PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

FACULTAD DE INGENIERIA

ESCUELA DE SISTEMAS

"DISEÑO Y DESARROLLO DE UN PROGRAMA DE APOYO DIDACTICO PARA EL ANALISIS SINTACTICO Y LOGICO A SER USADO POR LOS PROFESORES DE LA PUCE"

BORJA LOMBEIDA SANTIAGO FELIPE

Trabajo Previo a la Obtención del título de:

INGENIERO EN SISTEMAS

QUITO 2010

Dedicatoria

Desde el día en que nací, existen personas que me lo han dado todo,

Elvia Victoria que me dio la vida, mi primer amor, mi Mamita.

Gonzalo Gustavo que me ha enseñado a ser un hombre y como quien quiero llegar a ser, mi Padre.

Eugenia y Fabián mis hermanos.

Mi Abuelito Julio por darme sus palabras y brindarme su legado.

Cristina por ser mi gran amor y estar a cada instante con su dulzura y belleza. Agustín por darme fuerzas cada día con su sonrisa, su mirada, por tomar con su manito mi vida, a ellos les dedico este trabajo ya que son las personas que más amo en este mundo.

Agradecimiento

A todos los Profesores por su dedicación, en especial a Rafael Melgarejo, a los Secretarios y Personal de la Universidad que hicieron de esta nuestra casa. A los Amigos, las experiencias vividas quedaran grabadas, aprendimos no solo en clases sino fuera de ellas. A mis Padres, Abuelos, Hermanos, a mi Esposa e Hijo, gracias a todos por su apoyo a cada instante, su motivación hizo posible que este proyecto pueda culminarse, son la fuerza y el espíritu de todo lo que hago.

Índice

- Dedicatoria …….2

- Agradecimiento ……..…..…3

- Índice ……...4

- Índice de Tablas ……...……..7

- Índice de Gráficos ……..…….8

1. CAPITULO 1: Introducción ..……9

1.1. Introducción **..**……10

1.2. Marco General **..**…….11

1.3. Utilidad **..**……..…..11

1.4. Objetivos **..**……..…….12

1.4.1. Objetivo Principal **..**……..….12

1.4.2. Objetivos Secundarios **..**……...12

1.5. Justificación **..**……13

1.6. Viabilidad y Alcance **..**…….....14

1.6.1. Viabilidad **..**……...…...14

1.6.2. Alcance **..**……...14

1.7. Metodología **..**…….....14

2. CAPITULO 2: Plataforma y Metodología ……...16

2.1. Flash Macromedia y Adobe Flash ……..17

2.1.1. Características de Flash Macromedia ……18

2.1.1.1. Plataformas soportadas por Flash Macromedia ……...….18

2.1.1.2. Action Script ……..19

2.1.1.2.1. Estructura ……..19

2.1.1.2.2. Clases ……..20

2.1.1.2.3. Ventajas de Action Script 3.0 ……..22

2.2 Metodología ……23

2.2.1. Introducción a la Metodología ……23

2.2.2. La Animación ……24

2.2.3. Programación Extrema ……24

2.2.3.1. Actividades Cliente-Desarrollador ……..25

2.2.3.2. Esquema de Trabajo en Programación Extrema ……26

2.2.3.3. Ciclo de Vida ……28

2.2.3.3.1. Planificación ……29

2.2.3.3.2. Diseño ……30

2.2.3.3.3. Desarrollo ……...31

2.2.3.3.4. Pruebas ……31

2.3. Planificación ……..32

2.3.1. Historias del Usuario ……...32

2.3.2. Plan de Entregas ……..36

2.3.3. Velocidad del Proyecto ……...38

2.3.4. Plan de Iteraciones ……..39

2.3.5. Diagramas para cada Iteración ……...41

2.3.6. Diseño ……..41

2.3.6.1. Diseño de la Interfaz ……41

2.3.7. Manual de Usuario ……..47

3. CAPITULO 3: Teoría de Lógica: Marco Teórico y Ejercicios Prácticos …...50

3. Teoría de Lógica ……..……..51……...104

4. CAPITULO 4: Pruebas y Ejecución del Programa Multimedia Interactivo ..106

4.1. Desarrollo de Pruebas al Programa …….107

4.1.1. Equipos que los Estudiantes de la PUCE tienen a su disponibilidad ……109

4.1.2. Pruebas ……109

4.2. Ejecución y entrega del Programa Multimedia Interactivo …….112

4.2.1. Generalidades ……112

5. CAPITULO 5: Conclusiones y Recomendaciones ……124

5.1. Conclusiones ……125

5.2. Recomendaciones ……126

6. CAPITULO 6: Bibliografía ……129

ANEXOS:

Anexo 1. Diagramas de Iteraciones

Anexo 2. Manual de Usuario

- Índice de Tablas

Tabla 1. ……... 37

Tabla 2. ……... 40

Tabla 3. …… 103

Tabla 4. …… 111

Tabla 5. …… 113

Tabla 6. …… 114

Tabla 7. …… 115

Tabla 8. …… 116

Tabla 9. …… 118

Tabla 10. ……... 119

Tabla 11. ……... 120

Tabla 12. ……... 121

- Índice de Gráficos

Figura 1. ……... 28

Figura 2. ……..…….. 38

Figura 3. ……..…….. 41

Figura 4. ……..…….. 42

Figura 5. ……..…….. 43

Figura 6. ……..…….. 44

Figura 7. ……..…….. 45

Figura 8. ……..…….. 46

Figura 9. ……..…… 108

Figura 10. ……..…….. 109

Figura 11. ……..…….. 112

Figura 12. ……..…….. 114

Figura 13. ……..…….. 117

Figura 14. ……..…….. 119

CAPÍTULO I

INTRODUCCION

1.1 INTRODUCCIÓN

La cátedra dictada por el Dr. Emilio Cerezo no cuenta actualmente con un programa en español de ayuda en la materia de Lógica, salvo la tesis hecha a la par por un compañero de la facultad de ingeniería, dado que nuestra facultad y nuestras materias tienen la capacidad y la versatilidad para desarrollar un programa que pueda ser implementado como ayuda didáctica en materias como la Lógica Proposicional y Análisis Sintáctico en la Universidad se presenta este proyecto con todo el proceso que se ha seguido paso a paso hasta su terminación y puesta a prueba, sin olvidar las respectivas pruebas al software.

El programa cuenta con una parte teórica, otra de ejercicios y la ultima de preguntas, lo que lo vuelve un gran complemento a las clases de Lógica, además de poder usarlo no solo dentro de las aulas de clases sino para que el estudiante pueda tenerlo en sus propios computadores personales, de modo que se intensificara la ayuda que da el programa en el aprendizaje de la materia, pues contiene los temas como los ejercicios; y, dada la portabilidad de un software ejecutable, le brinda la posibilidad de ser corrido en los diferentes sistemas operativos comerciales usados en la actualidad.

Todos los temas que abarca esta tesis están presentes en la teoría ya corregida por el Dr. Emilio Cerezo, lo cual mejora el programa y le brinda mayor entendimiento. El diseño de la página debe contener un estudio en todo su desarrollo; ya que, por estudios realizados por en Diseño Grafico, se asegura que se pueda captar más la atención del estudiante mediante el uso de colores que llamen su atención, sin distraerlo, mejorando el entendimiento de la materia de la que trata el programa.

En la actualidad ha crecido la tendencia al uso de programas destinados a complementar mejor las clases, de forma que el alumno pueda seguir las clases con un programa interactivo, lleno de teoría, ejercicios y autoevaluación, lo que le permitirá un repaso tanto en la universidad como fuera de ella.

1.2 MARCO GENERAL

En el desarrollo del presente programa se usó Macromedia Flash 8, programa que cuenta con permisos, licencias, etc. El trabajo con este programa pretende hacerlo visualmente ergonómico al usuario, visualmente claro está, facilitando su uso y su atractivo para generar un interés grande en las personas que lo utilicen. El programa de Macromedia favorece el dinamismo, la versatilidad, de estudiar lógica, lo cual es necesario para cumplir, tanto los temas a tratarse como las expectativas del usuario en cuanto a interés visual, que es lo que nos interesa. También busca llenar un vacío que es el de la automatización del estudio en diferentes materias y, para el caso que nos compete, respecto de la lógica, que merece tener facilidades como esta.

1.3 UTILIDAD

Desde hace más de medio siglo atrás; el valor que ha ganado el área de la computación y los sistemas ha crecido de manera agigantada, gracias a su aplicabilidad en las demás ciencias.

La ayuda que presta un programa en el estudio de materias que están presente en las bases de otras ciencias, incluyendo las que hemos estudiado en la Facultad de Sistemas, nos

demuestra lo entrelazadas se encuentran, que es la Lógica. Su estudio brindará a los estudiantes la mejor comprensión de sucesos que a través de muchos siglos ha desembocado en la actual diversidad de ramas que la materia tiene, con su historia y los grandes pensadores que forjaron las normas, reglas, guías de esta ciencia en el estudio del pensamiento y los razonamientos válidos.

1.4 OBJETIVOS

1.4.1 OBJETIVO PRINCIPAL:

- Desarrollar e Implementar un programa como ayuda didáctica en las clases de Lógica en la Universidad Católica del Ecuador, aplicando conocidos métodos de programación que facilitarán su desarrollo

1.4.2 OBJETIVOS SECUNDARIOS:

- Detallar claramente los diversos aspectos relacionados con el desarrollo del proyecto, como lo es el estudio previo de lógica, el diseño, la programación, y las pruebas.
- Demostrara la facilidad de aprendizaje con la ayuda de programas interactivos de fácil uso para los estudiantes y profesores.
- Realizar un estudio de diseño previo que garantice que el programa sea ergonómico al usuario.
- Realizar pruebas que validen al programa, para de esta manera mejorarlo.
- Hacer un programa interactivo en base a un estudio de colores que faciliten la atención del usuario del mismo.

