

Todos los programadores visten de negro

Pérez Gaytán Ángel Gabriel

PRÓLOGO

Un mejor mañana. Eso es algo que añoramos todos. Un día en el que la vida sea mas amable con nosotros. Un minuto en el que la felicidad que parece rodear nuestro cuello no acabe nunca. Un sentimiento, una emoción, un impulso para poder lograr lo que parece imposible. Quizás cuando Turing ideo su maquina, pensó en ese mejor mañana. Cuando Von Neumann consumo su arquitectura, también lo pudo haber imaginado Imagina ser Ada Augustus Lovelace, imagina poder tener el cerebro y la capacidad de ver lo que tu sociedad no ve, lo que a los ojos de los demás seres humanos son solo fantasías lo que en el fondo de tu alma también sientes que lo son, imagina lo imposible.

Imagina lo extraordinario.Cuando el ser humano a lo largo de toda su historia pensó en como sobrevivir jamas se imagino que usaría aquellos números que en esencia son solo símbolos, son solo visiones de una dimensión completamente alejada de la nuestra y a su vez aquella de la que nosotros hemos sido sus esclavos. El tiempo. El título hace referencia en parte a una broma que realicé con un amigo de la facultad acerca de que la manera de vestir de varios programadores (incluyéndome), consta únicamente de prendas negras.

Conectando esa broma con el contenido del texto, es una perfecta oportunidad para tratar de hablar de un problema de la ciencia que la misma disciplina ha hecho autocítica y ese es el exceso de pensamiento abstracto y una aparente ausencia de criterio político o filosófico dentro de la sociedad actual. Y aunque los programadores y matemáticos han buscado solucionar problemas complejos, muchas veces sus resultados no han sido aplicables a las sociedades que tanto han intentado apoyar. El futuro, es un concepto único, un deseo y una fantasía del cual como seres humanos deseamos formar parte. Sin embargo creo que también es necesario pensar en el presente de la computación, de los alumnos que formaran parte de la nueva generación de programadores. Quiero creer que este trabajo puede marcar un antes y un después en mi vida, en lo que aspiro a conocer, a aquel conocimiento que aun no poseo pero deseo hacer mio, que aquella chispa de locura que puede hacernos soñar también la posea.

No puedo decir que soy como el gran Raman al haber tenido sueños proféticos de las matemática. Pero si de mi dependiera que aquella persona detrás de un monitor, de un libro impreso pueda soñar con la existencia de cosas imposibles, de maquinas que solo la mente humana sea capaz de entender, puedo morir tranquilo esta noche.

Trabajo de investigación.... He hecho muchos de esos en mi vida. Resúmenes, síntesis, cosas aburridas que a la larga no significan nada.

No quiero eso con este texto. Quiero que cuando tu lo leas recuerdes por lo menos una frase. Quiero que sea recomendado. Añoro que sea trascendente. Que signifique un antes y un después para ti también. Todos los programadores visten de negro es una carta de un loco que desea trascendencia en su vida.Todos los programadores visten de negro es una nueva forma de leer ciencias. Todos los programadores visten de negro en su base es una plegaria para aquellos que no nacemos siendo genios. Todos los programadores visten de negro es una invitación a creer en lo imposible de la computación.

Todos los programadores visten de negro es un texto que no querrás olvidar...

PARTE 1: HISTORICAMENTE ¿QUÉ ES UN PROGRAMADOR?

Un computólogo gracias a compañías ”modernas” como Google o Amazon ha adquirido una imagen más pulcra comparada con aquella que podía haber llegado a tener antaño. Pese a ello, es común que la gente cuando se le pregunta acerca del como cree que se ve alguien que ejerce esta profesión lo primero que responda es que cumple con algunos de las siguientes características:

- Tiene mala higiene
- No tiene vida social
- Usa lentes
- Es una persona que UNICAMENTE sabe de computadoras



En realidad ese es un estereotipo y muchas veces incluso como programadores compramos ese papel.

Podemos empezar esta sección definiendo que hace un computólogo. Un computólogo no solo se encarga de programar, sino que también tiene la tarea de encontrar la

solución a diversos problemas usando una computadora como HERRAMIENTA. Se hace énfasis en esto ya que cuando uno solicita el acceso a la carrera se comete el error de asumir que solo veremos formas de programar, cuando en realidad lo que se aborda son un conjunto variado de bases, fundamentos tanto teóricos y prácticos de diferentes disciplinas científicas.

Materias y contenido tales como matemáticas aplicadas o cálculo son necesarias para luego entender bases de diseño en niveles más avanzados. Ahora bien se dice que la computadora es una herramienta debido a que la solución de un determinado problema no debe de surgir únicamente de la linea de comandos, sino, debe de ser el resultado de un correcto análisis de la situación diseño y una correcta implementación en el ordenador.

Pero **¿Como se llegó a este punto en la historia de esta ciencia?**

Esta sección se encarga de abordar algunos hechos de carácter histórico que han dado forma a la computación tal como la conocemos hoy.

PROGRAMA Y NO CUESTIONES

Planteamiento inicial

Cuando me preguntaron acerca del propósito de mi escrito no tuve una respuesta clara o mucho menos coherente. Y es que al final de cuentas, seguimos hablando de un texto cuyo fin es explicar algunos de los temas básicos de la materia de Introducción a la Ciencia de la Computación. Cosas como historia de la computación, o un acercamiento a la lógica de la programación orientada a objetos son lo esperado en la entrega final.

Quiero citar a una persona con la que estudio la misma materia de donde surge la necesidad de escribir esto: La ventaja de este trabajo es que nos dieron demasiada libertad. La desventaja de este trabajo es que nos dieron demasiada libertad. Abordar un universo tan amplio como la historia de la computación, o la lógica necesaria para entender como programar en JAVA no es una tarea sencilla, mas no es una imposible. Si capte la atención del lector durante el prologo, ya podemos ser un poco mas específicos con nuestra pregunta detonante:

¿Como convenceríamos a una persona de ingresar/quedarse en la carrera de ciencias de la computación?

