

CARTA DE POSTULACIÓN

Premio Estatal de la Juventud 2017

Carta de Motivos y Méritos

ASPIRANTE:

Rosas Hernandez Oscar Andrés

Índice general

1. Carta de Postulación	2
1.1. Introducción	3
1.2. Compilando Conocimiento: Una vista general	3
1.3. Página WEB	4
1.3.1. Evidencias - Estadísticas	4
1.4. Libros De:	12
1.4.1. Evidencias- Fragmentos:	13
1.5. Códigos Libres:	23
1.5.1. Evidencias- Fragmentos:	23
1.6. Redes Sociales	28
1.6.1. Evidencias - Fragmentos	28
1.7. Liberar todo mi contenido bajo la GPL v2	31
1.8. Conclusiones - Razones para Postularme	32
2. Documentación	33

Capítulo 1

Carta de Postulación

1.1. Introducción

Por medio de la presente busco postularme para el Premio Estatal de la Juventud 2017 en la categoría B (Jóvenes de 18 a 24 años) en la denominación de: **Trayectoria Académica**.

Antes que nada me presento, mi nombre es Oscar Andrés Rosas Hernandez soy habitante del municipio de Ixtapaluca, Estado de México.

Actualmente estudio el 3 semestre de la carrera de Ingeniería en Sistemas Computacionales en la Escuela Superior de Cómputo (ESCOM) del Instituto Politécnico Nacional, así como estoy por empezar el primer semestre de la carrera de Ciencias de la Computación en la Facultad de Ciencias de la UNAM.

Y Busco este Premio por (y para poder continuar con) mi Proyecto: Compilando Conocimiento, así que antes de hablar mas de mi, quisiera presentarles a mi trabajo y como es que este premio puede ayudarme muchísimo.

1.2. Compilando Conocimiento: Una vista general

Compilando Conocimiento es un Proyecto digital de varios ámbitos (Página Web, Códigos fuente bajo GPL, Libros en \LaTeX , Redes Sociales, etc...) en el que se busca crear un lugar donde los jóvenes puedan aprender usando las nuevas tecnologías a nuestro favor, este proyecto esta enfocado en la enseñanza de materias del nivel Superior y Medio Superior, sobretodo relacionadas con las matemáticas y con los sistemas computacionales.

LLevo con este proyecto casi 8 meses y busco con este premio poder llegar a muchas más personas así como ampliar las divisiones de dicho proyecto y abarcar más materias y brindar un contenido de mejor calidad.

La página WEB esta dividida en varias secciones, cada una de ellas sobre una materia en especial y dentro de cada sección se puede encontrar un link a cada una de las lecciones de esa materia.



Figura 1.1: Logo

1.3. Página WEB

Desde finales de 2016 decidí liberar en internet todos mis apuntes que había acumulado a lo largo de todo mi bachillerato, para hacerlo ocupé mis conocimientos de programación web y adquirí un hosting del dominio “*www.CompilandoConocimiento.com*” en el empecé a subir antes que nada lecciones de las materias que había llevado ese primer semestre de carrera.

Mis objetivos con esta página fueron:

- Crear un sitio moderno, rápido y cómodo para que los estudiantes pueda aprender por medio de estas nuevas tecnologías.
- Transmitir los conocimientos que veía en la Universidad en la Carrera de Ingeniería en Sistemas Computacionales pero de una manera más amena, mas tranquila, sin tantos tecnicismos.
- Ayudarme del uso de imágenes, colores, diagramas, gifs y demás recursos que tenía a mi disposición en la web, dichos recursos los creo yo desde cero usando software libre.

Al día de hoy la página sigue siendo la parte principal del proyecto y sin más les dejo unas imágenes que hablan por si solas del index de la página así como fragmentos de varios de los que considero mis mejores Post o Lecciones:

1.3.1. Evidencias - Estadísticas

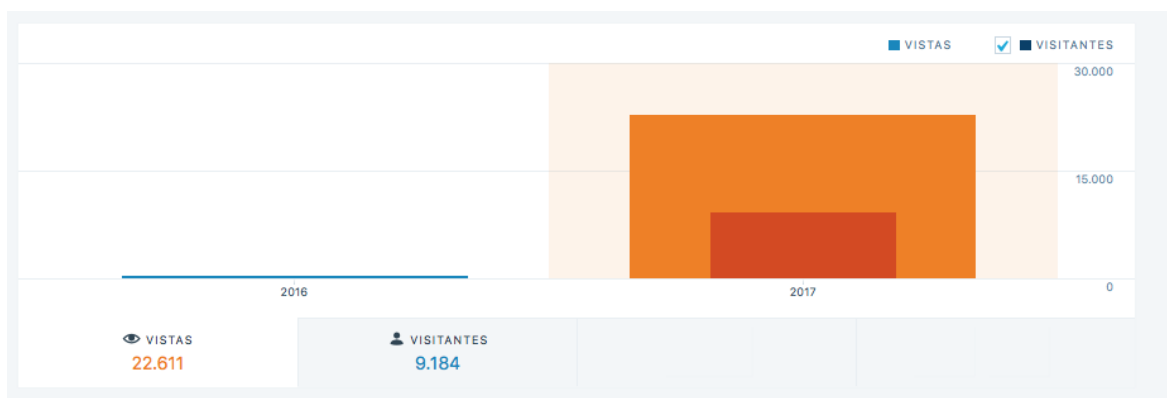


Figura 1.2: Evidencias de visitas



Figura 1.3: Evidencias de visitas



Figura 1.4: Fragmento de la Página



Curso: Cálculo Diferencial e Integral



Este es un clásico, es la llave que te va a abrir las puertas de las matemáticas como pocas materias pueden, créeme, desde formularios para secuencias del hombre que conocía el infinito, es hora de ver que hay en el fondo de la madriguera del conejo, de jugar con diferenciales.

75% Completado

Curso: Estructuras de Datos



La información lo es todo, quien controla la información controla el mundo, pero...¿Cómo se controla la información? ¿Cómo se almacena la información en una base de datos? ¿Cómo guarda tu historial el navegador? ¿Qué es una Tabla Hash? ¿Porqué no puede haber nada más rápido que un BTS para encontrar información? ¿Qué demonios es una LSE? ...Vamos a seguir la madriguera del conejo y veamos hasta donde nos lleva.