1.5 JUSTIFICACIÓN

"Vivir sin filosofar es, propiamente, tener los ojos cerrados, sin tratar de abrirlos jamás" [1]

Dado el hecho de que toda actividad humana tiene, o debe tener, en sí lógica y que la lógica se forma de lo que vivimos a diario, ligados profundamente, nos hallamos en un camino de una sola vía, por lo que la carrera de Ingeniería de Sistemas y la Lógica van de la mano en sus bases además de sus procesos, vinculándolos en forma y fondo casi desapercibido

Y dado que, como ya se ha mencionado, la sistematización de las materias dictadas en la universidad van tomando esta corriente como propia y no tan solo como una herramienta aislada, es preciso contar con la ayuda de este programa en la cátedra dictada de lógica para hacerlo más amigable y entendible ya que lo intangible suele complementarse con un refuerzo material como lo es un libro y dado el tiempo es que vivimos, globalizado y con innumerables facilidades, es menester contar con un programa personalizado para la materia dictada en nuestra universidad, además de merecerla por mantener la elite de alcance académico que guarda celosamente nuestra institución.

[1] Rene Descartes. Filósofo, matemático y científico Francés, fundador del "Racionalismo", La Haye 31 de Marzo 1596 – Estocolmo 11 Febrero 1650

1.6 VIABILIDAD Y ALCANCE

1.6.1 VIABILIDAD

Al contar el desarrollo del programa con la ayuda del Dr. Emilio Cerezo, además de la cantidad de libros existentes en la biblioteca de la Universidad e Internet sobre los temas a tratarse en este estudio, contamos con valiosos soportes que permitirán un eficiente desenvolvimiento de la presente tesis.

Una de las dificultades probables será el que tengamos que aprender la teoría, mas con la reiterada ayuda del Dr. Emilio Cerezo será posible hacerlo.

1.6.2 ALCANCE

La investigación, el estudio y el desarrollo del programa terminaran con las pruebas con las cuales quedara terminado su aplicación.

El programa contempla un estudio de diseño, de contenido teórico, de ejercicios y de prueba; es decir todos los componentes que debe tener una clase normal.

1.7 METODOLOGÍA

Para el desarrollo de este programa se utilizará Macromedia Flash 8 con los equipos necesarios para el desarrollo, como son una computadora portátil y una de escritorio. La información necesaria para el correcto cumplimiento de proyecto será obtenida de varias fuentes, entre ellas los foros de información, libros, ayuda de especialistas en la materia, y especialistas en el uso de Flash 8, junto con la mayor fuente de información y corrección que

en este caso es el aporte del Dr. Emilio Cerezo. Adicionalmente, para el desarrollo e implementación – pruebas, ejemplos, etc. –, se usara la Programación Extrema.

CAPITULO 2.

PLATAFORMA y METODOLOGÍA

CAPITULO 2. PLATAFORMA y METODOLOGÍA

PLATAFORMA

2.1. FLASH MACROMEDIA y ADOBE FLASH

La aplicación Adobe Flash, orientada a la animación, trabaja sobre fotogramas y permite producir material interactivo en diferentes plataformas. Actualmente Flash está siendo distribuido por Adobe Systems[2].

Entre las características utilizadas por esta herramienta podemos mencionar que usa gráficos vectoriales *los cuales son una imagen digital formada por objetos geométricos independientes (segmentos, polígonos, arcos, etc.) cada uno de ellos definido por distintos atributos matemáticos de forma, de posición, de color, etc.*[3] y de imágenes ráster, *también llamadas bitmap, imagen matricial o pixmap, es una estructura o fichero de datos que representa una rejilla rectangular de pixeles o puntos de color, denominada raster, que se puede visualizar en un monitor de ordenador, papel u otro dispositivo de representación* [4], sonido, código de programa, flujo de video y audio bidireccional. Al interior de este conjunto, Flash es el entorno y Flash Player es el programa de máquina virtual *sw que actúa como un computador real, con programas, etc.*[5] utilizado para ejecutar los archivos generados con Flash. Para el presente programa echaremos mano de ambos, en el desarrollo y la presentación respectivamente. Contando con la facilidad que brindan las múltiples funcionalidades de Flash, tanto el de Macromedia como el de Adobe, utilizaremos este

[2] *Adobe Systems Incorporated* es una empresa de SW con sede en San José California, USA, programas de edición de páginas web, video e imagen digital.
http://es.wikipedia.org/wiki/Adobe_Systems_Incorporated

[3] http://es.wikipedia.org/wiki/Gr%C3%A1fico_vectorial

[4] http://es.wikipedia.org/wiki/Gr%C3%A1fico_rasterizado

[5] http://es.wikipedia.org/wiki/Gr%C3%Flash_Player

programa para subir documentos de flash*(.swf)* a la internet, o ser reproducidos por un reproductor flash.

Junto con la ayuda de Flash, la Internet, ha permitido a sus navegadores beneficiarse de propagandas y animaciones llamativas, educativas y de entretenimiento.

En versiones anteriores, Macromedia amplió a Flash más allá de las animaciones simples, y lo convirtió en una herramienta de desarrollo completa para crear principalmente elementos multimedia e interactivos para Internet.

Hasta el año 2005, Flash perteneció a la empresa Macromedia y fue conocido como Macromedia Flash®. Posteriormente fue adquirido por Adobe Systems. Desde entonces se conoció con su actual denominación, *Adobe Flash*, y amplió con ello su portafolio de productos dentro del mercado.

2.1.1CARACTERISTICAS DE FLASH MACROMEDIA.

2.1.1.1 PLATAFORMAS SOPORTADAS POR FLASH MACROMEDIA.

De realizarse un programa en esta herramienta podremos correrla o verla en las diferentes plataformas soportadas por Flash son: Windows, Mac y GNU/Linux. La versatilidad prestada nos brinda un amplia gama de posibilidades de trabajar en equipos destinados al diseño, como es el caso de un equipo MAC, y poder desplegarlo en un entorno Windows.

2.1.1.2 ACTION SCRIPT

Flash usa ActionScript como su lenguaje propio, el cual maximiza las funcionalidades de esta herramienta, aumentando así el grado de contenido interactivo en las películas de Flash, que permite al diseñador o desarrollador añadir nuevos efectos o incluso construir la interfaz de usuario de una aplicación compleja. La nueva versión de ActionScript (3.0) se ha orientado a objetos, prescindiendo de prototipos y enfatizando las clases, la sintaxis se ha visto modificada y enriquecida. Más la versión que manejamos en este programa es una versión inferior, sin verse afectada la posibilidad de construir un programa eficiente y amigable al usuario.

2.1.1.2.1 ESTRUCTURA

Flash está compuesto por Objetos que en programación *es la unidad que en tiempo de ejecución realiza las tareas de un programa, y a un nivel más básico se define como la instancia de una clase*[6], con su respectiva ruta dentro de la extension *swf*. En la versión ActionScript, cada uno de ellos pertenece a una clase que en flash son declaraciones o abstracciones de objetos, es las declaración total de este; características, funcionalidades, etc., (MovieClip, Botones, Vectores –Arreglos de datos-, etc.), que contiene Propiedades y Métodos.

- *Propiedades*: Dentro del archivo raíz de esta clase, se declaran como variables (alpha, useHandCursor, length).

- *Métodos*: Dentro del archivo raíz de la clase, están declaradas como funciones (stop( ), gotoAndPlay( ), getURL( )).

[6] http://es.wikipedia.org/wiki/Gr%C3%Action_Script

2.1.1.2.2 CLASES

Algunas de las clases más usadas en ActionScript son:

- Accessibility (nivel superior)
- Array (instancias)
- Boolean (instancias)
- Button (instancias)
- Capabilities (nivel superior)
- Color (instancias)
- ContextMenu (instancias)
- ContextMenuItems (instancias)
- Date (instancias)
- Error (instancias)
- Key (nivel superior)
- LoadVars (instancias)
- Math (nivel superior)
- Mouse (nivel superior)
- MovieClip (instancias)

- MovieClipLoader (instancias)
- NetConnection (instancias)
- NetStream (instancias)
- Number (nivel superior)
- Object (instancias)
- PrintJob (instancias)
- Selection (nivel superior)
- Sound (instancias)
- Stage (nivel superior)
- String (instancias)
- StyleSheet (instancias)
- System (nivel superior)
- TextField (instancias)
- TextFormat (instancias)
- XML (instancia)
- XMLSocket

Para el desarrollo del siguiente programa usaremos algunas de estas clases; mas, dado que buscamos un desarrollo optimo, dejaremos de lado detalles que escapan del alcance del proyecto para futuras mejoras y ampliaciones suyas.

2.1.1.2.3 VENTAJAS DE ACTIONSCRIPT 3.0

La nueva versión del lenguaje de Abobe que estamos empleando, ActionScript 3.0, tiene posibilidades de creación de scripts, que son uno o varios conjuntos de instrucciones que permiten automatizar tareas, que se han diseñado para facilitar la construcción de aplicaciones muy complejas con grandes conjuntos de datos y bases de código reutilizables y orientadas a objetos.

Para futuras actualizaciones a nuestro programa podremos usar el código actual de ActionScript, aunque tomemos datos que estén en las versiones antiguas de flash como lo son *ActionScript 1.0* y *ActionScript 2.0*. Flash Player 9 admite *AVM1*[7] pues es compatible con contenido heredado de versiones anteriores.

