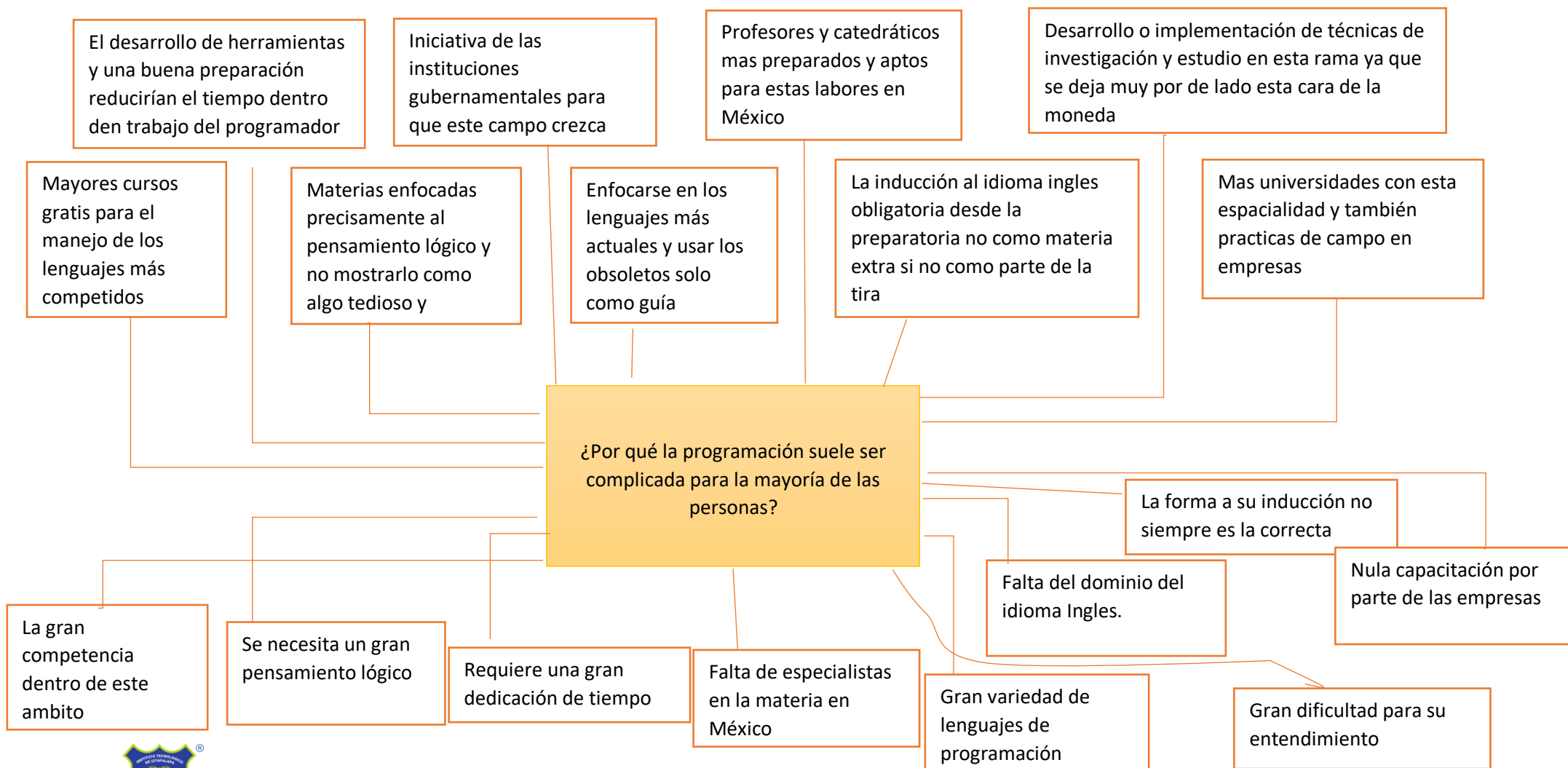
No es novedad ni alguna sorpresa que uno de los campos de la ingeniería en sistemas computacionales más importantes es la programación pero este también es uno de los más difíciles y suele ser muy complicado hasta para los especializados y personas ya egresadas del campo informático, tal vez es porque este campo tiene que tener una preparación muy complicada pasando por materias que desarrollan el pensamiento lógico matemático el cual le abre desarrolla mucho el pensamiento lógico y forma ese nivel analítico en el cual el programador no solo podrá entender la estructura de un programa sino que también podrá crear y desarrollar sus propios códigos haciendo que tenga un cuerpo y estructuras que sean de utilidad según sus requerimientos.