90% Completado


Curso: Ecuaciones Diferenciales



Esta es otra de esas materias que asustan pero que son útiles sin importar que estudies, esta es la culmine de Cálculo, es lo máximo, vamos, yo se que tu puedes, le puse colores a las ecuaciones para ti.

40% Completado

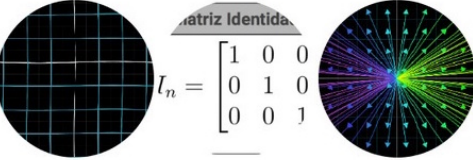
Figura 1.5: Fragmento de la Página



Curso: Java – Orientada a Objetos

Java es uno de los lenguajes más usados en la actualidad, gracias a Android, este lenguaje es el que tiene más salidas profesionales y además es clave pues muchas veces es lo que usamos para enseñarles al paradigma mas de moda últimamente: Programación Orientada a Objetos

95% Completado




Curso: Álgebra Lineal

"Difícilmente hay algo tan bonito como Álgebra Lineal, a pesar de las generaciones y generaciones de profesores y alumnos que han oscurecido su belleza y simplicidad con sus cálculos horribles sobre matrices." Si te das cuenta esta es una de las pocas materias que es necesario para casi todo.

¿Eres Físico? La necesitas. ¿Eres computólogo/ ingeniero? La necesitas. ¿Eres matemático? ¿Biólogo? ¿Economista? La necesitas.

75% Completado




Curso: Electrónica Analógica

En este curso veremos desde carga eléctrica, circuitos de CD, AC, Ley de Ohm y Kirchhoff, Thevenin y Norton, el uso de sensores y actuadores en circuitos, así como semiconductores, y todos los diferentes tipos de transistores.

50% Completado

Figura 1.6: Fragmento de la Página

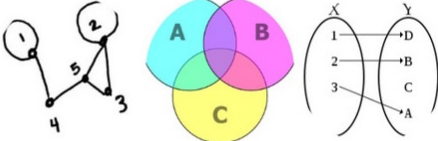
Curso: Teoría Computacional



¿Qué seríamos sin las demostraciones y los problemas clásicos de la computación? Pero...¿Qué es una computadora? ¿Es posible una máquina que lo haga todo? ¿Quién es Alan Turing? Vamos a investigar los fundamentos de la ciencia que más florecerá en este siglo... La ciencia de la Computación.

60% Completado

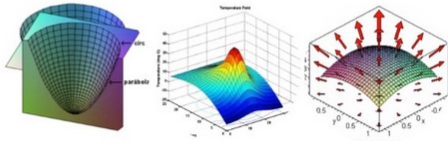
Curso: Matemáticas Discretas



Esta materia es la base de todas las matemáticas que se pueden aplicar a la computación, veremos de todo, desde lógica, conjuntos hasta los grafos, incluso pasaremos por la herramienta más poderosa: La inducción matemática.

90% Completado


Curso: Análisis/Cálculo Vectorial



Veremos sobre Vectores y como sumarlos, veremos problemas con el Producto Punto y Cruz, el Operador Nabla (gradiente, divergencia, rotacional y laplaciano) y derivación e integración Vectorial.

80% Completado

Curso: Entendamos C



Vamos a aprender uno de los lenguajes más conocidos e importantes de todos, el mítico C, veremos sobre tipos de datos, variables, funciones, sentencias de control, arrays, estructuras y memoria dinámica.

Figura 1.7: Fragmento de la Página

Series P: La Madre de todas las Armónicas

Para empezar hay que recordar que hay una serie muy famosa que se conoce como la Serie Armónica:

$$\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \dots$$

Un clásico

Podemos entonces hablar de las Series P, que es una generalización de las series armónicas, de la forma:

$$\sum_{n=1}^{\infty} \frac{1}{n^P}$$

Hola P, Soy una Serie P

Recuerda:

- Cuando $p \leq 1$ es la serie armónica (La cual diverge).
- Y también podemos saber (por el criterio de la Integral) que para cualquiera $p > 1$ la serie converge.

Figura 1.8: Fragmento de la Página

Lista

Ahora podemos dar hasta una mejor definición de lista: Un montón de nodos.



Las listas enlazadas son estructuras de datos semejantes a los `array` salvo que el acceso a un elemento no se hace mediante un índice sino mediante un puntero. La asignación de memoria es hecha durante la ejecución.

En una lista los elementos son contiguos en lo que concierne al enlazado. En cambio, mientras que en un `array` los elementos están contiguos en la memoria, en una lista los elementos están dispersos. El enlace entre los elementos se hace mediante un puntero. En realidad, en la memoria la representación es aleatoria en función del espacio asignado.

El puntero siguiente del último elemento tiene que apuntar hacia `NULL` (el fin de la lista).

Figura 1.9: Fragmento de la Página

Ecuación Diferenciales Exactas

Son las que podemos denotar de la manera:

$$M(x, y)dx + N(x, y)dy = 0$$

Para estar seguro que sean ecuaciones exactas, tenemos que hacer un paso más y es que:

$$\frac{\partial M(x, y)}{\partial y} = \frac{\partial N(x, y)}{\partial x}$$

Forma de comprobar

Si esto es cierto entonces podemos hablar entonces que tanto $M(x, y)$ como $N(x, y)$ son derivadas parciales de una función más básica, una $f(x, y)$:

$$M(x, y) = \frac{\partial f}{\partial x}$$
$$N(x, y) = \frac{\partial f}{\partial y}$$

Esta es la verdadera identidad de nuestras funciones

Figura 1.10: Fragmento de la Página

1.4. Libros De:

Desde muy pronto que cree la página web me di cuenta que eso no era suficiente, era común que muchas personas me felicitarán por mi trabajo pero que al mismo tiempo me pidieran más ejemplos o definiciones más exactas, que profundizará en esos temas aún más.

Así que gracias a todo eso y gracias a la ayuda de varios de mis compañeros en la Escuela Superior de Cómputo del IPN cree **Libros De:** que busca ser un complemento más serio y formal a la página.