Estas propiedades de nuestra versión brinda un avance a futuro pues, gracias a la metodología que utilizaremos, programación extrema, garantiza que las nuevas versiones del programa se beneficien tanto de código como del diseño, y la migración de código actionscript se llevaría a cabo en forma muy sencilla.

[7] Es la máquina virtual de ActionScript que se utiliza para ejecutar ActionScript1.0 y ActionScript 2.0. http://es.wikipedia.org/wiki/Gr%C3%Action_Script

2.2 METODOLOGÍA

2.2.1 INTRODUCCIÓN A LA METODOLOGÍA

Con el paso de los años los sistemas han evolucionado en función del desarrollo de las facilidades de los usuarios, y es así que a la par se han creado distintos programas para desarrollar programas, para diseñar, para modelar, para simular, etc. Tratando de esta manera de emular el mundo real, pero, incluyendo reglas que cada vez lo vuelven más proactivo evitando futuros problemas, corrigiéndolos y mejorando tanto su presentación como su parte medular en sí.

Todo va de la mano sin dejar escapar ningún punto débil, por lo que la metodología a implantarse durante el desarrollo de un proyecto debe ser el que mejor se acople a éste, para nuestro caso requeriremos de Programación Extrema, una joven pero fuerte manera de hacer las cosas, introduciendo revolucionarias reglas y métodos que garantizan el desenvolvimiento, correcciones y mejoras a nuestro programa.

¿Por qué usar Programación Extrema?, pues por su flexibilidad, ya que nos permite ser dueños de nuestros tiempos y de nuestra organización, plantearnos posibles caminos y desarrollarlos, corregirlos paralelamente mientras se programa junto con el usuario o dueño de la futura aplicación interactiva. Esto nos ofrece la fortaleza de la mejora continua, que en los tiempos actuales ayudará a que nuestro programa sea mejorado a través del tiempo no solo por quien lo desarrollo por primera vez sino por alguna otra persona en un futuro, si amerita algún cambio por mas grande o pequeño que este sea.

2.2.2 LA ANIMACIÓN

La animación pertenece al estilo de un programa y su aporte es dar la sensación de movimiento a dibujos o imágenes tomadas a objetos reales y actores. Incluye desde el más minúsculo movimiento hasta la creación de una presentación más real y en tercera dimensión, que exige un trabajo muy complicado.

La animación, y más aún la que podemos hacer en *Flash*, pretende dar a imágenes y dibujos una sensación de movimiento y el grado de realismo va dado por la minuciosidad de manejo del movimiento en corto tiempo, lo cual puede realizarse mediante el buen uso de los fotogramas y herramientas de diseño. En tercera dimensión, el trabajo se vuelve más intenso dado el nivel de detalle que requiere.

2.2.3 PROGRAMACION EXTREMA

El desarrollo de software tiene varios enfoques, uno de ellos es la Programación Extrema, liviano y radical, con una disciplina de trabajo enfocada a las tareas diarias y las necesidades del cliente, ambas paralelas y complementarias, que en el día a día logran del producto un porcentaje altísimo de eficiencia.

Existen algunas prácticas como XPTestFirst1, XPRefactoring2, las cuales se enfocan en un seguimiento al trabajo que se hace mediante un control permanente basado en los requerimientos del cliente, ya que, en esta propuesta de desarrollo, la interacción es mucho más fuerte. Aquí la realimentación es rápida y la comunicación con el cliente eleva el valor del trabajo de los desarrolladores o del desarrollador, que puede ser uno solo; aunque esta técnica está basada más bien en ayudar a un equipo de desarrolladores, comprobando así lo que está bien y corrigiendo todo lo que el cliente no desea tener o no es necesario.

La maximización del trabajo consiste en simplificar la programación, y el diseño es uno de los principales puntales de la Programación Extrema. Esta no es una metodología en sí, sino un proceso que puede emplearse en cualquier tema de desarrollo y en el lenguaje que se desee. Consta de pasos a seguir para emplear de mejor manera los recursos como el tiempo de pruebas y de instalación, entre otros.

2.2.3.1 ACTIVIDADES Cliente - Desarrollador

En un equipo de programación extrema hay una división fundamental entre los roles del Cliente y el Programador. El Cliente es dueño del «qué se obtiene» y el Programador es dueño del «cuánto cuesta», como ya lo hemos mencionado. ***1***

1. De Wikipedia.- Programación Extrema.

Entre las actividades delegadas al Cliente y bajo decisión del mismo están:

- Alcance lo que el sistema debe hacer.
- Prioridad de lo más importante.
- Composición de los elementos qué deben componer la entrega para que sea útil.
- Fechas: para cuándo se necesitan las entregas.

Estas actividades y las ya descritas se complementan con las que debe realizar el programador y bajo el mando de decisión de este están:

- Tiempo estimado para agregar una característica.

- Consecuencias técnicas: el Programador explica las consecuencias de las decisiones técnicas, pero el cliente es quien toma la decisión.

- El proceso: cómo trabajará el equipo.

- Cronograma detallado dentro de la iteración.

La Programación Extrema no contempla el caso de nuevos requerimientos que estén fuera del alcance del programa emprendido, ya que se vuelve absurdo comprometerse con nuevos retos si antes no están completados al 100% los que ya se han planificado: programa esté terminado y entregado. Por ello, para el presente proyecto nos basaremos en lo ya indicado en el Capitulo 1 y en el Plan de Tesis.

Así pues, se debe prever cualquier traspié mediante el buen uso de herramientas antes de comenzar el desarrollo del proyecto, como un buen manejo de tiempos en *Project*, dejando un espacio extra para solventar inconvenientes en cada uno de los pasos necesarios en el transcurso del presente proyecto.

2.2.3.2 ESQUEMA DE TRABAJO EN PROGRAMACION EXTREMA.

En Programación Extrema se trabaja bajo 3 capas, cada una de las cuales conlleva principios que deben ser respetados y que sirven como una guía para que estas descollen en su maximización, y son:

1-Programación

1.1 Diseño Simplificado (XpSimpleDesign)

1.2 Comprobaciones (XpTesting)

1.3 Modificación o reemplazo de código (XpRefactoring)

1.4 Estandarización en la Codificación (XpCodingStandards)

2-Practicas de equipo

2.1 Integración Continua XpContinuousIntegration

2.2 Marco Conseptual, MetaforaXpMetaphor

2.3 Estandarización en la Codificación (XpCodingStandards)

2.4 Trabajo en Horario (XpFortyHourWeek)

2.5 Programación en Pareja (XpPairProgramming)

2.6 Pequeñas Entregas (XpSmallReleases)

3-Procesos

3.1 Cliente trabajando conjuntamente (XpOnSiteCustomer)

3.2 Pruebas (XpTesting)

3.3 Pequeñas Entregas (XpSmallReleases)

3.4 Planificación en Entregas (XpPlanningGame)

Acompañan a este esquema unas Prácticas de ayuda y seguimiento:

- Simplicidad.
- Comunicación.
- Realimentación.

- Coraje.

Debemos recalcar que los *testing* propuestos por Programación Extrema son de comprobación con antelación: antes de adoptar definitivamente la rutina a seguirse durante la programación, así se prevee problemas y se delimita alcances para conjuntamente con el cliente validar todos los puntos a desarrollarse. De esta manera los chequeos serán frecuentes y elevarán el nivel de rendimiento y desempeño del programa

2.2.3.3 CICLO DE VIDA

Una vez definida la metodología veremos el ciclo de vida, el cual nos permitirá agrupar los diferentes pasos que se deben considerar a través del tiempo que dure el desarrollo e implementación del programa, estos cuatro pasos engloban la planificación a ser respetada durante el desarrollo y entrega del producto, el diseño que es fundamental en la interacción con el Cliente, el desarrollo de la programación que contempla los 2 puntos anteriormente mencionados y las pruebas donde ya todo estará terminado exceptuando algunos cambios que puedan darse a petición del dueño del programa.

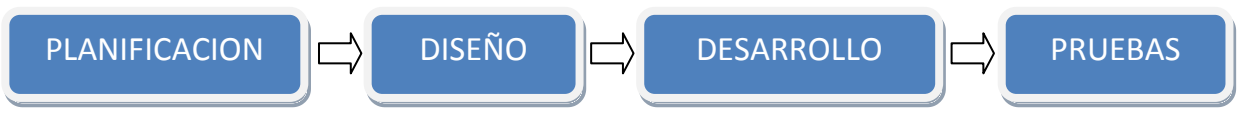


Figura 1

2.2.3.3.1 PLANIFICACION

A este nivel se medirán los alcances que enfocaran que debe hacer el programa y que debe contener, además de definir hasta que puntos y con qué herramientas, el proceso cuenta, como ya se ha mencionado antes en las *Actividades* de la ayuda del Cliente quien indicará lo que requiere y el desarrollador explicará cómo se lo hará y en cuanto tiempo.

La parte técnica informara paulatinamente de los avances o cambios que conjuntamente con el Cliente pulirán paso a paso lo que merezca cambios o aprobaran los avances.

Dentro de la Planificación tenemos pasos que deben llevarse a cabo:

1. Redactar historias de usuario: Las historias de usuario tienen el mismo propósito que los casos de uso, pero no son lo mismo. Las escriben los propios clientes, tal y como ven ellos las necesidades del sistema. Por tanto serán descripciones cortas y escritas en el lenguaje del usuario, sin terminología técnica.