Hipótesis

Dentro del dominio de los trabajos de investigación universitaria, soy consciente de que un lenguaje carente de la formalidad necesaria, no tiene cabida en una institución como lo es la Universidad Nacional Autónoma de Mexico y aunque históricamente el mantener una linea de formalidad ha permitido que nuestra universidad se mantenga como uno de los mayores exponentes de la educación a nivel superior en toda Latinoamérica desde mi perspectiva ha creado también una brecha entre las viejas y nuevas generaciones de maestros y alumnos de la facultad de ciencias.

Aunque vaya a ser polémico lo que vaya a decir y eso es que aunque son bastante útiles los libros introductorios de Canek o Gaona a la materia, son libros que desde un aspecto social o filosófico, no motivan a una persona común y corriente como yo a leer textos de programación. Puede que en apariencia no importe esto, puesto que el enfoque de la carrera es poder dar los conocimientos necesarios para saber programar a nivel competitivo. Como facultad estamos descuidando a un porcentaje (Que aun desconozco que tan alto sea) de personas que deciden abandonar sus estudios y preparación académica debido a la carencia de motivación por parte del sistema que

supuestamente busca crear a la siguiente generación de investigadores, catedráticos y profesionistas.

Experimento

Pero esas solo eran ideas mías que yo sentía que eran correctas. Es por ello que decidí hacer una encuesta a algunos alumnos de la misma carrera para conocer su punto de vista de la situación, por lo que a continuación adjunto el link a un cuestionario de *Google Forms* con el que quise verificar mis visiones:

<https://forms.gle/SVM58os2SmPVtgw9>

Resultados

Al momento de escribir esta parte del texto, tengo alrededor de 14 personas que han respondido el cuestionario en torno a la situación emocional que he tratado de analizar y honestamente me sentí sorprendido con los resultados, puesto que aunque esperaba una comunidad enojada, resentida o por lo menos un poco disconforme con su carrera me di cuenta de algo: EL LOCO AQUI SOY YO! Y es que al parecer hay bastante aceptación por parte de los alumnos de la posición en la que se encuentran y es que la mayoría al responder mis preguntas me dieron mucho en lo que pensar, sobre todo como expondré los temas post análisis.

Análisis

El cuestionario se enfocaba principalmente a la exploración de la salud emocional o estado anímico de mis allegados académicos, es por ello que algunas preguntas que considere necesario realizar fueron:

- ¿Porqué entraste a Ciencias de la computación?
- ¿Sientes que la facultad se preocupa por la salud mental de sus alumnos?
- ¿Has pensado en abandonar tus estudios?
- ¿Recomendarías a alguien la carrera de Ciencias de la computación?
- ¿Crees que los libros que ocupamos en la carrera te dan un incentivo EMOCIONAL para poder aprender programación?
- ¿Has leído algún libro de prensa de ciencias que te haya dejado alguna enseñanza a nivel PERSONAL?

Claro que es temprano para asumir que hay un grado de conformidad absoluto con relación a los estudios superiores de C.C, sin embargo es menester reflexionar un poco en el hecho de que con relación a la pregunta 5, mas de la mitad de lo encuestados nunca se había preguntado si sus materiales de estudio podían contener un incentivo emocional para poder aprender a programar, o en el caso de la pregunta 6, el 80% de los encuestados hasta el momento no han adquirido una enseñanza a nivel personal del conjunto de libros disponibles de programación, vaya, ni siquiera lo habían preguntado. Dentro del cuestionario hubieron en particular 2 preguntas que captaron mi atención, la pregunta 4, aquella que plantea si se recomendaría estudiar esta carrera y otra que manifiesta:

- ¿Te gustaría que hubieran más material enfocado a la filosofía de la programación en la facultad?

En el caso del recomendar la carrera, mas del 85% de los encuestados respondieron: Depende de si les gustan las matemáticas Por el otro lado con relación a la pregunta 7, 64% los encuestados si quisieran ver mas material enfocado a la filosofía de la programación, sin embargo un 21% no quisiera ver mas contenido relacionado a este tema.

Reflexión post-análisis

Para aquellos que no sean programadores y sean un poco como quien escribió este trabajo, tal vez parezca un poco o quizás bastante desalentador el analizar fríamente si un enfoque mas orientado a la historia o a la filosofía del porque programar es realmente viable. Honestamente yo si me sentiría desalentado. Creo que hay que pensar que aunque por el momento la computación es una ciencia relativamente nueva y en realidad hemos avanzado mucho en muy poco tiempo desde la ENIAC hasta la computación cuántica, considero que ya es tiempo de cuestionarnos la importancia de insertar código en una computadora mas allá de la frase: "*Para hacernos la vida mas fácil*".

Sería un poco inoportuno esperar que la programación no busque facilitar o agilizar procesos que tan solo hace una década parecían ser necesarios de ejecutar por un ser humano. Es en el calor del surgimiento de inteligencias artificiales tales como *Chat gpt* que considero una necesidad para la nueva generación de programadores y estudiantes interesados en este mundo en re-evaluar conceptos tales como lo que es un programador, que es el paradigma orientado a objetos, que hace relevante documentar todo el progreso histórico que hemos tenido para poder tener internet, laptops o

cualesquiera artilugios de computo.

Desde tener un celular, hasta contar con programas de animación digital la computación no es algo meramente matemático, es una rama que involucra a todas las áreas del conocimiento, desde la medicina hasta la pedagogía. En las siguientes paginas abordaremos nuevas preguntas y vamos a redefinir algunos conceptos para que si estas leyendo esto, no abandones la idea de ser un programador. Pero mas importante, lo que yo te quiero invitar es a cuestionar lo siguiente:

¿Qué tipo de programador quieres ser?