Mis objetivos al crear esta división del proyecto fueron:

- Crear y aprovechar las ventajas que te da el Sistema \LaTeX que permite crear PDF como estos de una manera muy sencilla y con ello crear una forma de distribuir el conocimiento sin importar desde donde se estuviera viendo el contenido, sea un telefono, una laptop, un sistema con GNU/Linux, etc...
- Crear una sección de este proyecto en \LaTeX también tiene otras ventajas, principalmente que si libero el código fuente de dichos documentos cualquiera, por ejemplo un profesor de cierta materia puede tomar los textos que creo y modificarlos para que funcionen mejor en su clases, desde reordenar el texto, cambiar fácilmente ecuaciones o eliminar secciones sin modificar la estructura del mismo.
- El contenido de estos Libros de esta pensado en ser mucho más profundo de lo que veo en la página, dando un lenguaje mucho más formal, además de incluir mas ejemplos explicarlos de manera mas detallada.
- Esta sección del Proyecto esta alojada en Github y gracias a eso nos permite que cualquier persona pueda ver y entender como es que se crean estos textos desde cero, todas las modificaciones que hemos hecho y el día y la razón.

Sin más aquí dejo algunos fragmentos de dichos textos así como estadísticas de cuantas personas han leído dichos libros.

1.4.1. Evidencias- Fragmentos:

Estadísticas para 2017	
Enlace	Clics
github.com	1,012
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/raw/master/TeoriaDeNumeros/TeoriaDeNumeros.pdf	193
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/raw/master/Logica/Logica.pdf	174
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/raw/master/Conjuntos/Conjuntos.pdf	126
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/blob/master/TeoriaDeNumeros/TeoriaDeNumeros.pdf	87
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/raw/master/Relaciones/Relaciones.pdf	81
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/blob/master/Logica/Logica.pdf	61
github.com/SoyOscarRH	51
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/blob/master/Conjuntos/Conjuntos.pdf	45
github.com/CompilandoConocimiento/LibroTeoriaDeNumeros/blob/master/TeoriaDeNumeros.pdf	43
github.com/CompilandoConocimiento/ApuntesMatematicasDiscretas/blob/master/Relaciones/Relaciones.pdf	29
github.com/CompilandoConocimiento/LibroMatematicasDiscretas/blob/master/MatematicasDiscretas.pdf	23
github.com/SoyOscarRH/Algoritmos	13
github.com/CompilandoConocimiento/ApuntesAlgebraLineal/blob/master/TransformacionesLineales/TransformacionesLineales.pdf	13
github.com/CompilandoConocimiento/ApuntesCalculo/blob/master/SeriesYSucesiones/SeriesYSucesiones.pdf	12

Figura 1.11: Cantidad de vistas sobre Libros en Julio

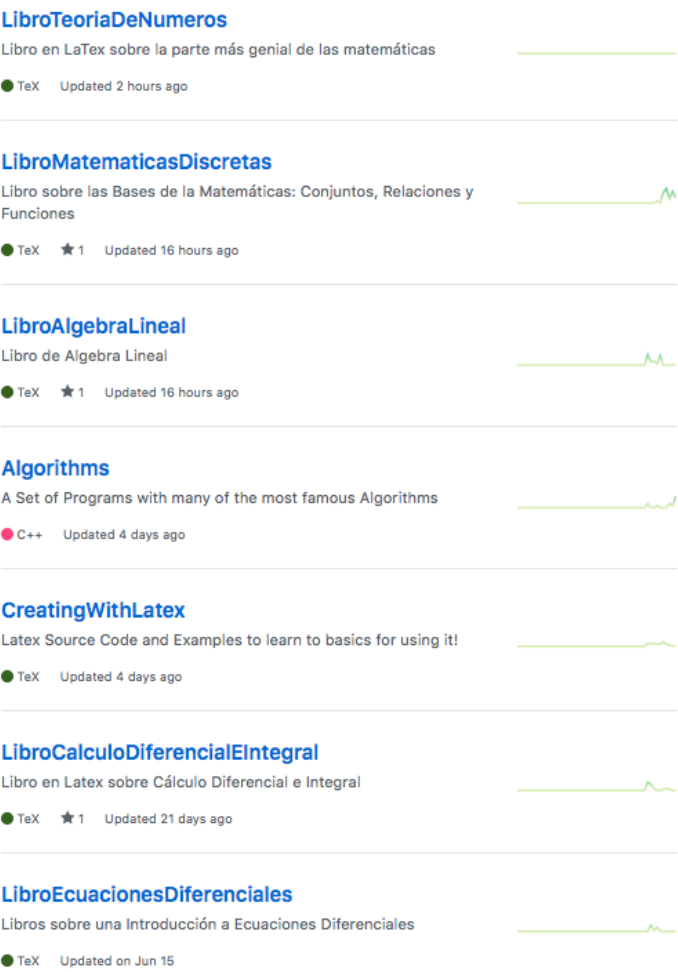


Figura 1.12: Algunos de los libros que he creado

Fragmento de Texto

PROYECTO COMPILANDO CONOCIMIENTO

MATEMÁTICAS DISCRETAS

Teoría de Números

Una Pequeña Introducción

AUTOR:

Rosas Hernandez Oscar Andres

Fragmento de Texto

Índice general

1. Naturales	3
1.1. Principio de Buen Orden	4
2. Divisibilidad	5
2.1. Algoritmo de División	6
2.1.1. Par e Impar	7
2.2. Divisibilidad	8
2.2.1. Ejemplos	9
2.2.2. Propiedades de Divisibilidad	10
2.3. Máximo Común Divisor: GCD/MCD	12
2.3.1. Propiedades de MCD/GCD	13
2.3.2. Identidad de Bezout	15
2.3.3. Propiedades de MCD/GCD: Bezout Edition	16
2.3.4. Primos Relativos	17
2.4. Algoritmo de Euclides	18
2.4.1. Como Aplicarlo	19
2.4.2. Ejemplo	20
2.4.3. Algoritmo Extendido de Euclides	21
2.4.4. Ejemplo	22
2.5. Mínimo Común Múltiplo: MCM/LCM	23
2.5.1. Propiedades de MCM/LCM	24
2.6. Ecuaciones Diofánticas	25
2.6.1. Soluciones	25