2. Crear plan de entregas: El plan de entregas se usará para crear los planes de iteración para cada iteración.

3. Controlar la velocidad del proyecto: La velocidad de proyecto se usa para determinar cuántas historias de usuario pueden ser implementadas antes de una fecha dada (tiempo), o cuánto tiempo es necesario para llevar a cabo un conjunto de historias (alcance).

4. Dividir el proyecto en iteraciones.

5. Planificar cada interacción antes de comenzarla.

6. Rotar al personal.

7. Reunión de seguimiento diaria.

8. Corregir la propia metodología.

2.2.3.3.2 DISEÑO

Los puntos a con considerarse como ayuda y fortaleza en este punto favorecerán al correcto diseño del programa, teniendo los siguientes:

1. Simplicidad: Es mejor implementar uh diseño sencillo susceptible de futuras mejoras o correcciones que uno complejo ya que esto cambiaria el enfoque principal del proyecto.

2. Elegir una metáfora: La metáfora servirá para llevar un historial que detalle el funcionamiento del sistema habiendo una historia por cada ítem, documentándole y dándole un seguimiento, así evitara redundancia en nombres y llevara coherencia con lo que ya se esta haciendo, permitiendo categorizar por ejemplo botones, líneas de código, entre otras.

3. Usar tarjetas CRC: Una tarjeta CRC representa un objeto, ya descrito antes, el mismo que tiene propiedades y métodos. Como titulo de la tarjeta se colocara el nombre del objeto, las responsabilidades se colocan a la izquierda, mientras que las clases que se implican en cada responsabilidad a la derecha, con su respectivo requerimiento.

4. Crear soluciones puntuales reduciendo el riesgo de que se genere un problema al tratar de controlar uno previamente generado.

5. No añadir funcionalidades en las primeras etapas hasta dejar sentadas las bases de un buen diseño.

2.2.3.3.3 DESARROLLO

Las características en este punto son mas bien de orden, sin ayudas de tarjetas u otro soporte.

1. El cliente debe estar siempre disponible.
2. Estandarizar la escritura de código.
3. Desarrollar la unidad de prueba primero: la implementación será más rápida.
4. Todo código debe programarse en pareja.
5. Solo una pareja integrará el código.
6. Todo el código es común a todos: cualquiera puede contribuir al desarrollo desde cualquier parte del proyecto basado siempre en los estándares.
7. Dejar las optimizaciones para el final, reduciendo así el nivel de errores.
8. No trabajar más de 40 horas semanales.

2.2.3.3.4 PRUEBAS

1. Todo código realizado debe estar acompañado de una unidad de prueba.
2. Todo el código debe pasar las unidades de prueba antes de ser implementado.
3. Ante un fallo se crea una unidad de prueba, de no existir previamente.

4. Se debe ejecutar pruebas de aceptación publicando los resultados, estos ya son creadas a partir de las historias de usuario.

2.3 PLANIFICACIÓN

2.3.1 HISTORIAS DEL USUARIO

HISTORIA 1

Como bienvenida al ejecutar el programa debe visualizarse una pantalla que conste de información básica respecto al presente trabajo: "Programa Multimedia Interactivo para el Desarrollo de la Lógica y Análisis Lingüístico", junto con un menú donde se pueda ingresar a revisar la materia y los ejercicios. Finalmente información de quien lo hizo y un vínculo donde consten las personas que realizaron el programa.

HISTORIA 2

Se creara una pantalla desplegada desde la pantalla principal donde constaran el programador, y mentalizador del "Programa Multimedia Interactivo para el Desarrollo de la Lógica y Análisis Lingüístico" con una síntesis de la carrera de cada uno de ellos, estudios y demás.

HISTORIA 3:

En cada tema a tratarse, dentro de "Lógica" por separado, deberán constar; la teoría (junto con ejemplos), ejercicios y evaluación. La información tendrá un desglose total en un vinculo mientras que en la teoría se la vera en resumen. Además los ejemplos se desplegaran solos y los ejercicios serán realizados por el usuario del programa.

Los ejercicios junto con los ejemplos no pueden ser mas de 3 por tema dependiendo del tema que se está tratando ya que para aclarar dudas por la dificultad del tema se tratara de involucrar más ejemplos, Los ejercicios y ejemplos están ya resueltos en la parte teórica del proyecto (Capitulo 3).

Al final de cada tema estará la evaluación y al igual que los ejercicios y ejemplos no será mas de 3 ejercicios los mismos que serán similares a los ya expuestos garantizando que no tendrán una dificultad significativa sino mas bien una ayuda para conocer las fortalezas y las debilidades, logrando incentivar las partes puntuales de la teoría que se quiere mejorar en comprensión. La evaluación indicara si es correcto e incorrecto, mas no calificara en puntuación.

HISTORIA 4:

El programa facilitara el movimiento de una parte de la teoría a otra, la navegación será fácil con la ayuda de los botones además de respetar el desglose por capítulos, menús y submenús. Podrá volver al inicio del programa o a la pantalla anterior.

HISTORIA 5:

En la evaluación no se podrá volver atrás, ya que de esta manera no podrá corregir la pregunta y sabrá en que esta fallando y volver a la teoría para reforzar un tema puntualmente.

HISTORIA 6:

Toda la teoría podrá ser revisada como un documento .PDF

HISTORIA 7:

El diseño del programa en las pantallas mantendrá la sencillez para no complicar al estudiante, pero con lo necesario para su correcta navegación dentro del mismo.

HISTORIA 8:

La instalación del programa se ve beneficiada ya que al ser un archivo ejecutable podrá cumplir con uno de los objetivos y es que sea transportable, no requerirá de el programa motor, Macromedia, tan solo ejecutarlo como ya se lo menciono.

HITORIA 9:

Se deberá desarrollar un manual de instrucciones donde el usuario final conozca detalladamente como navegar dentro del programa para facilitar su uso.

Historia 10:

Aparte de la teoría, ejercicios y evaluaciones, se deberá incluir algunas animaciones que capten la atención del usuario.

Historia 11:

Los colores de las pantallas y la distribución deben respetar a un orden determinado y a un estudio de colores, que aunque mantenga la sencillez también ayudara a captar la atención del estudiante.

2.3.2 PLAN DE ENTREGAS

El desarrollo y entrega del programa estará planificado terminarlo en 50 días lo que daría un total de 7 semanas, calendarizando así las siguientes entregas en el tiempo estipulado:

SEMANA	ACTIVIDAD	ENCARGADOS
1	Entrega de escenas: Menú Principal, Créditos terminadas. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo
2	Entrega de Menús por cada tema. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo
3	Entrega de escenas para tema 1 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo
4	Entrega de escenas para tema 2 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo

5	Entrega de escenas para tema 3 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo
6	Entrega de escenas para tema 4 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo
7	Entrega del producto multimedia finalizado y probado. Revisión de: diseño, animaciones, código, cambios o mejoras.	Desarrollador: Santiago Borja Usuario: Emilio Cerezo

Tabla 1

2.3.3 VELOCIDAD DEL PROYECTO

Contando las implementaciones de las historias según el tiempo de duración que tarden se tomaría en cuenta lo siguiente:

HITORIAS	TIEMPO
Mínimo 3	Cada 2 semanas

Figura2

El grafico nos dice que cada dos semanas se debería llevar a culminación por lo menos tres historias, lo cual puede variar, en todo caso se aceleraría sea cual sea la respuesta, eso si no existiera problemas y se probara la eficacia de cada avance para continuar con el siguiente.

Las meta es desarrollar lo mas pronto posible basándonos en lo que nos a indicado la *Programación Extrema*, tratando de corregir cualquier error para que no se acumulen y decaiga la calidad del producto terminado. De haber problemas se deberá medir el tiempo que nos toma la realización de cada avance y modificar las fechas de entrega.

2.3.4 PLAN DE ITERACIONES

El plan de entregas deberá respetar en lo posible los alcances por semana, teniendo así 7 iteraciones una para cada semana, así:

SEMANA	ACTIVIDAD	ITERACIÓN
1	Entrega de escenas: Menú Principal, Créditos terminadas. Revisión de: diseño, animaciones, código, cambios o mejoras.	r0
2	Entrega de Menús por cada tema. Revisión de: diseño, animaciones, código, cambios o mejoras.	r1
3	Entrega de escenas para tema 1 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	r2
4	Entrega de escenas para tema 2 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	r3

5	Entrega de escenas para tema 3 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	r4
6	Entrega de escenas para tema 4 teoría, ejercicios y autoevaluación listos. Revisión de: diseño, animaciones, código, cambios o mejoras.	r5
7	Entrega del producto multimedia finalizado y probado. Revisión de: diseño, animaciones, código, cambios o mejoras.	r6

Tabla 2

2.3.5 DIAGRAMAS PARA CADA ITERACIÓN

Los diagramas están en el ANEXO UNO.

2.3.6 DISEÑO

2.3.6.1 DISEÑO DE LA INTERFAZ

ESCENA MENÚ PRINCIPAL, BINVENIDA Y CREDITOS

Figura 3

Elementos de la escena:

- Fotografías en movimiento continuo de la Universidad (PUCE).
- Menú que contiene los cuatro botones de los distintos temarios.
- Baton créditos permite abrir la escena con los datos de los creadores del programa.
- Animación del titulo.