O tal vez sea el problema la gran variedad de lenguajes de programación que existen hoy en día ya que suena a broma, pero tenemos más de 50 y cada uno tiene sus características propias, es cierto que muchos comparten ciertas similitudes en cuanto a estructura y palabras reservadas también en cuanto a su manejo, pero no debemos olvidar que son llamados “lenguajes” por una razón esta es que son amplios y completos.

Sea cual sea el motivo o los motivos en concreto desarrollaremos un árbol de problemas con este tópico en el centro para enfocarnos a encontrar algunas soluciones. Pero también es justo decir que se tiene que plantear el problema en un cierto espacio a este lo situaremos dentro de nuestro país con todo y sus limitantes y beneficios ya que no es lo mismo hablar del desarrollo del campo de programación y desarrollo de software en la india, china o Japón que en México.

Por último, cabe destacar que esto puede llegar a tomar aspectos personales y no solo de capacidad porque entendemos que estas últimas son distintas en todas las personas así que este árbol y su desarrollo está enfocado a grupo de personas que tienen interés en este ámbito y han estado dentro ya sea de un corto a un largo periodo de tiempo.





“Aprender a escribir programas exprime tu mente, y ayuda a pensar mejor, crea una manera de pensar sobre cosas que creo es útil en todos los dominios”-Bill Gates

La programación es parte esencial e integral de cualquier programa de ingeniería. Los estudiantes de ingeniería en sus últimos semestres tienen que enfrentar tareas de solución de problemas empleando esta competencia. El contar con buenas habilidades de desarrollo les ayudará a dar solución a estos problemas fácilmente. Es importante que los estudiantes de ingeniería y tecnología aprendan programación básica en sus primeros años de su preparación universitaria [3]. Aprender programación no es como adquirir otro conocimiento. No es un proceso algorítmico, es decir, no es como el cálculo diferencial donde se aprende el procedimiento o fórmula se aplica repetidas veces. No es de memorización, es decir, no es como aprenderse una lista de fechas importantes y repetirlas. Para aprender a programar no basta con aprender las palabras reservadas de un lenguaje para poder aplicarlo. Aprender a programar consiste en plasmar, mediante un lenguaje de programación, la forma de solucionar un problema. Cada problema se soluciona de manera distinta y cada programador lo resuelve de una forma diferente. Es allí donde radica la dificultad de aprender a programar; de tener un problema y crear una solución.

Todo ingeniero debe programar. El mercado laboral exige que todo ingeniero tenga la competencia de crea aplicaciones, tal vez, no al grado de un ingeniero en software, pero sí que pueda realizar rutinas que le apoyen en su trabajo. La competencia del desarrollo de software se adquiere en los primeros semestres de las ingenierías; en ingeniería electromecánica en el primer semestre en la materia Introducción a la Programación; mientras que en ingeniería en sistemas en primer semestre en la materia Fundamentos de Programación y en Segundo Semestre en la materia Programación Orientada a Objetos. El índice de reprobación en estas materias es elevado y se desconoce la causa fundamental. Entre las propuestas que el instituto ha intentado ha sido cambiar a los maestros que imparten la clase, teniendo los mismos resultados, alto grado de reprobación. Otra de las estrategias que el instituto ha implementado son los cursos de regularización o asesorías y los resultados no han variado. Un punto importante a observar es que aquellos alumnos que aprueban no cuentan con la competencia completamente desarrollada, incluso piensan en sólo aprobar la materia sin considerar aprender a programar, Las dificultades de los estudiantes al aprender a programar pueden ser identificadas para poder generar y clasificarlas para generar estrategias que las mitiguen y se logre incrementar el





aprovechamiento en las materias relacionadas con la programación al fortalecer dicha competencia.