Fragmento de Texto

ÍNDICE GENERAL

ÍNDICE GENERAL

2.6.2. Soluciones Generales 26

2.7. Función Phi de Euler: ϕ 27

2.7.1. Ejemplo 28

2.7.2. Propositiones Importantes 28

3. Números Primos 29

3.1. Definición 30

3.2. Propositiones Importantes 30

3.3. Como Saber si $n \in \mathbb{P}$ 31

3.3.1. Fuerza Bruta Inteligente 31

3.4. Teorema Fundamental de la Aritmética 32

3.4.1. Factorización Prima 33

4. Algoritmos Útiles 34

4.1. Exponenciación Binaria 35

5. Congruencias 38

5.1. Congruencia Módulo N 39

5.1.1. Congruencia: Una Relación de Equivalencia 40

5.1.2. Propiedades 41

5.1.3. Módulo: $A \% B$ 43

5.2. Exponenciación Modular: $b^e \equiv s \pmod{n}$ 43

5.3. Criterios de Divisibilidad 45

5.4. Ecuaciones Módulo N 46

5.5. Aritmética Modular: Clases $[a]_n$ y \mathbb{Z}_n 47

6. Grupos, Anillos y Campos 48

6.1. Grupo 49

6.1.1. Grupo Abeliano 49

6.2. Anillo 50

Fragmento de Texto

2.1. Algoritmo de División

Definición Formal

Dados dos enteros a, b donde $b \neq 0$, existen otros dos enteros únicos q, r , donde $0 \leq r < |b|$ tal que se cumple:

$$a = bq + r \quad (2.1)$$

Vemos que básicamente nos dice cuántas veces cabe b en a sin pasarse (esto es q) y cuantos le faltan para alcanzar a a (esto es r).

Demostración:

El primer paso es crear el conjunto $Residuos = \{a - |b|q \mid q \in \mathbb{Z}, (a - |b|q) \geq 0\}$.

Ahora lo primero que tenemos que ver que es $|Residuos| \neq 0$. Para hacerlo veamos por casos, si $a < |b|$, entonces intenta a $q = -1$ y vemos que $a + |b|$ siempre será mayor o igual que 0. Si $a > |b|$, entonces intenta a $q = 1$ y vemos que $a - |b|$ siempre será mayor o igual que 0. Finalmente si $a = |b|$ cualquiera de los 2 ejemplos anteriores te sirven. Por lo tanto mínimo $Residuos$ tiene mínimo un elemento.

Esto es un conjunto que básicamente contiene a los residuos, o visto de otra manera a los números que salen como resultado de sumarle múltiplos de $|b|$ a a y que son mayores que 0.

Ahora gracias al principio de buen orden (y que $Residuos$ es el conjunto de los Naturales más el cero) podemos llamar a r al elemento más pequeño de este conjunto.

Ahora, gracias a la definición del conjunto $Residuos$ podemos decir que $r = a - |b|q_1$ que es decir $a = |b|q_1 + r$.

Ahora podemos poner esto como $a = bq + r$ donde si $b < 0 \Rightarrow q = -q_1$ y si $b > 0 \Rightarrow q = q_1$.

Para ver que $0 \leq r < |b|$, bueno, es mayor o igual que 0 porque pertenece a los Naturales más el cero, ahora para ver que es menor que $|b|$, basta con ver que si no fuera así pasaría que $r - |b| \geq 0$ (donde r es el elemento más pequeño del conjunto $Residuos$) que es lo mismo que poner $(a - |b|q_1) - |b| \geq 0$ que es lo mismo que $a - |b|(q_1 + 1) \geq 0$, ahora basta con ver que esa no es la r más pequeña, pues entonces si $a - |b|(q_1 + 1) \geq 0$, también $a - |b|q_1 \geq 0$, por lo que la nueva r_2 (donde $r_2 = a - |b|q_1$), es más pequeña que r , pero elegimos a r como la más pequeña, por lo tanto contradicción.

Y ya por fin, para demostrar que q, r son únicos dados a, b , tendría que pasar que $a = bq_1 + r_1 = bq_2 + r_2$.

Recordemos que r debe de ser única, pues r es el menor elemento del conjunto del que tendríamos que sacar a la otra, así que r solo hay una.

Dado eso, tenemos que $a = bq_1 + r = bq_2 + r$ que es lo mismo que $bq_1 = bq_2$ que es lo mismo que $q_1 = q_2$ y bingo. Demostrado.

Fragmento de Texto

2.1.1. Par e Impar

Dado un 2 como divisor, osea $b = 2$, nuestra r siempre será 0 ó 1. Digo recuerda que $0 \leq r < |b|$.

Pares

Por lo tanto puedo definir a un número entero par como aquellos números que podemos escribirlos gracias al algoritmo de la división como $2q + 0$ o de manera más común como $2k$.

$$\begin{aligned} Pares &= \{a \in \mathbb{Z} \mid a = 2q + 0, \ q \in \mathbb{Z}\} \\ Pares &= \{2k \mid k \in \mathbb{Z}\} \end{aligned} \tag{2.2}$$

Impares

Por lo tanto puedo definir a un número entero impar como aquellos números que podemos escribirlos gracias al algoritmo de la división como $2q + 1$ o de manera más común como $2k + 1$.

$$\begin{aligned} Impares &= \{a \in \mathbb{Z} \mid a = 2q + 1, \ q \in \mathbb{Z}\} \\ Impares &= \{2k + 1 \mid k \in \mathbb{Z}\} \end{aligned} \tag{2.3}$$

Y de esto sacamos algunas ideas bastante obvias:

Ideas Importantes

- Un número n es un cuadrado $n = m^2$ si y solo si al aplicarle el algoritmo de la división con $b = 4$ implica que $r = 1$ ó $r = 0$.

Demostración:

Si es un número par $m = 2k$, entonces $(2k)^2$ que es igual a $4k^2$ donde podemos decir que $n = 4(k^2) + 0$.

Si es impar $m = 2k + 1$, entonces $(2k + 1)^2$ que es igual a $4k^2 + 4k + 1$ donde podemos decir que $n = 4(k^2 + k) + 1$.

Fragmento de Texto

- Si $b|a$ y $b|c$ entonces $b|a \pm c$

Demostración:

Sabemos que $a = bq_1$, y $c = bq_2$ por lo tanto podemos decir que sumar o restar ambas ecuaciones, lo que nos daría $a \pm c = bq_1 \pm bq_2$ que es lo mismo que $a \pm c = b(q_1 \pm q_2)$ por lo que podemos decir que $b|a \pm c$.