ESCENA CRÉDITOS

Figura 4

Elementos:

- Fotografías de los realizadores del Programa y estudio de Lógica.
- Cuadros con leyendas referentes a de los personajes, cada uno bajo cada fotografía, respectivamente.
- Botones para adelante y atrás bajo cada una de las fotografías, permite desplegar mas información de los personajes.
- Botón "salir" permite regresar a la escena principal, sin necesidad de cerrar el programa.
- Botón adelante que permite avanzar en la teoría en la que se esta navegando, en los distintos fotogramas, el mismo desaparece si estamos en el ultimo de los fotogramas de cada uno de los temas.
- Botón atrás permite retroceder en los temas y fotogramas, desaparecerá si estamos en el primer fotograma de cada uno de los tema

ESCENA DE CONTENIDOS DE CADA UNO DE LOS TEMAS

Figura 5

Elementos:

- Titulo de la escena
- Cuadro de botones.
- Tres botones, dependiendo del tema pueden variar y ser mas botones, que permiten navegar hacia los diferentes temarios.
- Botón "menu" permite regresar a la escena principal, sin necesidad de cerrar el programa.
- Botón adelante que permite avanzar en la teoría en la que se esta navegando.

ESCENA DE TEORÍA E HISTORIA DE LOGICA

Figura 6

Elementos:

- Leyenda del tema que identifica en que escena se esta navegando.

- Cuadro de teoría, animaciones, botones y ejemplos: aquí se ubicara la descripción teórica del tema presente además de algún botón que interactuara con un ejercicio, y distintas animaciones.
- Botón "menu" permite regresar a la escena principal, sin necesidad de cerrar el programa.
- Botón adelante que permite avanzar en la teoría en la que se esta navegando, en los distintos fotogramas, el mismo desaparece si estamos en el ultimo de los fotogramas de cada uno de los temas.
- Botón atrás permite retroceder en los temas y fotogramas, desaparecerá si estamos en el primer fotograma de cada uno de los tema

ESCENA DE EJERCICIOS PRÁCTICOS

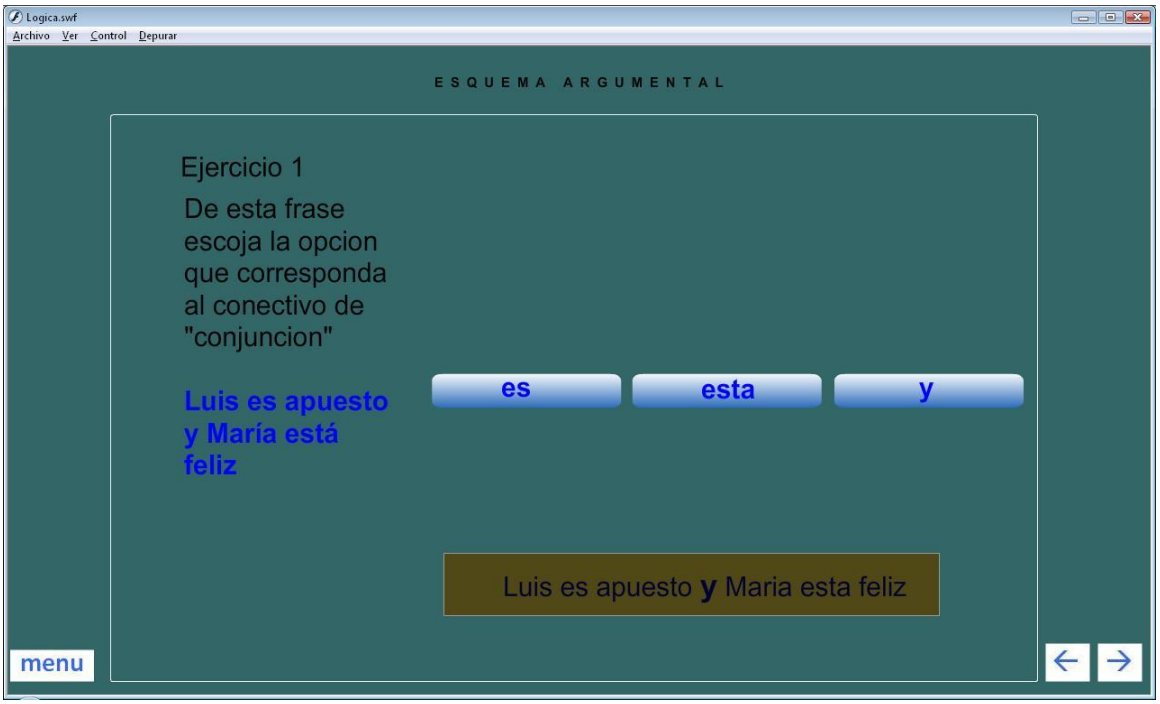


Figura 7

Elementos:

- Leyenda del tema que identifica en que escena se esta navegando.

- Cuadro en donde constan los botones y las leyendas del ejercicio
- Leyenda del ejercicio planteado
- Descripción del Ejercicio
- Botones opción primer fotograma de cada uno de los tema
- Cuadro que contiene la ayuda para el ejercicio
- Botón para avanzar o retroceder en fotogramas
- Botón "menu" que permite regresar a la escena principal sin necesidad de cerrar el programa.

ESCENA DE AUTOEVALUACIÓN

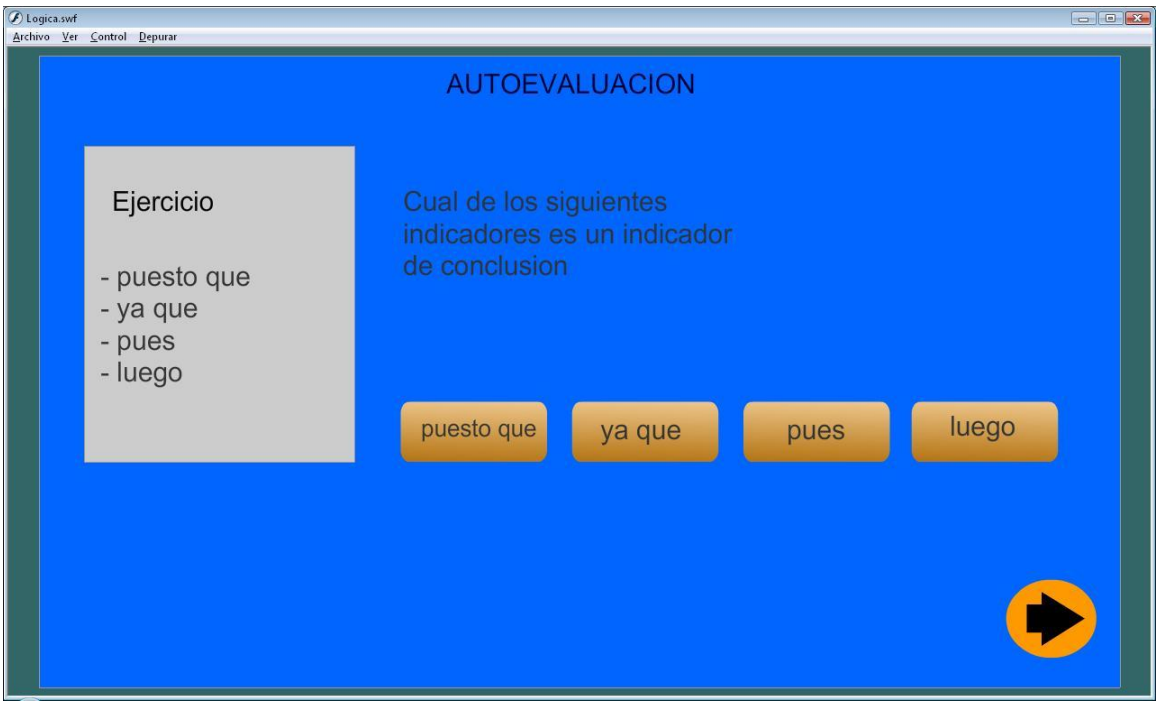


Figura 8

Elementos:

- Leyenda del tema Autoevaluación.
- Cuadro de descripción del ejercicio

- Leyenda de la pregunta
- Botones con las opciones a escoger
- Botón para avanzar en pregunta de la autoevaluación.

o Nota: Una vez empezada la autoevaluación los botones para salir están deshabilitados hasta que termine la autoevaluación, una vez escogida la respuesta se habilitara el botón siguiente y así hasta poder terminar con las preguntas, donde podremos salir de la autoevaluación.

2.3.6.2 ESTANDARES DEL PRODUCTO

Para llevar un buen orden en la ejecución del programa debemos considerar la estandarización en los nombres de las variables, botones y todos los elementos que conforman el programa, para estandarizarlos tomaremos el nombre y la funcionalidad que debe realizar dicho elemento sea este una clase o un objeto, por ejemplo el botón que nos permita navegar dentro de la escena de historia debe componer su nombre del tipo de elemento y el lugar donde lo llevara, así quedara compuesto; btnHistoria.

Tendremos normas para cada uno de los siguientes elementos:

Botones

El nombre se compondrá de el tipo del elemento; botón pero tomando solo las consonantes *"btn"*, y de la acción que realiza el mismo empezando este nombre con mayúscula.

Ejm.

btnCreditos

Este botón nos serviría para ingresar a la escena que contiene los datos de los Creditos.

Movie Clip

Debemos crear el nombre referencial para MovieClip, por ejemplo: mvcAccionaAnimacion, "mcv" indica el objeto, movie clip;la accion realizada "Acciona", "Animacion", la acción que realizara dicho objeto.

Gráficos

Para los gráficos hay una consideración especial, ya que los guardaremos en la Biblioteca de Flash, su nombre debe hacer referencia directa al objeto a ser representado. Así: para la fotografía de autor del programa que es una imagen que la guardaremos en biblioteca su nombre será; "autor1"

Escenas

La escena debe contener el nombre principal de la hoja, por ejemplo la escena principal se llamara "principal".