más desarrolladores de software, debido a que vivimos en un mundo altamente digital que sigue creciendo a pasos enormes. La demanda de aplicaciones de software dejó de ser una necesidad empresarial o industrial y se volvió parte de la vida diaria. Retos actuales como Internet de las Cosas hace que surjan nuevas necesidades de aplicaciones. Áreas como la salud requiere de software especializado, no sólo para la administración, sino aplicaciones que controlan los equipos de alta tecnología que realizan diagnósticos más oportunos o trasplantes electrónicos que dan una mayor calidad de vida a sus portadores. México requiere capacitar a más gente en disciplinas como el Desarrollo de Software e ingenierías tecnológicas para crear la tecnología que el país y el mundo necesita y convertirse es un productor de esta tecnología y no meramente consumidor.

Programar es una habilidad muy útil y puede ser una recompensante carrera. Programadores novatos sufren de un amplio rango de dificultades y deficiencias. Los estudiantes deben descubrir la necesidad de seguir un algoritmo para poder solucionar problemas. Es importante entender qué hay que hacer, establecer un plan de acción, establecer las pruebas de validación, codificar y por último hacer las pruebas. Los estudiantes no realizaban el diagrama de flujo porque no reconocen la importancia del diseño o el plan de acción. Los estudiantes directamente intentaban codificar sin haber entendido el problema o pensando en la solución. Es importante que el facilitador de la materia motive a los estudiantes a realizar los diseños previos al intento de codificación.

Algunos de los problemas que más resaltan son:

Fobia a los problemas “complejos”: “La fórmula es muy complicada de hacer. No he estudiado cálculo integral.”. El estudiante antes de intentar solucionar el problema, en su primera percepción del problema, si ve algo que lo impacte, tiende a no solucionarlo. Se debe enseñar al estudiante de una manera práctica la estrategia divide y vencerás, la cual consiste en dividir el problema en pequeños subproblemas que pueda solucionar de una manera más sencilla.

- Lógica incompleta: Al no lograr dividir el problema en subproblemas no consiguen establecer los pasos necesarios para llegar a una solución completa. Los facilitadores podrían favorecer la lógica al sugerir la solución de acertijos y permitir que los estudiantes solucionen con tus recursos los problemas





- Desconocer el lenguaje: “No sé cómo leer archivos” Los estudiantes entienden el problema, saber cómo solucionarlo, pero no logran codificarlo debido a que ignoran las palabras reservadas o librerías del lenguaje que podrían aplicar para realizar el software correspondiente. A manera de estrategia, los estudiantes podrían crear sus propias guías de bolsillo con las palabras reservadas y funciones que emplean.

- Desconocer las herramientas del entorno de desarrollo (IDE): “No corre como yo esperaba y no pude encontrar el error”. Los estudiantes escribían sus códigos, pero no lograban corregir los errores de lógica que se les presentaban en el código. Los errores de sintaxis eran eliminados por el IDE, es decir, no hubo errores tales como olvidar los puntos y comas, la falta de declaración de variables, o los tipos de variables. Si los estudiantes supieran como realizar un debuggeo, correr el programa paso a paso empleando el entorno de programación, hubieran podido encontrar los errores faltantes.

- Falta de motivación: “No sé hacerlo”. Aunque los estudiantes reconocen que el desarrollo de software es importante, hay quienes expresan que no tiene el gusto en realizarlo. El estudiante debe reconocer que una de las ventajas de programar es el desarrollo profesional. Un estudiante desmotivado no realizará las prácticas, y lamentablemente para aprender a programar hay que programar.