- Si $a|b$ y $a|b \pm c$ entonces $a|c$

Demostración:

Sabemos que $b = aq_1$, y $b \pm c = aq_2$, si restamos tenemos que $b \pm c - b = aq_2 - aq_1$, que es lo mismo que $\pm c = (q_2 - q_1)a$, que es lo mismo que $c = \pm(q_2 - q_1)a$ que es lo mismo que $c = q_3a$.

- Si $b|a$ entonces $b|ak \forall k \in \mathbb{Z}$.

Demostración:

Sabemos que $a = bq$ por lo mismo podemos decir que $ak = b(qk)$ por lo tanto $b|ak$.

- $b|a$ si y solo si $b| -a$ si y solo si $-b|a$ si y solo si $-b| -a$

Demostración:

Sabemos que existe q_1 tal que $a = bq_1$ para nuestro primer ssi basta con decir que $-a = b(-q_1) = bq_2$ y listo, encuentre a q_2 con lo que puedo afirmar que $b| -a$.

Para el segundo basta con ver que $a = -bq_3$ donde $q_3 = q_2$, con lo que puedo afirmar que $-b|a$.

Para el último ssi basta con con ver que $-a = -bq_4$ donde $q_4 = q_1$ así que puedo afirmar que $-b| -a$.

- Si $b|a$ y $a \neq 0$ entonces $|b| \leq |a|$.

Demostración:

Supongamos entonces que b divide a a y que $a \neq 0$, por lo tanto la frase $a = bq$ nos da mucha información, pues obliga a que b y q no sean ninguno 0, entonces tenemos que $a = bq$ donde $b \neq 0$ y $q \neq 0$.

Luego ya que no son 0, tenemos que $|q| \geq 1$ y $|b| \geq 1$, ya que sabemos como funcionan los números enteros tenemos que sin importar cuanto valgan q y b se cumple que $|b||q| \geq |b|$ esto es lo mismo que $|bq| \geq |b|$ y sabemos que $a = bq$, por lo tanto tenemos que $|a| \geq |b|$.

Esto es lo mismo que $|b| \leq |a|$

Fragmento de Texto

2.3. Máximo Común Divisor: GCD/MCD

Definición Formal

Dados dos números cualquiera $a, b \in \mathbb{Z}$ pero con mínimo alguno de ellos dos diferentes de 0.

Entonces decimos que el máximo común divisor de a y b denotado por $MCD(a, b) = GCD(a, b)$ es el entero positivo d que satisface:

- $d|a$ y $d|b$
- Si $c|a$ y $c|b$ entonces $c \leq d$.

Ideas:

Decimos que d es un división común de a y b si $(d|a) \wedge (d|b)$.

Ahora podemos construir el conjunto de los divisores comunes. $Divisores = \{d \in \mathbb{Z} \mid (d|a) \wedge (d|b)\}$

Ahora si, con todo esto listo, podemos ver que este conjunto nunca estará vacío. como 1 es un división común de todos los enteros.

Ahora podemos ver que el conjunto no es infinito siempre que alguno de ellos no sea cero, hay sólo una cantidad finita de divisores comunes positivos. Dentro de ellos hay uno que es el mayor.

La segunda condición se asegura de que d sea el máximo elemento dentro del conjunto.

Fragmento de Texto

2.3.1. Propiedades de MCD/GCD

Antes que nada, recuerda que para que tenga sentido hablar del máximo común divisor alguno de los dos a, b debe de ser diferente de cero. Porfis.

Recuerda también llamaré c a lo que salga de $c = \max(|a|, |b|)$.

Ahora supongamos que es a el que es diferente de 0, después de todo $MCD(a, b) = MCD(b, a)$

- Siempre se cumple que $0 < MCD(a, b) \leq \max(|a|, |b|)$

Demostración:

Para lo primero basta con recordar que 1 divide a todos los enteros, así que 1 siempre será un divisor común, por lo tanto, cualquier otro divisor que aspire a ser el MCD/GCD tendría que ser mayor que 1, o bien, si son primos relativos, ser el 1.

Basta con pensar que $c = \max(|a|, |b|)$ es más grande o igual que 1, y ahora veamos que es imposible que existe un número n que sea el máximo común divisor donde $c < n$. Ya que de ser así pasa que $\max(|a|, |b|) < n$. Digamos que puedo escribir a $n = c + k$.

Y eso nos diría que si $|(c + k)|a$ y $a \neq 0$ entonces $|c + k| \leq |a|$

Pero, c es positiva, y también k , por lo tanto la proposición $|c + k| \leq |a|$ es falsa. Espero que se vea claro porque, ya si c es el mayor de sus valores absolutos, si le añadimos otro natural a ese número solo se puede hacer más grande, haciendo imposible la frase $|c + k| \leq |a|$.

Por lo tanto, es imposible que exista dicha n .

Y el máximo común divisor queda atrapado en esos límites.

- Siempre se cumple que $MCD(a, 0) = GCD(a, 0) = |a|$

Demostración: Basta con pensar que $|a|$ divide a ambos, y es más grande que 1, así que vamos bien, y después pensar que si existiera algún divisor más grande que $|a|$ entonces se cumpliría que $|(a| + k)|a$ por lo tanto también se cumpliría lo que dijimos antes, (que si $|(a| + k)|a$ y $a \neq 0$ entonces $|(a| + k)| \leq |a|$) y eso claro es una contradicción por lo tanto, $|a|$ es siempre el mayor divisor común.

1.5. Códigos Libres:

Al yo ser un estudiante de dos universidades que estan enfocadas en la computación cree un espacio en Github donde además de subir los Libros que vamos creando en PDF y Latex subó las implementaciones de los algoritmos más importantes que he visto a lo largo de mi carrera para ayudar a las personas, ya que es común encontrar en internet dichos algoritmos pero mal implementados y sin comentar porque es que dicho algoritmo funciona.

- Dichos programas están documentados en inglés que es la lengua en la que se encuentra todo toda la información de la carrera, además de funcionar como incentivo para que mis lectores ejerciten su inglés.
- Dichos programas están creados y liberados bajo la GPL, de la cual hablaré después.