Frame

Los frames van a ser muy usados en ActionScript, ellos vienen nombrados por numeración.

Nombres de las capas

El nombre de la capa deberá ser lo que contengan, así para la capa que contiene la historia, la llamaremos "historia".

Nota: Teniendo en cuenta que el programa es susceptible de futuras mejoras o cambios se realizaran los mismo respetando los estándares previamente definidos.

2.3.7 MANUAL DE USUARIO:

Este manual esta hecho para facilitar el uso del programa, indicando las características básicas del mismo y las partes que lo constituyen.

VER ANEXO 2: MANUAL DE USUARIO

CAPITULO 3.

TEORIA DE LÓGICA:

MARCO TEÓRICO Y EJERCICIOS PRÁCTICOS

CAPÍTULO 3.

TEORIA DE LOGICA: MARCO TEORICO Y EJERCICIOS PRACTICOS

Lógica

Definición.- Es la ciencia del **Razonamiento** *bien construido*: estudia las propiedades del razonamiento correcto a fin de distinguirlo del incorrecto.

Todo **razonamiento** se construye para defender una opinión discutible. Existen variadas formas de razonamientos, dado que el ser humano tiene diversas maneras de ordenar sus afirmaciones (llamadas premisas) para apoyar la opinión que se quiere defender ante los demás (que es la conclusión a la que se llega).

La lógica dictamina si un razonamiento es legítimo (se rige por reglas que permiten obtener conclusiones verdaderas a partir de premisas verdaderas) o ilegítimo.

Ej.-

"Todas las aves son reptiles y el pato Donald es un ave, por lo tanto el pato Donald es un reptil".

En el ejemplo vemos claramente las 2 premisas y la conclusión a la que se llega a partir de estas, claro esta que son erróneas lo que desembocara en un error.

En base a lo mencionado antes diremos de la Lógica que:

- Consiste en el estudio de métodos y principios que se usan para distinguir entre lo que es y no es un razonamiento correcto o legítimo.

- Proporciona técnicas para verificar la corrección de razonamientos de diferentes tipos, pues toda razón o argumento es un pensamiento pero no todo pensamiento es un razonamiento.

Y a continuación veremos todas estas reglas, métodos y principios.

Razonamiento

Definición.- El razonamiento es una forma especial de organizar el pensamiento, por la cual la mente realiza inferencias, es decir, extrae conclusiones a partir de premisas. Quien emite un razonamiento lo utiliza con el propósito de probar una tesis ante los demás. Y justamente este es el principal indicador de la conclusión de un razonamiento: la afirmación que se defiende, que se trata de probar o demostrar.

No toda serie de proposiciones constituye un razonamiento; pues la construcción de este exige que se cumplan algunos requisitos: el razonamiento es un conjunto de proposiciones en el cual, una de ellas –denominada conclusión– se sigue de otra u otras en las que se apoya. Estas afirmaciones que *ayudan a fundamentar* la verdad de *la conclusión* se llaman

premisas. Otro modo de decirlo es que las premisas son *puntos de partida* que sirven para terminar afirmando la tesis que nos interesa defender, la **conclusión**, que es el *punto de llegada*. Se supone que las premisas son afirmaciones verdaderas.

Inferencia.- *Proceso simbólico* (*mental*) *por el cual se llega a una proposición* (*la conclusión*) *que se afirma sobre la base de una o más proposiciones* (*las premisas*) *que se presentan como punto inicial del proceso.*

Volviendo al Razonamiento, todos ellos se de al menos dos proposiciones: la conclusión y una o más premisas.

Si usamos las abreviaturas: **PREMISA** (= P) y **CONCLUSION** (= C), las afirmaciones que componen un razonamiento pueden organizarse de múltiples maneras:

Estructuras.-

1. [C P]

2. [P P C] (la forma más ordenada)

3. [C P P]

4. [P C P]

5. [. . . P C P . . .]

Términos

Afirmación Enunciado o **Proposición**.- Es toda frase que cumple una *función informativa*, pues con ella queremos indicar cómo es una situación: toda afirmación se lanza con la PRETENSIÓN de ser verdadera, de coincidir con lo que es la realidad. Tanto la conclusión como las premisas de los razonamientos son afirmaciones o proposiciones; por lo tanto pueden ser verdaderas o falsas, es decir, estar de acuerdo o no con la realidad de la situación a la que se refieren.

Toda *proposición* se diferencia de frases que cumplen *otras funciones*, como son las preguntas u órdenes (cuya función es dirigir la conducta del receptor); y también las felicitaciones o sátiras (que expresan la actitud emotiva, positiva o negativa, de su emisor).

Premisas y Conclusiones

Premisa.- Es una afirmación que usamos como una razón o motivo para hacer que se acepte la conclusión. Se puede reconocer mediante 'indicadores de premisa'.

Conclusión.- Es el punto de llegada de un razonamiento: la afirmación que se busca probar. No tiene un lugar fijo en este; puede estar al inicio, en medio o al final del conjunto de premisas. Se puede reconocer mediante 'indicadores lingüísticos de conclusión'.

En resumen, cada razonamiento incluye proposiciones que juegan uno de los dos papeles que lo constituyen (hacen de conclusión o de premisa) y pueden ir o no acompañados de indicadores que señalan cuál de estos dos roles cumplen. En todo caso, la propiedad más importante de un razonamiento es la intención de probar la proposición que hace de conclusión, a cuyo servicio están la demás proposiciones: las premisas.

Reconocimiento de Razonamientos

Hay que diferenciar la *proposición* de otros tipos de frases como son: la orden, solicitud, mandato, explicación (científica, histórica o de la vida cotidiana), datos referentes al tiempo, enumeración, clasificación, relato o una mera *opinión* que se emite sin apoyarla en razón ninguna. Y si tenemos una sola afirmación, tampoco basta para obtener un razonamiento, en vista de que este consta de dos o más proposiciones.

Su estructura básica tiene forma condicional: Si P, entonces Q

Además suele contener indicadores de premisa, de conclusión y, a veces, de ambos.

La construcción de todo razonamiento exige que se cumpla su principio fundamental: el de tratar de probar o demostrar la conclusión, de la opinión que se defiende.

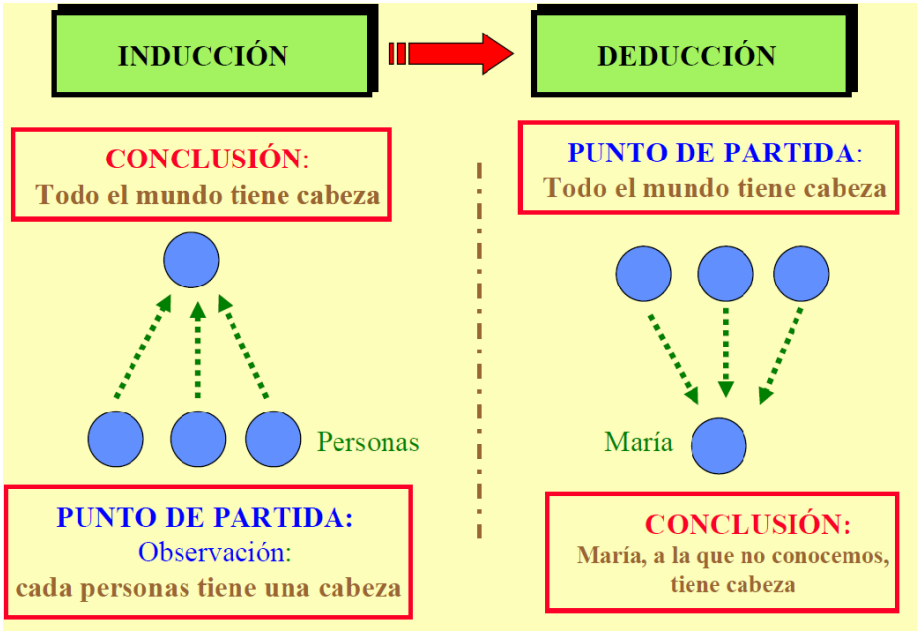
Ej:

"**Licenció el antiguo ejército y creó uno nuevo**; dejó las amistades viejas y se hizo de otras; y así, **rodeado por soldados y amigos adictos, pudo construir sobre tales cimientos cuanto edificio quiso.**" *El Príncipe* de Nícolo Maquiavelo

Dos tipos de Razonamientos.

→ **Deductivos**.- Pretenden que sus premisas sean fundamentos concluyentes, de modo que la *verdad* de la conclusión sea *absolutamente segura*; por eso son *válidos* o *inválidos*.

→ **Inductivos**. Solo ofrecen algún fundamento para concluir la *verdad* meramente *probable* de la conclusión; por eso son *plausibles*: su conclusión tiene una mayor o menor probabilidad de ser verdadera.



Para Saber:

El razonamiento que más utilizan todas las culturas de la humanidad es el **analógico**, en el cual se concluye una afirmación nueva por comparación con alguna situación emparentada o similar a ella que tomamos como punto de partida o premisa segura, por ser conocida para la comunidad. Se basa en procesos mentales como el símil (esto es *como* aquello) o la metáfora.

Argumentos Deductivos

Un argumento es correcto –desde el punto de vista lógico–, si siempre que las premisas son verdaderas su conclusión lo es por razones formales. O, dicho de otro modo, si es imposible por razones formales que las premisas sean verdaderas y la conclusión sea falsa. En este caso se dice que la conclusión es **consecuencia lógica** de las premisas o que las premisas **implican** la conclusión. La **argumentación** que exhibe esta relación de implicación entre premisas y conclusión se denomina deductiva.