- Administración del tiempo: “No me alcanzó el tiempo”. Los estudiantes dentro de sus comentarios expresaron que el tiempo no fue suficiente para resolver el examen. Para un trabajo futuro se considerará dar más tiempo para la solución. La programación en el caso de sistemas computacionales está ligada con las materias siguientes: Programación Orientada a Objetos, Métodos Numéricos, Simulación, Estructura de Datos, Tópicos avanzados de programación, bases de datos, ingeniería de software.

Si el estudiante no desarrolla las competencias debidas se verá afectado en el aprovechamiento de las materias subsecuentes. La ingeniería en electromecánica en el plan de estudios actual no cuenta con ligaduras para la materia de fundamentos de programación, además el autor sugiere más cursos de programación para esta ingeniería, para que los estudiantes logren desarrollar la lógica requerida.



## Objetivos.

Nuestro objetivo principal sería llegar al fondo de la problemática.

### ¿Por qué?

Principalmente nos enfocamos al desarrollo de las habilidades lógicas individuales las cuales no siempre han sido desarrolladas de la misma manera y parecerá chiste pero ejercicios o cosas que requieren un pensamiento lógico en toda la extensión de la palabra puede llegar a parecer casi imposible de analizar por algunas personas y aquí es donde partimos hacia atrás y nos damos cuenta que definitivamente no todos recibimos la misma preparación en estos ámbitos aquí influyen tanto aspectos sociales hasta personales desde pensamiento propio o cuestiones ajenas a nuestro control como niños o jóvenes.

### ¿Quién o quiénes?

Actualmente conocemos ya de experiencia los estudiantes, ingenieros, licenciados, etc. Que existe un problema principalmente en nuestro país no solo de falta de conocimientos si no una gran falta de interés por ciertos campos de la ingeniería en sistemas computacionales, pero según algunos testimonios también nos arrojan unos datos interesantes acerca de la difícil tarea y las responsabilidades que lleva el ser un programador.

“La media no cumple con los aspectos y requisitos para este puesto” es lo que mas de una vez llegue a escuchar de algún especialista y en general se tiene entendido que es un puesto el cual absorbe gran parte del tiempo de una persona a lo cual este problema conlleva una gran presión social y tenemos en cuenta la sociedad moderna donde ya esta muy normalizado el fracasar de manera FRECUENTE y no dedicarle el tiempo justo y necesario al aprendizaje.

### ¿Cómo, cuándo y dónde?

Podemos decir que el “como” también entra en el porqué, pero ¿de qué forma se define este problema? En lo personal a base de la investigación en el campo y ambiente que me rodea (universidad) muchos compañeros reniegan de la programación como especialidad gracias a testimonios, anécdotas o





experiencias de conocidos que se dedican de lleno a ser programadores teniéndolos en un mal concepto como personas socialmente ineficientes y que dedican todo su tiempo a eso, pero ¿en realidad es de esa manera? Yo al igual que todos ellos he escuchado historias de terceros y experiencias de conocidos, pero nunca me he tomado mi tiempo para poder decir si en verdad te toma todo un día completo o hasta meses el desarrollar un código para desempeñar una función pero por lo poco que tengo de experiencia en la universidad pienso que el problema radica en que no me instruí ni tuve ningún contacto con este campo antes ni en secundaria ni en preparatoria y tomando como ejemplo compañeros que son buenos programando y conociendo un poco de su historial resulta que ellos ya habían tenido aunque sea una introducción en este campo y aquí es donde llegamos a una pista LA EXPERIENCIA, ¿acaso será la falta de experiencia? Si es así al menos por ahora en este análisis podemos contemplar que vamos a requerir de tiempo y dedicación para forjarnos nuestra propia acumulación de conocimientos y experiencia. Esto también conlleva otros aspectos tanto como la disciplina y principalmente el interés nada de otro mundo.