1.5.1. Evidencias- Fragmentos:

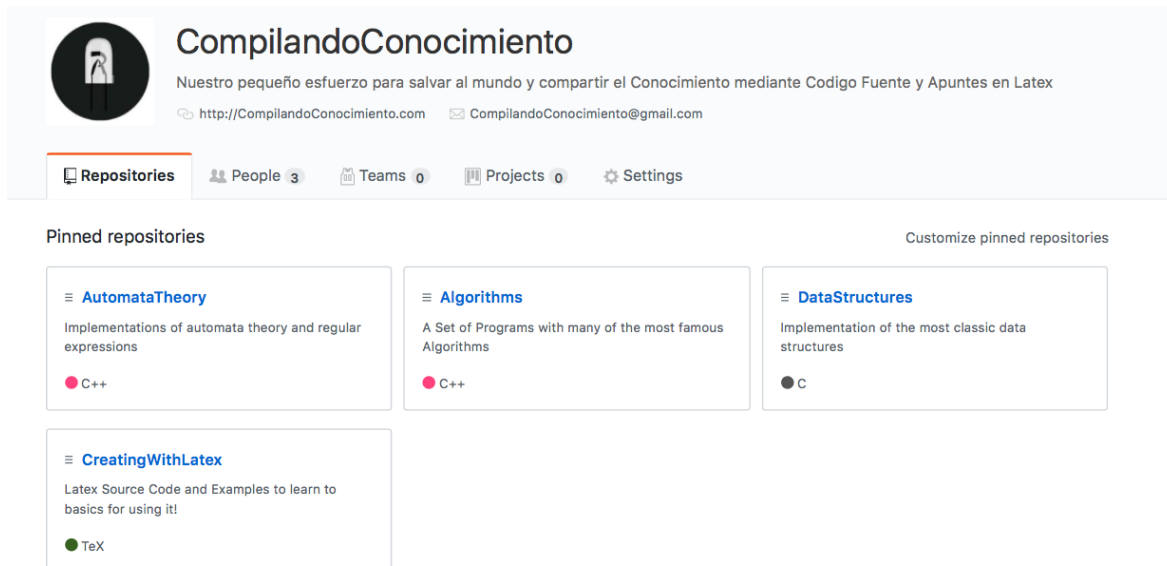


Figura 1.13: Github del Proyecto

This repository | Search

Pull requests Issues Marketplace Gist



CompilandoConocimiento / Algorithms

Watch 0

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Settings

Insights ▾

Branch: master ▾ Algorithms / Math / Euclidean-GCD-LCM.cpp

Find file

Copy path

SoyOscarRH Go man!

543d14c 4 days ago

1 contributor

277 lines (186 sloc) 14.9 KB

Raw

Blame

History



```
1  /*=====
2  ===== NUMBER THEORY FUNCTION FOR HUMANITY =====
3  =====*/
4
5  #include <math.h> //Include Libraries
6  #include <stdlib.h> //Include Libraries
7  #include <iostream> //Include Libraries
8  #include <vector> //Include Libraries
9  using namespace std; //Bad practice, dont do it kids!
10
11 typedef unsigned long long ull; //Just a so long name, sorry
12 typedef long long lli; //Just a so long name, sorry
13 typedef pair<lli,lli> PairOflli; //Just a so long name, sorry
14 typedef vector< vector<lli> > MatrixOflli; //Just a so long name, sorry
15
16
17
18
19
20 /*=====
21 ===== JUST TO LEARN AND SEE STEP BY STEP =====
22 =====*/
23 // ***** LONG DIVISION ALGORITHM *****
24 PairOflli DivisionAlgorithm(lli a, lli b, bool ShowIt){ //FN: Return the only q and r
25     lli q, r; //Variables, remember [0,|b|) in r
26
27     if (b != 0) { //If we have work to do
28         q = a/b; //Get me the floor of this
29         r = a%b; //Get me the reminder
30
31         if (r < 0){ //If we have a problem
32             (q >= 0)? q--: q++; //If q>0 add 1, else sub 1
33             r = a - (b*q); //Recalculate reminder
34         }
35     }
36     else {q = 0; r = a;} //b=0, this is always the result
37
38     if (ShowIt) { //You want to see it?
39         cout << "(a:<a<< ") = (b:<b<<")"; //Show it!
40         cout << "(q:<q<< ") + (r:<r<<")\n"; //Show it!
41     }
42
43     return {q,r}; //Return the info
```

```

44 }
45
46
47 // ***** EUCLIDEAN ALGORITHM *****
48 MatrixOflli EuclideanAlgorithm(lli a, lli b, bool ShowIt){ //FN: Return data for each step
49     MatrixOflli Data; //Step by step Data[i]={a,b,q,r}
50     lli RealA = a, RealB = b, r, q; //The last 2 variables for algorithm
51     bool SpecialCase = false; //This is a special case
52     if (b == 0) SpecialCase = true; //Just activate this flag
53
54     do { //Do this at least one time
55         auto Step = DivisionAlgorithm(a, b, false); //Get the long division
56         q = Step.first; //Get values
57         r = Step.second; //Get values
58
59         Data.push_back({a,b,q,r}); //Add the result to the data
60         if (SpecialCase) break; //Break if we are over
61
62         a = b; //The new a is b
63         b = r; //The new b is r
64     }
65     while (b != 0); //Do it! while we can
66
67     if (ShowIt) { //You want to see it?
68         cout << "=== Process: ===\n"; //Title
69         for (auto x : Data) { //For each element in the data
70             cout << "(a:" << x[0] << ") = (b:" << x[1] << ")"; //Show it!
71             cout << "(q:" << x[2] << ") + (r:" << x[3] << ") \n"; //Show it!
72         }
73         cout << "\nSo GCD(" << RealA << ", " << RealB << ") = " << abs(a) << "\n"; //You maybe want to know this
74     }
75
76     return Data; //Return the Data
77 }
78
79
80
81 // ***** EXTENDED EUCLIDEAN ALGORITHM *****
82 MatrixOflli ExtendedEuclideanAlgorithm(lli a, lli b, bool ShowIt){ //FN: Return data for each step
83     vector< vector<lli> > Data; //Step by step Data[i]={a,b,q,r,x,y}
84     lli r, q, Temporal; //Variables for algorithm
85     lli RealA = a, RealB = b, LastM = 1, LastN = 0, m = 0, n = 1; //Variables for Bezut Identity
86
87     bool SpecialCase = false; //This is a special case
88     if (b == 0) SpecialCase = true; //Just activate this flag
89
90     do { //Do this at least one time
91         auto Step = DivisionAlgorithm(a, b, false); //Get the long division
92         q = Step.first; //Get values
93         r = Step.second; //Get values
94
95         if (SpecialCase) {Data.push_back({a, b, q, r, 1, 0}); break;} //If b=0, original info is all we need
96
97         Temporal = m; //Lets save m
98         m = LastM - m*q; //Lets create the new m as lastm-lastm*q
99         LastM = Temporal; //Now you are the last m
100
101         Temporal = n; //Lets save n
102         n = LastN - n*q; //Lets create the new n as lastn-lastn*q
103         LastN = Temporal; //Now you are the last n