Un argumento deductivo tiene la propiedad de trasmitir la verdad; es decir, dadas ciertas condiciones formales, a partir de premisas verdaderas se obtienen necesariamente conclusiones verdaderas.

Veamos un ejemplo de argumento deductivo:

Todos los peces son mamíferos
Moby Dick es un pez
Moby Dick es un mamífero

Este ejemplo nos sirve para ilustrar un argumento deductivo en el que las premisas son llanamente falsas. El hecho de que las premisas sean falsas no impide que el argumento sea deductivo: si uno estuviera dispuesto a aceptar que las premisas fueran verdaderas, entonces estaríamos obligados a aceptar la verdad de la conclusión, puesto que no podríamos pensar en ninguna situación en que las premisas fueran verdaderas sin que automáticamente la conclusión también fuera verdadera. Dicho de otro modo, no se da ningún caso en que las premisas fueran verdaderas y la conclusión falsa.

Otro ejemplo;

Todos los caballos son mamíferos

Todos los caballos son vertebrados

Todos los mamíferos son vertebrados

En este ejemplo, tanto las premisas como la conclusión son efectivamente verdaderas, sin embargo el argumento **no es deductivo.** ¿Por qué? Porque aceptar la verdad de las premisas no nos obliga a aceptar la verdad de la conclusión, ya que es fácil imaginar una situación en la que debido a alguna evolución distinta de los mamíferos, no todos ellos fueran vertebrados. Es decir, aunque la conclusión es de hecho verdadera, bien podría ser falsa. No es necesariamente verdadera.

Indicadores

Premisa

Indicador de una premisa.

Cuando en un argumento nos encontramos con expresiones como: *ya que, puesto que, seguidamente, si seguimos con este planteamiento, teniendo en cuenta, etc.,* nos encontramos ante una premisa.

ALGUNOS INDICADORES:

puesto que	*(merced a que)*	*Considerando que*	*en virtud de que*
ya que	*Partiendo de que*	*toda vez que*	*a partir de que*
pues/ como	*visto que*	*por cuanto,*	*lo cual obedece a que*
por/ porque	*siendo (así) que*	*Ello es imputable a que*	*por el hecho de que*
dado que	*en vista de que*	*arguyendo que*	*y eso gracias a que*
debido a que	*al fin y al cabo*	*A raíz de que*	*después de todo*
lo que se genera en que	*ello viene/nace de que*	*esto se sigue de que*	*en razón de que*
tomando en cuenta que			

Conclusión

Indicador de conclusión.

Si en el argumento nos encontramos expresiones como: *en conclusión, en consecuencia, podemos afirmar, para terminar, así...* estamos ante una conclusión.

ALGUNOS INDICADORES:

por (lo) tanto	*en consecuencia de*	*se infiere que*
así pues	*se desprende que*	*por consiguiente*
luego	*se colige que*	*consecuentemente*
(entonces)	*Consiguientemente*	*se llega a que*
por ende	*se concluye que*	
en suma	*Por tal(es) motivo(s)*	*de aquí/ahí que*
así que	*Naturalmente*	*de[esta] suerte (que)*
por eso	*ello trae consigo que*	*de modo que*
se sigue que	*de este modo*	*esto da lugar a que*
en conclusión	*de esta manera*	*se deduce que*
es obvio que	*por esa(s) razón(es)*	*no queda sino aseverar que*
ello da pie a que	*es claro que*	*(todo) lo cual: se debe a/ implica que*
de (todo) ello/esto resulta que	*(se puede colegir que)*	*(se puede inferir que)*
(todo) ello refluye / se refunde: en que	*(todo) lo cual viene/va a parar en que*	*no hay más remedio que afirmar que*

nos vemos obligados a reconocer que	*sería irracional no defender que*	*por las razones señaladas/indicadas*
lo que redunda en que	*en virtud de lo cual*	*donde (se deriva que)*

Regularmente esta distribuidos de esta manera:

INDICADORES DE PREMISA	**INDICADORES DE CONCLUSION**
Puesto que	*Por lo tanto*
Porque	*Por consiguiente*
En tanto que	*Podemos inferior*
Ya que	*Podemos concluir*
Debido a	*Podemos determinar*
Por la razón de que	*Concluyo*
Por el motivo de	*Determino*
Por esa razón	*He decidido*
Por ese motivo	*Se ha determinado*
Por lo antes expuesto	*Luego*

Por lo antes dicho	*Así*
Por lo antes mencionado	*Se sigue que*
Pues	
Por eso	

Una vez que se ha reconocido un razonamiento, esas palabras y frases nos ayudan a identificar sus premisas y su conclusión".

Pero el lenguaje es una herramienta más complicada de lo que parece. Cualquiera de esas palabras pueden ofrecer pistas falsas; según las circunstancias. Si una persona extrae una pistola de su bolsillo y le dice a su interlocutor: **"le ruego que mire mi mano derecha, y *luego* siga atentamente mis instrucciones..."**, obviamente no está razonando, sino mandando; y es común que muchas peticiones se hagan cortésmente en forma tal que podrían aparentar un razonamiento.

Resumen:

En un razonamiento tenemos una o varias proposiciones y, apoyados en ella (o ellas), inferimos una conclusión. Entiéndase que hay razonamiento si la conclusión se apoya en las premisas; de cualquier otra manera puede haber pensamiento (en sentido general), reflexión, cavilación, especulación, discusión, descripción, explicación o lo que se quiera... pero no hay *razonamiento*

Para crear un razonamiento hay que tratar con "proposiciones"; solo estas pueden servir como premisas o conclusiones en un razonamiento.

Verdad y Validez

(Son propios de Razonamientos / Pertenecen a razonamientos deductivos, pero nunca a proposiciones)

Como las premisas o la conclusión de un razonamiento son proposiciones, su propiedad principal es la de ser verdaderas o falsas. Mas los razonamientos no son ni verdaderos ni falsos, no se los debe catalogar así. De un razonamiento se dice que es correcto o no. Y para hablar con mayor precisión, si se trata de un razonamiento inductivo se puede decir que es más o menos probable. A los razonamientos deductivos se les juzga diciendo que son válidos o inválidos.

En un razonamiento debemos ser capaces de separar unas proposiciones de otras y, entre estas, identificar la conclusión. Solo así podremos determinar si el razonamiento es correcto o incorrecto y, por consiguiente, si su conclusión es o no es legítima; y, en el caso de un razonamiento deductivo, este puede ser o bien válido o inválido sin otra opción.

Cuando un razonamiento deductivo es *válido y* además de esto sus *premisa son verdadera, se dice que es un Razonamiento Sólido.*

Esquema Argumental

- Indicadores de premisa y conclusión.
- El contexto, el sentido.

Cuando en un argumento hay varias premisas, estas suelen ir unidas en *conjunción* por el conectivo 'y' o por alguno de sus equivalentes lingüísticos. Con menor frecuencia van enlazadas con en disyunción, es decir, con el conectivo 'o', o con algún sustituto suyo.

Hay que tener un cuidado con otro modo de conexión entre dos afirmaciones, el que tienen las *proposiciones condicionales*. Estas suelen tener la forma *Si … Entonces*... (La existencia de la primera situación genera necesariamente la existencia de la segunda)

Debemos poder discernir entre un razonamiento y una proposición condicional, lo cual se logra entendiendo el sentido global de la frase: si con la cláusula *entonces* se quiere probar o demostrar algo, en ese caso se trata de un razonamiento.

Los indicadores lingüísticos de premisa o de conclusión ayudan a encontrar un argumento, pero pueden confundir al lector distraído, por ello se debe analizar a fondo cada razonamiento. La mejor forma de entenderlos y discriminarlos es cuando, en un texto, captamos que hay una afirmación cuya verdad es discutible, dudosa, pero que se defiende con fuerza; esa será la conclusión del razonamiento.

Pasajes con varios Argumentos.

En un mismo pasaje puede haber uno o más argumentos, los cuales pueden estar o no entrelazados, creando un tipo de dependencia entre ellos, hasta llegar a la conclusión final o tesis que se defiende.

El receptor debe estar en capacidad de separar, uno por uno, los varios argumentos unitarios, y saber qué papel cumplen en la estructura argumentativa global: un papel principal o de puente. Tras analizar cada uno de los componentes, hay que descubrir sus enlaces, entender la articulación de conjunto y desplegar de manera clara el camino completo que lleva a la conclusión.

Así el número de argumentos en un pasaje está determinado por el número de conclusiones que contiene.

ESQUEMA LÓGICO-SINTÁCTICO DE LAS PROPOSICIONES CATEGÓRICAS.

FORMAS BÁSICAS.- Aristóteles Definió 4 formas o tipos básicos de proposiciones categóricas:

-**Universal Afirmativa** Todo S es P -**Universal Negativa** Ningún S es P

-**Particular Afirmativa** Algún S es P -**Particular Negativa** Algún S no es P

La traducción verbal a forma típica de una proposición categórica debe incluir el verbo ser: *Es* o *Son*, y uno de los tres cuantificadores: *Todos* (as), *Ninguno* (as), *Alguno* (as).

Así entenderemos más claramente lo que ella expresa y podremos realizar de manera fácil y rápida su análisis lógico-sintáctico.

Ej.

Hay niños presentes → Algunos niños son seres que están presentes.

Los perros son carnívoros → Todo perro es carnívoro

Normas para las proposiciones:

1.- Ordenar las proposiciones basándose en la forma:

CUANTIFICADOR- SUJETO-VERBO SER-PREDICADO

Ej.