Comparando el desarrollo de este campo en México podemos concretar que es muy pobre a comparación de otros países que son estandarte en cuanto a la programación y el desarrollo de software en general como lo pueden ser la India o China siendo este último gran impulsor de no solo aplicaciones móviles sino también de sistemas operativos completamente funcionales y eficientes capaces de competir con los gigantes de hoy en día en ese ámbito así como también el desarrollo de plataformas y herramientas virtuales para distintos tipos de campos, entonces también entraría la cuestión de la educación en general en nuestro país la cual se ve muy obsoleta a comparación de otros países a falta de un modelo que se renueve constantemente que cabe aclarar el actual no es malo del todo ya que existen grandes profesionistas que son ejemplo y han logrado resaltar no solo en el campo de la programación si no que en general. Estos personajes suelen por lo general buscar oportunidades en el extranjero gracias a la falta de apoyo y de interés en México y eso no es novedad, pero en este aspecto entran demasiados aspectos sociales de los cuales podríamos desglosar una gran cantidad de problemas y posibilidades así que solo se está tomando muy por encima.



## ***Justificación.***

Que puede ser una solución para este problema pues sinceramente existen demasiados, algunos parecen utopías y otros no lo son, pero están muy lejos y fuera del alcance así que pensamos en el desarrollo de un entorno virtual que tiene por fin principal englobar las técnicas de programación en general

¿Qué tiene de diferente?

Lo que puede parecer trivial ya que es integrar un nuevo modelo ya a los existentes pero el fin de este es poder ser en realidad fácil de manejar y aprender en toda la extensión de la palabra sin limitaciones de lenguaje ya que este tendrá un concepto de globalización, pero esto es a largo plazo. De primera se tiene como un proyecto para el apoyo de aprendizaje en segundo plano una también apto para el trabajo y que pueda aplicarse dentro de la función laboral y empresarial y por último a largo plazo ser mundialmente utilizada con un estudio de mercado como respaldo para así poder contemplar problemas o aspectos que pueden llegar a pasar desapercibidos en México.

Esto atraerá muchos beneficios en general se englobará a todos los programadores en un lenguaje que estará disponible en todos los idiomas esto en cuanto a palabras reservadas e interfaces ya que los principios lógicos y matemáticos son universales es algo que por competencia o tiempo las grandes empresas dejan muy de lado ya que tenemos que aceptar que esto sigue siendo más una competencia dentro del mercado y si esto puede parecer una solución temporal ya que este mismo obstáculo es el mayor por ahora. Se buscará llegar a una especie de acuerdo con grandes como Oracle para minimizar este impacto. Un gran ejemplo de que esto es posible tenemos en cuenta como se han globalizado los videojuegos y la manera en cómo ha evolucionado la interacción de nosotros con personas de todo el mundo enfocándonos en el concepto de Cross play entre consolas de todo el mundo ya que es una especie de compatibilidad entre juego de consolas de empresas diferentes para así tener una experiencia más estable dejando de un lado la competencia para tener un beneficio en común sin afectar de manera mutua la estabilidad del mercado y así haciendo más FÁCILES los modos de comunicación que ya estaban presentes pero sin pulir que también queda resaltar que esto un esta correctamente perfeccionado del todo falta mucho que implementar y a si es el rumbo que tiene el proyecto ya que eso es lo que se busca facilitar el aprendizaje y la aplicación.