```

```

104
105     Data.push_back({a, b, q, r, m, n});           //Add the result to the data
106     a = b;                                       //The new a is b
107     b = r;                                       //The new b is r
108 }
109 while (b != 0);                                //Do it! while we can
110
111 if (ShowIt) {                                   //Nou want to see it?
112
113     cout << "=== Process for GCD: ===\n";        //The title
114     for (auto x : Data) {                       //For each element in the data
115         cout << "(a:"<<x[0]<<" ) = (b:"<<x[1]<<" )"; //Show it!
116         cout << "(q:"<<x[2]<<" ) + (r:"<<x[3]<<" )\n"; //Show it!
117     }
118
119     cout << "\n\n=== Process for Bezut Coeficients: ===\n"; //The title for the Bezut Process
120     cout << "(a':"<<RealA<<" ) = " << "(a':"<<RealA<<" )(m:1) + "; //The original lineal combination of a
121     cout << "(b':"<<RealB<<" )(n:0)\n"; //The original lineal combination of a
122
123     cout << "(b':"<<RealB<<" ) = " << "(a':"<<RealA<<" )(m:0) + "; //The original lineal combination of b
124     cout << "(b':"<<RealB<<" )(n:1)\n\n"; //The original lineal combination of b
125
126     for (auto x : Data) {                       //For each element in the data
127         cout << "(r:"<<x[3]<<" ) = " << "(a:"<<x[0]<<" ) - "; //Show it (r) = (a) - (b)(q)
128         cout << "(b:"<<x[1]<<" )" << "(1:"<<x[2]<<" ) = "; //Show it (r) = (a) - (b)(q)
129
130         cout << "(a':"<<RealA<<" )(m:"<<x[4]<<" ) + "; //Show it!
131         cout << "(b':"<<RealB<<" )(n:"<<x[5]<<" )\n"; //Show it!
132     }
133
134     cout << "\nSo GCD("<<RealA<<" , "<<RealB<<" ) = "<<abs(a)<<"\n"; //You maybe want to know this
135
136     cout << "So BezutNumbers("<< RealA <<" , "<< RealB <<" ) = "; //You maybe want to know this
137     cout << "("<< LastM <<" , "<< LastN <<" )\n"; //You maybe want to know this
138
139     cout << "So Bezut Indentity: (GCD:"<< abs(a) <<" ) = "; //You maybe want to know this
140     cout << "(a':" << RealA <<" )(m:" << LastM <<" ) +"; //You maybe want to know this
141     cout << "(b':" << RealB <<" )(n:" << LastN <<" )\n"; //You maybe want to know this
142 }
143
144 return Data;                                   //Return the Data
145 }
146
147
148
149
150 /*=====
151 ===== BEST OPTIMIZATIONS OF THIS FOR REAL LIFE =====
152 =====*/
153
154 // ***** LONG DIVISION ALGORITHM *****
155 PairOflli DivisionAlgorithm(lli a, lli b){       //FN: Return the only q and r
156     lli q, r;                                   //Variables, remember {0,|b|} in r
157
158     if (b != 0) {                                //If we have work to do
159         q = a/b;                                  //Get me the floor of this
160         r = a%b;                                  //Get me the reminder
161
162         if (r < 0){                               //If we have a problem

```

```

163         (q >= 0)? q++: q--;           //If q>0 add 1, else sub 1
164         r = a - (b*q);               //Recalculate reminder
165     }
166 }
167 else {q = 0; r = a;}                //b=0, this is always the result
168 return {q, r};                      //Return the info
169 }
170
171
172
173 // ***** GREAT COMMON DIVIDER: EUCLIDEAN EDITION *****
174 ull GCD(lli a, lli b){               //FN: Return GreatCommonDivider of 2 #
175     lli reminder;                   //Lets create a reminder
176
177     while(b != 0){                  //Remember GCD(A,0) = |A|
178         reminder = a % b;           //Get me the reminder of the 2 numbers
179         a = b;                     //Know A=BQ+R -> GCD(A,B) = GCD(B,R)
180         b = reminder;              //Let's calculate GCD(B,R)
181     }
182
183     return abs(a);                  //Get me the A when B is 0 GCD(A,0)=|A|
184 }
185
186
187
188 // ***** LEAST COMMON MULTIPLE *****
189 ull LCM(lli a, lli b){               //FN: Return the LCM of 2 numbers
190     return (abs(a*b)) / GCD(a, b);  //THEOREM: LCM(a, b) = |ab| / GCD(a,b)
191 }
192
193
194
195 /* ***** BEZUT COEFFICIENTS AND GCD *****
196 Info:
197     This function will return a pair so:
198     - The first of the pair is m
199     - The seconf of the pair is n
200
201     So GCD(a,b) = am + bn
202     Tips:
203     - If you need GCD(a,b) and the Bezut Coeficients just
204       ejecute this function and then find the GCD as:
205       GCD(a,b) = am + bn
206
207     - If you need LCM(a,b) and the Bezut Coeficients just
208       ejecute this function and then find the LCM as:
209       LCM(a, b) = |ab| / am + bn
210 */
211 PairOflli BezutCoefficients(lli a, lli b){           //FN: Return GreatCommonDivider of 2 #
212     lli Temporal, q, r, m = 0, n = 1, LastM = 1, LastN = 0; //The variables
213
214     while(b != 0){                                     //Remember GCD(A,0) = |A|
215         auto Step = DivisionAlgorithm(a, b);           //Get the long division
216         q = Step.first;                                //Get values
217         r = Step.second;                              //Get values
218
219         a = b;                                         //Know A=BQ+R -> GCD(A,B) = GCD(B,R)
220         b = r;                                         //Let's calculate GCD(B,R)
221
222         Temporal = m;                                  //Lets save m

```

1.6. Redes Sociales

Si bien este proyecto es puramente académico, la forma en la que tengo para llegar a más personas y que cada día más jóvenes puedan aprovechar los contenidos que yo creo para ellos es a través de las redes sociales, donde la principal (por gran margen) es Facebook, donde en menos de 8 meses he creado una comunidad con mas de 7,000 personas que esperan, opinan y ven los contenidos en todo momento, dando que más del 70 % de los visitantes llegan de estas plataformas.