Breve historia de las computadoras... antes de que existieran monitores, teclados, o tan siquiera un enchufe toma corriente

Si algo he aprendido en el poco tiempo en el que me he visto envuelto en el mundo de la computación es que en realidad el aparato que nosotros conocemos como "PC" es un aparato que aunque ya paso su primera infancia sigue siendo relativamente joven y aun queda mucho por explorar en su arquitectura, implementación, etc.... Se que suena a una frase que no tiene ni pies ni cabeza, **¿Como va a ser posible que se hayan realizado avances de tal magnitud en la historia de la humanidad sin una computadora?**

Decir que la explicación es simple seria mentir, seria insultar la inteligencia de mis lectores, puesto que en esencia, la historia de la computación es una que va de la mano con la matemática. Es quizás aquel hijo de las ciencias exactas que si bien no ha logrado igualar a su progenitora al poder calcular conjuntos infinitos de " x " o " y " magnitud, ha logrado emular y simplificarmuchos de los procesos que antes hubiera requerido meses e incluso décadas en realizar. Para poder comprender de mejor manera dicha linea del tiempo, me tomare la libertad de realizar una síntesis de las fechas mas trascendentales que aparecen en el libro **"A history of computing technology"** escrito por Michael R. Williams.

Ciitare algunos hechos relacionados con la ética de esta época que se encuentran disponibles en el libro **"Del ábaco a la revolución digital"** de la prensa de National Geographic.

Nosotros empezamos esta historia en el periodo Antes de nuestra era, en el que durante siglos nuestros antepasados no solo con el afán de sobrevivir sino con un deseo humano de conocer mas allá de lo que sus ojos calculan decidieron empezar a contar objetos, clasificarlos, darles un determinado valor. Civilizaciones como las babilónicas o egipcias son algunas de los grupos de humanos mas antiguos en haber registrado sistemas numéricos. También es importante no olvidar los aportes a la matemática otorgados por los griegos y romanos.No olvidemos que estamos ante un texto de carácter formativo, por lo que seria irresponsable de mi parte no mencionar dichos aportes de una forma mas concisa como lo seria una tabla.

Civilizaciones antiguas	Aportes a la matemática
Babilonia	Sistema numérico sexagesimal desarrollo en el pensamiento algebraico y geométrico
Egipcia	Registro mas antiguo del numero Pi
Griega	Refinamiento de pi e implementación de los números primos
Romana	Cuestionamiento moral del progreso en la matemática

Algo interesante es que no son pocos los que afirmamos que el numero cero es una herencia de la civilización maya, ahora bien eso no significa que hubieran personas en otros rincones del mundo que buscaran tener una aproximación a cuando no tenemos elementos, ya que existen también registros de la cultura china en los años 850 donde ya era utilizado. De la misma forma aunque ya saturada, es muy común escuchar en los cursos de introducción a ciencias de la computación como la invención del ábaco como un parteaguas en los comienzos del calculo electrónico. Es extraño, maquinas que nos parecen obsoletas como ábacos o perlas fueron tan importantes para el calculo de miles de cifras en un tiempo en el que la tecnología era bastante limitada. Es muy importante también mencionar el aporte de la civilización romana ya que de manera concisa, los romanos y los egipcios compartían una visión utilitaria de la matemática por lo que los aportes de estos grupos fueron mas dirigidos a la aplicación de dichos conceptos. Ahora bien, desde estos días los cuestionamientos morales acerca del Que tanto estamos aportando realmente? no es nada nuevo, ya que autores como Cicerón en el texto *"Disputaciones tusculanas"* mencionaban:

Nosotros hemos limitado este arte a la medida y al calculo

Por lo que, a pesar de que como en el libro ***"Del ábaco a la revolución digital"*** mencionan que las construcciones de los romanos siguen en pie hasta nuestros días debido a una magnifica matemática aplicada, eso no significa que sus exponentes estuvieran conformes con sus logros. Es mas, probablemente sentían deseos de lograr mas con esta ciencia. En el libro Elogio a la pereza, mucho de su contenido abarca esta historia de las matemáticas, habla acerca de sus sistemas numéricos, los patrones y redacción que permitió a sus sabios transmitir su información a cada nueva generación. Cada matemática le permitió a su civilización afrontar diferentes

problemas, desde la construcción de edificios hasta la distribución de recursos. Sin embargo podemos empezar a ver un patrón...

Los egipcios guardaron sus avances en el **Papiro Rhind**.

Los griegos preservaron los suyos en los **Elementos de la geometría**.

Los chinos hicieron lo mismo en **Los nueve capítulos sobre procedimientos matemáticos**.

El registro de datos y ejecución de procedimientos era incapaz de ser ejecutado de forma automática, requería el trabajo de centenares de genios de diferentes países y eras para hacer las primeras aproximaciones al cálculo y a la trigonometría que nosotros conocemos el día de hoy. No fue hasta muchos siglos después que se empezó a pensar en una pregunta: *¿Qué pasaría si nosotros no tuviéramos que realizar estos cálculos?* Y es aquí donde tenemos que dar un brinco de siglos de estas civilizaciones antiguas.

Imaginando una máquina que realiza milagros

Hemos trascendido el oscurantismo y hemos llegado a una Europa donde genios como Pascal existen. En particular estamos en año 1642 y es en este momento que Pascal diseña la maquina sumatoria. Pascal no sería el único matemático/ingeniero en diseñar los primeros antecesores de la computadora moderna, ya que en el mismo siglo gente como Leibniz construye su maquina calculadora en el año de 1674. Las maquinas de Leibniz y Pascal no realizaban los cálculos que podemos realizar el día de hoy, para nada...

Pero fueron un peldaño sumamente trascendente en la historia de la computación debido a que fueron artílugos que solucionaban un problema que se arrastraba desde la época de los antiguos sabios que era el poder realizar cálculos aritméticos de forma mas sencilla y automatizada.