## ***Cronograma de actividades.***

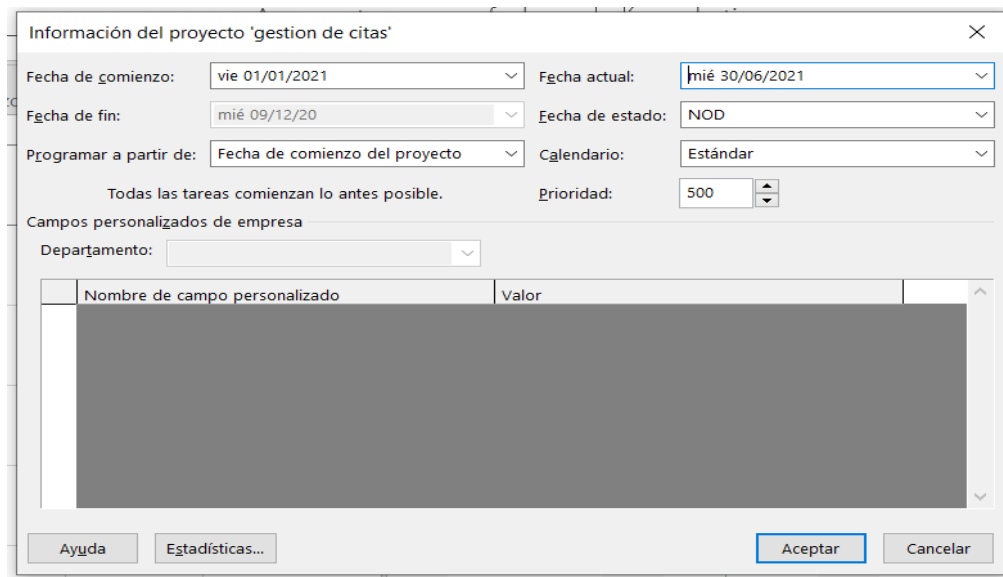
### *Estimación de tiempos.*

Definición de tareas:

investigación de los antecedentes de la problemática.  
Planificación sobre herramientas y personal.  
Desarrollo del programa base e interfaz  
Testeo e identificación de errores  
Corrección de errores y tiempo de pruebas  
Implementación del software.

### *Estimación de duración de tareas.*

Plazo.



Información del proyecto 'gestion de citas'

Fecha de comienzo: vie 01/01/2021 Fecha actual: mié 30/06/2021

Fecha de fin: mié 09/12/20 Fecha de estado: NOD

Programar a partir de: Fecha de comienzo del proyecto Calendario: Estándar

Todas las tareas comienzan lo antes posible. Prioridad: 500

Campos personalizados de empresa

Departamento:

Nombre de campo personalizado	Valor
-------------------------------	-------

Ayuda Estadísticas... Aceptar Cancelar







**Cambiar calendario laboral**

Para calendario: **proyecto** Crear calendario...

El calendario 'proyecto' es un calendario

**Legenda:**

- ☐ Laborable
- ☐ No laborable
- 31** Horas laborables modificadas

En este calendario:

- 31** Día de excepción
- 31** Semana laboral no predeterminada

Haga clic en un día para ver sus periodos laborables: **Periodos laborables del 09 diciembre 2020:**

- 9:00 a 13:00
- 15:00 a 19:00

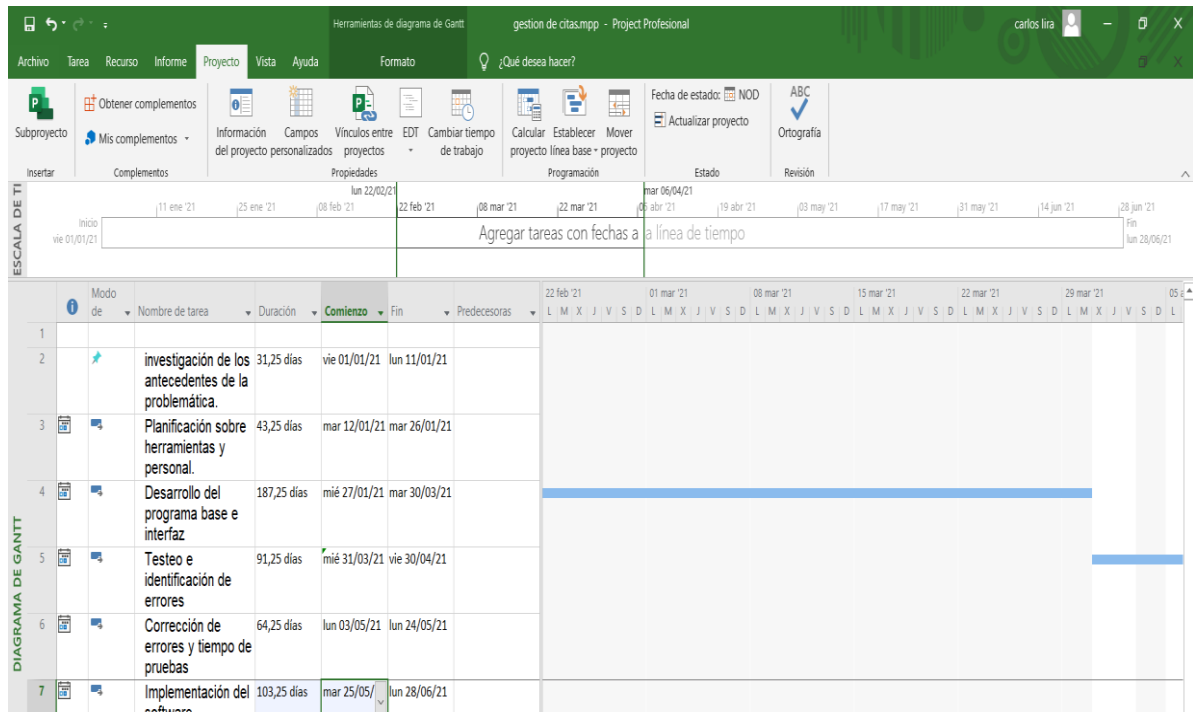
Basado en:  
Semana laboral predeterminada del calendario 'proyecto'.