Mi objetivo al crear esta división del proyecto fue:

- Dar a conocer el Proyecto a más personas
- Demostrar que las redes sociales si se ocupan bien pueden ser una gran herramienta para mejorar y complementar la educación, creando un espacio de dialogo y discusión.
- Compartir fragmentos del trabajo que hago para que incluso si no entran a la página o a Libros de puedan llevarse algo que los ayude con sus estudios y a recuperar el amor a la ciencia.

1.6.1. Evidencias - Fragmentos

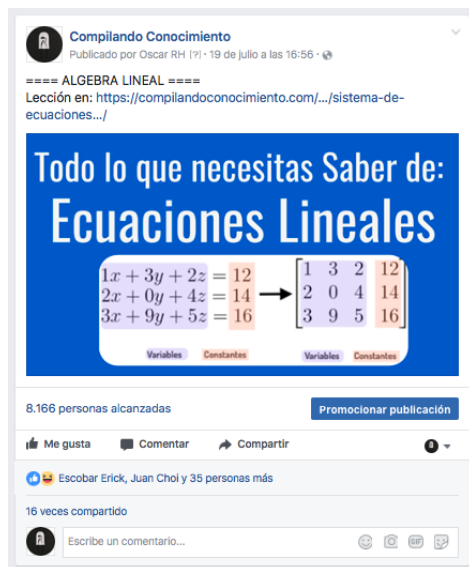


Figura 1.14: Interacciones con el Proyecto en Redes Sociales



Figura 1.15: Interacciones con el Proyecto en Redes Sociales

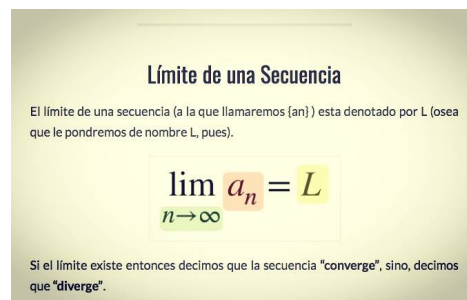


Figura 1.16: Interacciones con el Proyecto en Redes Sociales

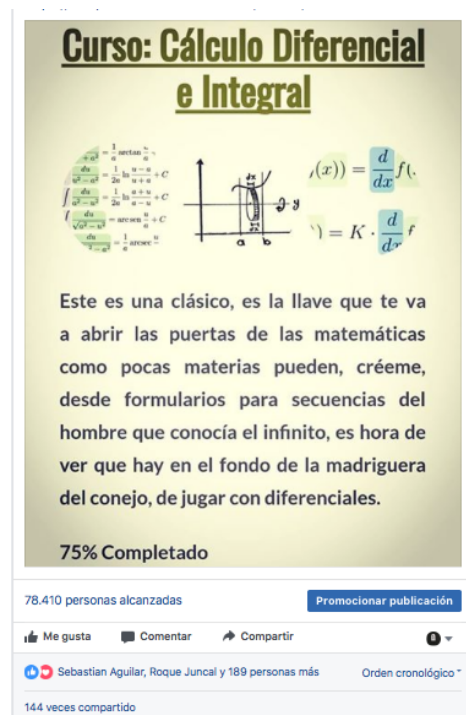


Figura 1.17: Interacciones con el Proyecto en Redes Sociales

1.7. Liberar todo mi contenido bajo la GPL v2

Todo el contenido de este proyecto, desde las imágenes, los PDFS, los códigos libres estan liberados bajo una licencia de software conocida como la GPL o General Public License que es una licencia de código abierto, las ventajas de usar esa licencia son:

Cualquiera de nuestros contenidos tiene las libertades básicas del Software Libre:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puedes ayudar a tus semejantes (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

O visto de otra manera, la licencia GPL:

- Permite la copia, modificación y redistribución del software.
- Proporciona garantía de los derechos del usuario a la copia, modificación y redistribución del software.
- Cualquier patente sobre el mismo debe ser licenciada para el beneficio de todos.
- El software modificado no debe tener costo por la licencia.
- Tiene que incluir el código fuente.
- Los cambios en la licencia deben mantener ciertos términos generales.

1.8. Conclusiones - Razones para Postularme

Antes que nada tengo que agradecer por leer esta carta hasta el final, este proyecto es joven y veo yo en el gran potencial de ayudar a mas personas cada día y de usar las grandes ventajas que nos dan las nuevas tecnologías para hacer de este un lugar mejor, para ayudar a nuestro jóvenes sin importar donde esten o a que casa de estudios pertenezcan.

Busco este Premio para poder darle más convertura a la página y al proyecto en general, busco ayudar cada día a más personas, pero sobretodo, me temo que al ser una única persona que hace este proyecto al mismo tiempo que trabajaba y estudiaba y ahora que voy a estudiar dos carreras simultaneas (Ciencias de la Computación en Facultad de Ciencias de la UNAM (1 semestre) y Sistemas Computacionales en ESCOM-IPN (4 semestre)) me temo que muy difícil para mi darme abasto con todo esto, y quisiera gracias a ese apoyo monetario contratar a más personas y crear así una pequeña organización y que entre todos nosotros nos encargemos de crear contenido de más calidad, crear cursos para mas materias y quizá incluso crear para más plataformas, como por ejemplo Youtube.

Si necesitan más evidencias o conocer mas a fondo el proyecto pueden acceder a:

- CompilandoConocimiento.com
- facebook.com/CompilandoConocimiento
- github.com/CompilandoConocimiento

Capítulo 2

Documentación