A estos antecesores podemos agregar lo que algunos siglos después haría Jacquard con la invención de las tarjetas perforadas en año de 1801. Así mismo es muy importante recordar también el aporte realizado por Babbage quien ideó la Maquina analítica para 1834. Sin embargo no podemos dejar atrás también el aporte de Ada

Lovelace, quien habiendo sido discípula de Babbage, se encargo de introducir el concepto de "algoritmo informático" (*Información recuperada de National Geographic*). Cualquiera podría citar estos hechos, todos podemos memorizarlos y si no tenemos conocimiento de estas fechas o personas probablemente busquemos dicha información en internet, seria de hecho lo esperado. Para esta parte quiero que no nos detengamos únicamente en estos hechos, y hagamos hincapié en algo muy importante:

La computadora es un invento formado de muchos pequeños inventos

Ahora tras hacer hincapié en ese hecho, necesito que comprendamos que hemos relegado el papel histórico de esta ciencia una mera linea del tiempo con inventos y nombres que no hemos escuchado nunca. Y eso lo considero un problema. No es solo un capricho el querer hablar desde la filosofía de esta rama del conocimiento, sino que es **NECESARIO** comprender la historia de los inventos que antecedieron a los desarrollos de Turing o la arquitectura de Pascal. Si bien en ICC se estudia esta linea del tiempo, realmente no se le da el mérito a estos logros, claro que comprendiendo que la facultad y la carrera exige que se estudien muchos temas en tan poco tiempo, la realidad es que lo que menos tenemos es tiempo. Quizás para lograr atraer no solo a aquellos estudiantes con ciertas bases de la programación, sino también a aquellos interesados en la historia del mundo, se le debe de dar mas énfasis al estudio de los inventos que antecedieron a las maquinas que hoy manejamos. Si bien, el enfoque del trabajo no busca únicamente tratar el periodo que abarca únicamente logros bastante antiguos pero para comprender los logros más cercanos hay que darle un pequeño vistazo a lo que le antecedió y es por ello que puedo mencionar a través de la siguiente tabla aquellos aportes fundamentales para el desarrollo computacional del siglo 20.

Científicos pre-siglo XX	Año	Desarrollo
Pascal	1642	Invención de la maquina sumatoria
Leibniz	1674	Construcción de la maquina calculadora
Babbage	1834	Diseño de la maquina analítica
Lovelace	1842	Visión de una maquina programable a través de una serie de instrucciones

EL PROBLEMA DE UNICAMENTE MEMORIZAR FECHAS, NOMBRES E INVENTOS EN LA HISTORIA DE LA COMPUTACION

Teniendo en claro que para este punto de la historia no solo se diseñaban maquinas capaces de realizar operaciones que hubieran tomado generaciones completas en un pasado no tan lejano, que depararía a los científicos del próximo siglo? Bueno, parte del porque me interesa abordar este trabajo desde un punto de vista humanista es debido a que la computación si bien reporto largos avances a lo largo de la historia de la humanidad no fue hasta el siglo XX que comenzamos a realmente aprovechar el potencial de dichas maquinas. Bueno, eso dirían muchas fuentes... Sin embargo, creo que también es necesario cuestionarnos:

¿No fue el propio gobierno ingles quien ceso las investigaciones de Babbage?

¿No han sido también responsables de la la falta de inversión por parte de políticos y dirigentes a nivel mundial?

Si bien como un contra argumento a la falta de apoyo a la investigación computacional durante sus comienzos podría ser aquel caso donde se uso la maquina de Hollereit a finales del siglo XIX para calcular el censo de la población estadounidense, los avances de la computación si analizamos las lineas de tiempo, reportan avances graduales incluso algunos de carácter intrascendente para lo que los ordenadores modernos son capaces de ejecutar. No es hasta el ascenso del tercer Reich y el estallido de la segunda guerra mundial que se desarrolla una revolución en el campo de la computación. Y el medio por el cual esta ciencia se ha continuado desarrollando, de una o de otra forma sigue sin ser tan distante. Como una nueva generación de computólogos deberíamos de ser capaces de preguntarnos que tan responsables fueron los programadores detrás de estas maquinas de contribuir a una guerra? Y aun más importante es observar nuestra realidad y revisar si hemos realmente cambiado el enfoque de desarrollo de hardware y de software a casi 100 años de esta primera generación de ordenadores.Y si bien la industria de la computación así como todas sus posteriores entradas han buscado favorecer y facilitar la vida de millones de personas también vale la pena preguntarnos que tan necesario fue que se desarrollara una guerra a escala global para invertir en el equipo tecnológico que hoy conocemos.

Hoy en día ya no es útil simplemente hablar de tablas clasificando los logros o aportes a esta ciencia ya que desde la construcción del ordenador z1, se han generado tantos desarrollos que el presentárselos a un alumno de nuevo ingreso no solo no es una tarea sencilla, creo que difícilmente este va a conservar el interés en recordar los aportes de Turing o de Von Neumann si no tiene un incentivo histórico o filosófico detrás de esa consigna.

El memorizar todos estos hechos no va a garantizar que el alumno comprenda la importancia social que conlleva dedicarse a la programación. Como conclusión de la breve historia de la computación me importa dejar en claro que comprendo la asignación de este trabajo, que es saber explicar las bases de esta ciencia. Pero es también entendiendo dicha consigna que mi propuesta para poder lograr captar a un mayor numero de estudiantes de la carrera, reducir un poco la deserción académica y llamar la atención de aquellos que estén pensando en elegir estudiar Ciencias de la Computación, es necesario que antes de pensar en términos como algoritmos, programas o clases, reflexionemos acerca del como llegamos a las maquinas tan potentes que ocupamos para jugar un videojuego, buscar videos en internet o incluso hasta poder encontrar pareja. Es importante reconocer el valor histórico de cada peldaño que hemos subido pero también no serviría reconocer dichos progresos sin cuestionar la ética detrás de cada uno. La ciencia de la computación no es únicamente el acto de programar, diseñar o estructurar. Es un acto político y social de la misma forma.

No podemos ignorar la trascendencia que nuestro trabajo como programadores puede influir sobre asuntos tan delicados como en la economía o en diferentes conflictos bélicos. Y su bien podemos ignorar estos hechos y concentrarnos únicamente en el acto de programar de uno o de otro modo no estamos proponiendo soluciones a dichos conflictos éticos acerca de que tanto debemos de programar, para quien debemos de hacerlo y sobre todo:

¿Qué tanta libertad de pensamiento y de ética estamos relegando como programadores a empresas y gobiernos?

Finalmente Anexo una tabla con los logros mas importantes de la computacion para aquellos interesados en el estudio historico.