**Excepciones** **Semanas laborales**

Nombre	Comienzo	Fin
año nuevo	01/01/2021	02/01/2021
reyes magos	06/01/2021	06/01/2021
semana santa	01/04/2021	04/04/2021
día de las madres	10/05/2021	10/05/2021

Ayuda Opciones... Aceptar Cancelar

Tiempos y fechas estimadas.



AV. Telecomunicaciones S/N, Col. Chinampac de Juárez, C.P 09208, alcaldía de Iztapalapa, Ciudad de México. Tel. 5773-8210, e-mail: [division@iztapalapa.tecnm.mx](mailto:division@iztapalapa.tecnm.mx)  
[www.tecnm.edu.mx](http://www.tecnm.edu.mx) | [www.iztapalapa.tecnm.edu.mx](http://www.iztapalapa.tecnm.edu.mx)



### Estimación de costos.



### Recursos.

	Nombre del recurso	Tipo	Texto1	Iniciales	Grupo	Capacidad	Tasa	Tasa horas	Costo/Usr	Acumula	Calendario	Códi	lgregar nueva column
1	Analistas	Trabajo	3 personas	A	Analistas	300%	0,00 €/hora	11,00 €/hora	0,00 €	Prorratio	proyecto		
2	Programadores	Trabajo	3 personas	P	Programaci	300%	0,00 €/hora	15,00 €/hora	0,00 €	Prorratio	proyecto		
3	diseñadores	Trabajo	2 personas	d	Diseño	200%	0,00 €/hora	12,00 €/hora	0,00 €	Prorratio	proyecto		
4	testers	Trabajo	2 personas	t	Testeo	200%	0,00 €/hora	14,00 €/hora	0,00 €	Prorratio	proyecto		
5	instalaciones	Material	mensual	i	area		100,00 €		500,00 €	Comienzo			
6	equipo computo	Material	mensual	e	material		50,00 €		200,00 €	Prorratio			







## Salarios.

	Nombre del recurso	Tipo	Texto1	Iniciales	Grupo	Capacidad	Tasa	Tasa horas	Costo/Usr	Acumula	Calendario	Códi	lgregar nueva column
1	Analistas	Trabajo	3 personas	A	Analistas	300%	0,00 €/hora	11,00 €/hora	0,00 €	Prorratio	proyecto		
2	Programadores	Trabajo	3 personas	P	Programaci	300%	0,00 €/hora	15,00 €/hora	0,00 €	Prorratio	proyecto		
3	diseñadores	Trabajo	2 personas	d	Diseño	200%	0,00 €/hora	12,00 €/hora	0,00 €	Prorratio	proyecto		
4	testers	Trabajo	2 personas	t	Testeo	200%	0,00 €/hora	14,00 €/hora	0,00 €	Prorratio	proyecto		