SIGLO XX	APORTES A LA COMPUTACIÓN
1938	Konrad Zuse construye la maquina Z1
1943	Se construye el ordenador ENIAC que estaba fuertemente inspirada en el trabajo de Zuse
1945	Von Neumann desarrolla su arquitectura de ordenadores
1951	Computadora UNIVAC entregada al gobierno de E.U.A
1958	SAGE System es el primer sistema de computadoras conectadas de forma internacional
1963-64	Implementación del código ASCII y creación de Basic
1971	Creación de la primera calculadora electrónica
1976	Steve Wozniak desarrolla la apple 1
1981	IBM y microsoft lanzan al mercado la PC (Personal Computer)
1985	Creación del lenguaje C++
1991	Se lanza el primer Kernel de Linux
1995	La primera versión de Java es lanzada al mercado
1998	Apple lanza al mercado al I-Mac

PARTE 2: LO INCREIBLE DETRÁS DE TU COMPUTADORA

Ya vimos brevemente como surgieron las computadoras, sin embargo si deseas cumplir el objetivo de lograr que te quedes o ingreses a la carrera, sería interesante abordar algunas otras generalidades. Usualmente en las primeras clases en la facultad de ciencias buscan explicarte también lo que vamos a ver en esta sección.

No te enseñarán a programar en Phyton o Java la primera clase, sino que se trata de hablar de la arquitectura de un ordenador, cuales son sus dispositivos de entrada, salida, que funciones cumple cada parte de tu dispositivo así como algunos otros conceptos como la definición de los paradigmas de programación y en específico para la materia de ICC se define el paradigma orientado a objetos. Suena y es bastante información ...que no todos los profesores saben abordar correctamente. Hasta me duele la cabeza de solo imaginarlo



Yo he estado en la posición de necesitar una guía rápida para consultar alguna generalidad del curso de Introducción, muchas veces lo útil es aproximarse a la prensa

de ciencias y en específico a los textos de Canek. Habiendo criticado en la sección anterior la ausencia de contenido social en su material, debo de reconocer el valor práctico que ofrecen sus textos.

Sin subestimar la inteligencia de un recién llegado, saber que autores son los indicados para la lectura en los primeros semestres tampoco es sencillo y no debería de ser trivializado. Como final de esta introducción yo anexo algunas referencias que consideraría vitales de hacerle saber a una persona que acaba de llegar a ciencias o regresar tras un periodo de ausencia y necesitara ponerse al corriente.

- **Lógica de programación orientada a objetos** escrito por **Efraín Oviedo**
- **Introducción A Las Ciencias De La Computación Con Java** de **Canek Peláez**.

Creo que para poder entender el material de Canek de una mejor manera se necesita tener bases teóricas sólidas, así que no prometo que al terminar esta parte sepas todo lo necesario para afrontar todos los desafíos de tu primer semestre pero espero poder abrir un puente entre la teoría y tú.

Para programar no siempre gana la curiosidad o la inteligencia, en su lugar gana la constancia y dedicación. ¡NO TE RINDAS!

¿CÓMO EMPEZAR A ESTUDIAR TEORÍA DE ICC SIN MORIR EN EL PROCESO?

Hasta poco antes de escribir este trabajo yo pensaba únicamente quedarme en el contenido enfocado a la historia de la computación, sin embargo me di cuenta de que para atrapar la atención de una persona que no sepa que estudiar, como docentes tenemos una responsabilidad social que involucra darle un vistazo al futuro. Líneas de comandos, terminales, lenguajes orientados a objetos, lenguajes meramente funcionales. ¡ESAS PALABRAS LE VOLARÍAN LA CABEZA A CUALQUIERA!. Sabiendo ahora un destello de lo que fue la historia de la computación yo quiero invitarte a reflexionar en lo siguiente:

¿Cuando fue la primera vez que usaste una computadora?

Si regresas sobre tus propios pasos, tal vez sentiste emoción, tal vez no supiste como ocuparla, tal vez solo querías jugar algún videojuego. Imaginemos ahora este escenario. Al entrar a la facultad quizás te parezca bobo, pero quisiera preguntarte:

¿No sentiste que estabas usando una computadora por primera vez de nuevo?

Y es que, al tratarse de una forma diferente a este aparato se sienten muchas cosas, desde la emoción por instalar una distribución de Linux, la frustración de no saber usar Overleaf o Latex hasta aprender que de la computadora más vieja que tengas, puedes darle nueva vida si sabes exprimirla correctamente.

Yo mismo sé que la introducción de la parte 2 peca de lenguaje formal y de citar autores, pero créeme que tengo una razón para hacerlo. Y esa razón se llama intuición. Soy consciente de que mi trabajo no es el único en cubrir estos temas, soy consciente de que faltará información y tengo presente en mi mente que no existe una forma perfecta de aproximarse a las bases de la programación. Sin embargo también confío en ti como lector. Quizás nunca hayas pensado en el impacto emocional que tus textos o tus materiales de consulta te hayan dejado, sin embargo aunque no lo creas, trasciende más de lo que puedes imaginar. Para esta parte del trabajo te voy a pedir que pienses y recuerdes ese primer momento en el que con tus manos tomaste el teclado, te sentaste frente al monitor y confiaste en que lograrías entender lo que había detrás de esa pantalla.

Yo hoy te pido que confíes en las referencias que te di y al mismo tiempo que lees mis palabras, vayas a buscar esos libros. En este momento vas a aprender como funciona tu computadora por primera vez....

¿Estás list@?

RECUERDA EL PRINCIPIO DE LAS CUATRO U'S

Tras toda esta introducción te deberás estar preguntando ¿Cuáles son los dispositivos de entrada y de salida? Bueno para entender eso, hay que hablar de un par de cositas llamadas Unidades. La unidades de la computadora se encargan de diferentes tareas por que podemos clasificarlas en 4 tipos. Lo importante si es que tienes un examen es que recuerdes que hace cada una.

Pero a nivel personal también es importante que recuerdes que cuáles son estas unidades. Entonces en la siguiente lista nos referiremos a cada tipo de unidad como una U Cabe aclarar que las siguientes definiciones son la manera más simple en la que yo podría entender estos conceptos.

- **U DE ENTRADA:** *Es lo que nosotros ingresamos al computador.* Se logra por medio de un dispositivo que no está dentro de la computadora, sino que nosotros operamos por fuera. Puede parecer bastante obvio, pero muchas veces se nos olvida esta definición. Podemos agrupar a todos los dispositivos en los que ingresamos datos dentro de la computadora como parte de la unidad de entrada.
Así que si al leer esto piensas en un **teclado, una usb, o un Scanner** estás en lo correcto.
- **U DE SALIDA:** *Es lo que nosotros obtenemos de la computadora* La invitación para esta parte es es dejar de ver nuestras copias de la universidad o lo que vemos en el monitor únicamente como algo trivial, y lo veamos como el resultado de un proceso en el que tras ingresar datos, nosotros vamos a recibir algo a cambio. No tanto como una recompensa sino como alguien que pide un favor y recibe exactamente lo que pidió.
De esta forma cosas como el **escuchar música a través de tus audífonos por Bluetooth, ver las fotografías en tu página de Instagram o recibir las fotocopias de un libro que de otra forma no podrías pagar** son ejemplos de la unidad de salida
- **U DE MEMORIA:** *Es el medio por el cual la computadora almacena información* Las cosas se vuelven mucho más interesantes, ya que aquí NO ingresamos algo nosotros como usuarios y mucho menos recibimos algo como una impresión o un audio. Cuando hablamos de la unidad de memoria, a manera de

analogía yo lo vería como esa parte del cerebro encargada de almacenar nuestros recuerdos y clasificarlos como positivos y negativos para luego guardarlos en baúles en los que o nos sentimos orgullosos de ese recuerdo o nos da vergüenza ememorar

Una computadora almacena información a través de circuitos y para permitir o negar la transmisión ocupa valores de tipo "booleano" (True o False) para prender o apagar el circuito. Esta información se podría decir que es recolectada. Y cuando decimos que una computadora recolecta estamos diciendo que separa la información en grupos de datos numéricos (*que se basan en estos estados de tipo verdadero o falso*)que conocemos como:

- **Bit:** Es un numerito con valor o de 1 o de 0, siendo uno verdadero y cero falso
 - **Byte:** Imagina ahora una lista o "cadena" de los numeritos que definimos como bits y dicha lista al trabajar en equipo puede representar una letra u otro tipo de valor que no sean solo ceros o unos.
 - **Palabra de computador :** Es algo complicado de definir esta palabra, pero ahora imagina que nuestra cadenita de bits ya no solo es una cadenita, sino que es un grupo de muchos bits. Al ser más grande es más respetada por el resto de las unidades entonces puede ser vista como 3 cosas, una orden, un dígito y una entidad.
-
- **U DE CONTROL:** Podemos verla vulgarmente como *es el alcalde de las chicas superpoderosas*. Es realmente ridícula la referencia pero cuando la memoria necesita realizar algo, llama a la unidad indicada para realizar una determinada tarea. Le da una serie de instrucciones a realizar y la unidad encargada las ejecuta. Básicamente esta unidad es como el jefe de Tom Cruise en *Misión imposible* que le encomienda una misión si decide aceptarla, y la unidad que debe realizar el proceso en el papel de Tom Cruise siempre acepta la misión de realizar el proceso adecuado.
 - **U ARITMÉTICA Y LÓGICA:** Si lograste entender la unidad de memoria en realidad esta es la más sencilla de asimilar puesto que volvemos al punto en que TU como usuario NO HACES NADA en este paso. Podemos decir que la unidad aritmética y lógica *es la calculadora de bolsillo de al computadora* . Y si lo volvemos a ver de forma vulgar es como una de las chicas super poderosas que

responde al teléfono y va a la acción, solo que en lugar de golpear monstruos se encarga de realizar procesos aritméticos y de carácter booleano con otros datos.

Cuando nosotros empezamos a programar se nos pide que entendamos de forma casi perfecta conceptos bastante abstractos como el manejo de cadenas de bits, bytes, y unidades de conversión. Repasamos estos temas mediante varios ejercicios de matemáticas. ¡Y TENEMOS PÉSIMOS RESULTADOS! Creo que parte del problema se debe que aunque las definiciones que aprendemos son correctas, no dejamos de ver al manejo de la información como insípidas cadenas de números que debemos de convertir a números en base decimal o hexadecimal. Es por ello que es MUY importante que entiendas el concepto de unidades de control, porque quizás aunque pueda ser como cuando le queremos atribuir algún significado divino a nuestro sufrimiento para no padecer tanto dicho dolor, quizás te pueda ser de utilidad recordar que si realizas esos tediosos ejercicios de conversión de unidades en realidad no harás eso toda la carrera sino que al estar conociendo a tu computadora de manera más íntima estás simulando ser un procesador aritmético convirtiendo largas cadenas de bits.

Traté de sintetizar de una forma más amena el contenido que aborda este tema. Usualmente en la bibliografía disponible para los computólogos de nuevo ingreso se suele hablar en un lenguaje bastante técnico. SI, las formalidades son necesarias. No, no estoy descartando la importancia de aprender dicho lenguaje pero *no vamos a entender mejor cosas como estas si únicamente hablamos en este lenguaje*

¿Cuantas personas no conoces que han abandonado la carrera por pensar que es muy compleja?

¿Cuantas veces no has sentido que tus docentes no te tratan más que como un ignorante aun cuando si estudias?

Como conclusión de este apartado, te invito a pensar en estas preguntas. Creo firmemente en que no es tan complejo lo que acabamos de leer en esta sección, sin embargo mucho material sigue careciendo de un acercamiento más cotidiano con el programador novato por lo que el esperar que entendamos únicamente mediante

tecnicismos aunque puede resultar funcional a la larga podría generar una generación de programadores que realmente no entiendan su equipo de trabajo.

EL ARTE DEL DISEÑAR

Habiendo entendido un poco cómo trabaja tu computadora ¿Qué te parece si vemos algunas bases para programar de forma hecha y derecha? Perdonen la ausencia de formalidad para este punto, pero realmente me emociona el hecho de poder explicar estos temas, y realmente quiero compartir lo interesante que puede llegar a ser el aprender de computadora. Aunque a niveles avanzados claro que es complejo programar, no significa que sea imposible y para que no sea imposible llegar a los problemas difíciles, repito lo que dicho durante todo mi trabajo y es que tenemos que tener buenas bases de programación.

Lo más bello de la programación no reside en lo que está en el monitor necesariamente, sino en todo aquello que lo rodea. Tal como decía Cerati en su canción Vivo:

El fin de amar es sentirse más vivo

¿Cómo vamos a adquirir ese amor por la programación si lo que aprendemos no nos hace sentir más vivos? Pensemos un poco en esto, ya que si reducimos el diseño de un programa a una simple plantilla a seguir, le estamos arrancando el valor artístico y creativo al acto de diseñar. Y aunque en esta sección vamos a ver algunas bases del buen diseño de programas me gustaría que ahora rompamos el esquema de lo que es el diseño de programar y tomemos un minuto para apreciar que el buen diseño es producto de eliminar y destrozar aquello que no funciona, crear y destruir. Es el arte de amar y perder. Soltar y aprender. Si necesitan un índice del contenido, veremos como se crean programas desde el análisis del problema hasta su mantenimiento. Tomando como base lo escrito en el libro **"Introducción al desarrollo de programas con JAVA"** escrito por **Amparo López Gaona**. Para variar un poco y agilizar el trabajo de aprendizaje para el lector pensé en incluir un mapa conceptual en el que se hable de estas fases. Tenemos que tener en cuenta que tenemos una serie de alrededor unos 7 pasos:

1. ¿Qué tengo que hacer? — **Definición del problema**

Al hablar de la definición del problema lo que se hace es dejar en claro cual es la consigna a cumplir. Cuál va a ser la consigna de nuestro programa así como determinar especificaciones técnicas de lo que el cliente nos va a solicitar

2. ¿Qué cosas necesito para resolver mi problema? — **Análisis del problema**

Analizar el problema requiere que evaluemos las posibles soluciones de nuestra incógnita. No es sencilla esta tarea porque requiere que nosotros al diseñar seleccionemos entre este abanico de elecciones. Definimos también el análisis como nuestro de-limitador, ya que pueden haber factores internos que limiten nuestro rango de acción, como el tener compañeros que no sepan trabajar en equipo o que tu como programador aún no tengas suficiente experiencia solucionando un determinado tipo de problema. Es por ello que en nuestro mundo lo que se hace es evaluar quien puede realizar determinadas tareas para agilizar el desarrollo. Entonces en resumen se ven nuestras posibilidades, ventajas y desventajas antes de empezar como tal a diseñar

3. Ok, no me queda otra opción más que pensar en una solución. — **Diseño de la solución**

Diseñar es la parte más compleja porque en realidad ¡NO OCUPAMOS LA COMPUTADORA EN ESTE PASO! Entonces cuando nosotros diseñamos como computólogos tenemos que realizar un trabajo manual a papel y lápiz para determinar si nuestro mecanismo de aproximarnos al problema es el correcto. Me uno al consejo de voces que recuerda la importancia de este paso, ya que en realidad si no realizamos un diseño de algoritmo, se presentan muchos problemas en las posteriores fases. *Nosotros podemos definir un algoritmo como uno de esos rompecabezas infantiles de 50 piezas hechos en serie que al unir las piezas necesitamos que la imagen final sea la misma para todos los rompecabezas.*

4. Para que otros no sufran lo mismo que yo, vamos a anotar cada paso de mi desarrollo — **Documentación**

Es en este momento que me río de mi mismo por no colocar este paso en primer lugar, ahora bien, la documentación en muchos libros del paradigma orientado a objetos lo dejan hasta el último punto del desarrollo de soluciones y es un poco muy contradictorio considerando que tendría que ser uno de los 3 primeros

pasos. El documentar es MUY MUY MUY importante, créelo de alguien que ha tenido problemas por no documentar que realiza cada método de su programa, ha requerido repasar de manera casi diaria las buenas prácticas de la documentación en java y realmente ha sufrido para aprender Markdown para poder ofrecer a su asesor de prácticas la documentación necesaria que justifique programas de no más de 100 líneas. Entones podemos resumir este punto como *Documentar es la buena costumbre de marcar tus pasos sobre un sendero peligroso, para que si tu mueres, el siguiente explorador no sufra el mismo destino, o si lo sufres, por lo menos llegue ligeramente más lejos.* La correcta documentación se realiza como mencioné anteriormente anotando aquellos pasos que vamos tomando, lo que si puedes hacer, lo que no puedes, lo que absolutamente no debes realizar, entre otras cosas.

5. Tiene sentido lo que hago.... Entonces a codificar! — **Implementación**

Este paso consiste en llevar lo que encontramos en la fase de diseño a nuestro ordenador. Hay que programar de forma ordenada y coherente entonces mi recomendación, escribe primero el código a mano, porque aunque en materia de diseño sepas que tienes que hacer, puede que haya problemas a la hora de adaptar tus avances en un lenguaje de programación. Programa correctamente y en realidad puede que tus errores no sean tan graves ya que los errores por mala escritura son más comunes de lo que crees.

6. Ok, me equivoqué en esto, aquello, el otro, hay que corregirlo. — **Depuración**

¡ERRORES POR DOQUIER! Esta parte del proceso se trata de identificar los errores que pueden hacer que tu programa no compile, o que compile pero no realice la tarea que le pides. Hace relativamente poco realicé una práctica en markdown en la que hablo de los tipos de errores más comunes. Fue un trabajo que me ayudó a entender esta clase de errores, entonces les anexo la liga para que lo puedan consultar, ya que es un trabajo de acceso público:

<https://github.com/CCLaboratorio/GabrielPerez/blob/main/documentos/practica2.md>

7. Me costó mucho hacer que este programa funcionara, debo de mantenerlo así — **Mantenimiento**

Es aquí donde si ejecutamos bien todos los pasos nosotros tenemos que ser

capaces de mantener vigente nuestra información. Algo muy interesante es que en los libros de teoría se nos indica que mantengamos un uso constante del programa, y debemos de tener en cuenta que va a haber ocasiones que vamos a tener que realizar errores del tipo "*correctivo*" si los errores son a causa del cambio de software, del tipo "*adaptativo*" para mejorarlo, básicamente como realizar una expansión del programa original.

Ahora tras haber visto los pasos más importantes del desarrollo de software, creo que cualquiera se podría asustar debido sobre todo al énfasis que realicé al trabajo del paso de diseño y el paso de la documentación. Es aquí donde te digo que no te presiones. Muchas veces hay profesores y ayudantes que son muy comprensivos a la hora de explicar los problemas que puedes llegar a tener en cuestión de no saber cómo implementar la solución a un problema, por el otro lado hay personal académico que de forma constante puede llegar a ser hiriente con su retroalimentación. Lo más importante en esos casos es no darles demasiada importancia a esos comentarios cargados de lógica o tacto con el alumno todos tienen un ritmo diferente de aprendizaje en esta facultad, por lo que si no sabes cómo compilar o manejar determinados comandos en tu terminal, no dejes de practicar, y si tienes la oportunidad de tener sesiones adicionales con tus profesores no dudes en solicitarlas.

Recuerda que el progreso en el camino de la programación pocas veces es lineal y que pocas veces dominamos un idioma, una distribución o un buen diseño de programas a la primera de cambios, es a veces el fallar mucho la clave para poder recapacitar aquellas malas prácticas de programación y volverte un buen programador@.

¿PORQUÉ LOS PROGRAMADORES SE VISTEN DE NEGRO

Tras este pequeño viaje en el que vimos diferentes puntos de las bases de la programación... quisiera primero agradecerte por llegar hasta aquí. La verdad me ha costado bastante realizar cada palabra de este texto. No se que tan bien he de haber hecho el trabajo, puesto que cada vez que intento hacer algo diferente en mi carrera siento que me alejo más del propósito de lo que debe de ser un computólogo. Algo que me ha tenido frustrado desde que retomé la carrera ha sido que para todo se usa chat gpt, que su propósito de "*consultor*", realmente solo es una sugerencia. Me imagino que tal vez, solo tal vez no soy el único loco que busca trascendencia en su vida y un propósito aquí. Y créeme que estoy tratando de ser lo más objetivo aquí puesto que no busco el "*ad misericordiam*" de uno o dos docentes. Quiero saber que si llegaste hasta esta parte del reporte, al menos una de las cosas que haya dicho te permitan entender mejor lo que veas en tu clase de ICC.

Si leíste con atención el contenido de la parte uno, tal vez no lo alcances a ver completamente, pero ser un programador va mucho más allá de la pantalla, si leíste con atención la parte dos, programar también requiere creatividad y espontaneidad. En este trabajo traté de alejarme lo más posible de los estándares de los reportes de la misma materia, de los típicos cuadros sinópticos sin alma o de las reseñas otorgadas por una IA. En este momento de la historia de la programación si hablamos con puros términos científicos casi nunca sabremos si lo que se ha escrito realmente ha sido pensado y cosechado por un ser humano.

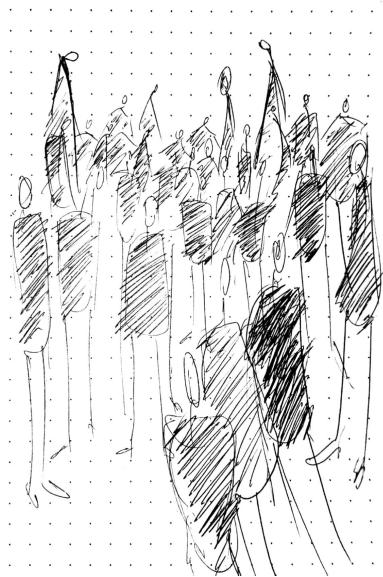
El texto no únicamente hace crítica del material y la enseñanza actual en la carrera sino de la propia idiosincrasia del alumnado al solo buscar pasar las materias y realmente no buscar entender lo que se le presenta. Lo digo de alumno a alumno. Pero bueno, antes de cerrar esta investigación debo responder a una última pregunta la cuál es la que indica el título. Los programadores al menos en la facultad suelen vestirse mucho con el color negro, no se si haya una relación entre ser programador y usar el color negro en la ropa, pero a veces siento que es una cuestión fuíebre. Retomando la encuesta de al inicio del proyecto, al preguntar:

¿Qué carrera estudiarías en lugar de Ciencias de la computación si tuvieras la oportunidad?

Más de la mitad eligieron carreras desde Neurociencias hasta Historia. Claro que hubieron respuestas que afirmaban sentirse completamente a gusto con la carrera pero cuando dejé la encuesta abierta, de 19 personas que respondieron, 12 personas eligieron otros caminos. El pensar de manera tan consciente que elegirías otro camino antes que la programación si bien no es malo es preocupante porque si una persona llega, se le es humillada y no se le da el apoyo necesario para poder volverse más fuerte en un aspecto académico, **¿cómo esperamos reducir los índices de abandono escolar?**

Quizás com este trabajo no haya ofrecido una respuesta definitiva pero busqué dar un enfoque diferente para aproximarse al estudio de la computación. Porque estoy cansado de que los estudiantes de la carrera de Ciencias de la computación vengan a enterrar sus sueños por la promesa de un futuro estable. Porque estoy harto de ver que no se hable del problema que representa el tener a una nueva generación de programadores nihilistas sin incentivos para poder crear cosas nuevas. Porque estoy harto de ver que mi generación de programadores vista de negro. Tal vez esa sea la versión larga de mi respuesta. Si deseas una versión más corta:

No conozco la razón por la cual nos vestimos de negro, pero me gusta imaginar que no será así para siempre .



FUENTES DE CONSULTA

1. *Research guides: LaTeX for COAD: Images and Figures.* (s. f.).
2. *How to make clickable links in LaTeX - LaTeX-Tutorial.com.* (2021,12 agosto). LaTeX-Tutorial.com. <https://latex-tutorial.com/tutorials/hyperlinks/>
3. Regino, E. M. O., & Carrascal, A. I. O. (2015). *Lógica de programación orientada a objetos*.
4. *Introducción al desarrollo de programas con Java* (3ra edición). (2013). [Versión impresa]. La prensa de ciencias.
5. Reventós, V. T. I. (2012). Del ábaco a la revolución digital: algoritmos y computación.
6. Williams, M. R. (1997). *A history of computing technology*. Wiley-IEEE Computer Society Press.