Son peligrosas todas las víboras → Toda víbora es peligrosa

Cuantificador = Toda Sujeto = víbora Verbo = es Predicado = peligrosa

2.- Todo numero y las expresiones que no significan Todo (a) o Ningún (a) o cualquier cuantificador equivalen a **Algunos.**

Ej.

Es falso que **muchos** (= pocos) perros sean blancos → **Algunos** perros son blancos

Cuantificador = Algunos

Sujeto = perros

Verbo = son

Predicado = blancos

En este ejemplo vemos a un cuantificador que se transforma en "algunos" y que, además, que se respeta la primera norma.

3.- Las palabras: las, los, cada, cualquier, cualquier cosa, cada cosa, cuantos (as) "referidas a objetos o personas" y: cualquiera que, quien quiera que, cada uno, quien aquella, aquel, el que "referidas solo a personas" se traducen por TODOS (AS) o por su negación Ningún, y: una, un, la, el.

Ej.

Un murciélago no es un ave → Ningún murciélago es un ave

La ameba es unicelular → Toda ameba es unicelular

4.- Cuando se está queriendo negar se usan palabras como *nadie*, ninguno (= *ni uno*), que expresan negación, pero en castellano se expresan dobles negaciones *aparentes*:

No llegó **nadie/nunca/jamás/ninguno/nada** = **nadie/nunca/jamás/ninguno/nada** llegó

Que en realidad son una sola negación; y, como se ve en el lado derecho de la igualdad que precede, dándoles la vuelta se ahorra la primera negación "no", y se ve no era más que un mero refuerzo de la otra negación de la frase, que es la principal y que se mantiene:

Aquí **no** queda **nada** → **Nada** queda aquí **o** Aquí **nada** queda

Son reiteraciones idiomáticas con las cuales se enfatiza la única negación que incluyen y que no hay que confundir con una **doble negación auténtica**, como es el caso de: "es un cuento que sea mentira que…" que –como en aritmética– sí anula la negación:

(-) (-) = (+).

Otro ej. de doble negación:

Es una **idiotez** pensar que no sea **falso** que **= Es verdad**

Otro ej. de doble negación aparente:

A la sesión **no** se presentó **nadie** = A la sesión **nadie** se presentó

5.- De no aparecer el Verbo Ser, en cualquiera de sus formas, se lo debe incluir en la proposición. Si esta tuviera indicadores de lugar como la palabra "donde", se añade un parámetro de lugar dos veces; uno en el sujeto y otro en el predicado. Así mismo se hará si hay indicadores de tiempo, pero con parámetros de tiempo.

Ej.

Todo va bien si empieza bien → *Toda* **cosa** que empieza bien *es* **cosa** que va bien

En el ejemplo vemos que se respeta el orden propuesto en la norma 1, además de la introducción de nuevos indicadores lingüísticos –llamados *parámetros*– para poder introducir el verbo SER y que la frase se entienda bien.

Ej.

Algunas personas pelean →

– Alg. personas son *peleadoras* (*atributo* en vez del verbo)

– Alg. personas son *personas* que pelean (igual *parámetro* en el sujeto y el predicado)

– Alg. personas son *gente* que pelea (*parámetros*, pero equivalentes, diferentes en ambos)

Se prohíbe entrar con perros → Todo perro es un *animal* con el que no se puede entrar

Se prohíbe entrar con perros → Ninguna *persona* con perro es *persona* a la que se le permite entrar

Nada es a la vez redondo y cuadrado → Ningún *objeto* redondo es a la vez *objeto* cuadrado

Donde las dan las toman → Todo *lugar* en que las dan es *lugar* en que las toman

Jamás lo escuche → Ninguna *vez* es v*e*z en que lo escuche

No existen gatos verdes → Ningún gato es verde

6.- En las **proposiciones condicionales**, la partícula "SI" va con la **condición suficiente (= CS)**, que también se llama *antecedente* de la proposición categórica. La palabra ENTONCES (en ese caso), *la cual se puede omitir*, va seguida de la **condición necesaria (= CN)**, denominada también *consecuente*. Las proposiciones condicionales se traducen al tipo de

proposición categórica Universal Afirmativa si son afirmativas, o al tipo Universal Negativa si son negativas. Por ej.:

Si uno es terco entonces peleara → Todo terco es un ser humano que peleara

Si se trampea, no se es ético → Ninguna persona que trampea es ética

Las cláusulas equivalentes lingüísticamente al SI condicional sirven de indicadores de condición suficiente, y se les da el mismo tratamiento que al SI.

Un listado de los posibles equivalentes del SI, a los que siempre sigue la CS, son:

Siempre que	*Cuando*	*Siempre y cuando*
Las veces que	*Las ocasiones en que*	*Dada la situación de que*
En el caso de que	*En la eventualidad de que*	*En la circunstancia de que*
Basta que	*En la hipótesis de que*	*En el supuesto de que*
Con solo	*Como (+ subjuntivo)*	*En la medida de que*
De haber	*Si hubiera*	*En no habiendo*
A falta de	*Si no hay*	*Es suficiente que*

Hay otros indicadores que van precedidos de una CS, tales como:

Es suficiente para que	*Basta para que*	*Causa que*	*Hace que*
Motiva que	*Produce que*	*Genera que*	*Ocasiona que*
Significa que			

Algunos Ejemplos:

- Si (= las veces que) vienes, te doy café →

  Toda ocasión en que vengas es ocasión en que te daré café

- El que alguien tosa significa que tiene gripe porcina →

  Toda persona que tose es persona con gripe porcina

- Como la gente siga pegada a la TV acabara con el cerebro almidonado →

  Todo ser humano que siga pegado a la TV es ser humano que acabara con el cerebro almidonado

Ejemplo para resolver:

- Cuando baja se cae → Toda vez que baja es vez que se cae

7.- Expresiones como:

Tan solo	*Solamente*	*Únicamente*	*Exclusivamente*	*Nada más que/los/las/el*
Nadie	*Sino*	*Nada sino*	*Meramente*	

Estas unidas inmediatamente a un **SI** o a uno de sus parientes señalados en el **punto 6**, son **indicadores lingüísticos** de la **C N** que les sigue.

Pero cláusulas tales como:

Es necesario: (indispensable, imprescindible, inevitable, insoslayable, impepinable, vital), para que,		
Es efecto de	*Es resultado de*	*Resulta de que*

Van asociadas a una **C N**.

En general toda situación que sea un requisito obligado para que se dé, produzca, genere o cause otra cosa será una **C N**.

Ej.

A fin de graduarse, ***no queda sino*** **(= *más* que**) defender bien la tesis = Todo graduado es persona que defendió bien la tesis

Es necesario ser carnívoro para ser gato = Todo gato es carnívoro

En las formulas lingüísticas:

Para que (llegue…, triunfe…, se convenza..., se logre…)	
Para (llegar…, triunfar…, convencer…, lograr…)	
No queda mas que	*no hay otra solución (salida) que*
se tiene que	*no hay más remedio que*
hay que	*se debe*

A estas expresiones les sigue una **C N**, mientras que, en ellas, el *"para que"* o el *"para"* van seguidos de una **C S**.

Así como toda **C S** se coloca en el sujeto de una proposición categórica de forma típica la **C N** debe ponerse siempre en el predicado de la misma.

Ej.

-Una explosión resulta de una gran expansión de gases → Toda gran expansión de gases es causante de una gran explosión

8.- La proposición de tipo **Universal Afirmativo y** de tipo **Particular Negativa** son contradictorias entre sí, es decir, cuando se niega una de ellas se obtiene la otra; y lo mismo sucede entre las de tipo **Particular Afirmativa y Universal Negativa**, que también resultan ser mutuamente *contradictorias*. Así:

Ej.

Es falso que *toda* persona sea negra = *algunas* personas *no son* negras

Es mentira que *algunos* tontos *no sean* buenos = *Todo* tonto es bueno

No es verdad que *algún* pillo sea honrado = *Ningún* pillo es honrado

Es mito que *ningún* lobo sea manso = *Algunos* lobos son mansos

9.- Las cláusulas *a menos que*, *a no ser que*, *salvo que* y *excepto que* tienen el sentido de una **disyunción exclusiva** (= negación de una equivalencia). Su traducción a forma típica puede expresarse por la conjunción de dos proposiciones categóricas. Así, la formula *A salvo que B* equivale a decir:

(*si B entonces no A*) *y* (*si no B entonces A*) = *Todo B es NO A* Y *todo NO B es A.*

Ej.1 Me iré de la casa, a menos que termine la bulla → Toda vez que termine la bulla es vez que NO me iré de la casa Y toda vez que NO termine la bulla es vez que me iré de la casa

Ej.2

A no ser que me toque el premio mayor de la lotería *no* podré comprar la casa →

Toda vez que me toque el premio mayor de la lotería es vez que ("NO *no*" se anula por doble negación) podré comprar la casa, pero (= Y) toda vez que NO me toque será vez que *no* podré comprar la casa.

10.- Norma general: el resultado de **la traducción a forma típica** de la afirmación del lenguaje ordinario que se analiza debe **tener el mismo sentido** que la proposición original.

De aquí radica la importancia de entender el contexto de dicha proposición.

ESTRUCTURA LOGICO - SINTACTICA

Extracción de la estructura Lógico Sintáctica de párrafos consistentes en un argumento carente de cuantificación.

Definición.- Un Argumento se compone generalmente de indicadores de **premisa y de conclusión**, luego la idea de *argumento* viene a nuestra mente como la oración que debe ser defendida, discutida, apoyada o todo lo contrario, y en el peor de los casos incomprendida.

Un argumento no siempre es claro y puede tener ambigüedades. La apariencia de un párrafo puede engañarnos haciéndonos pensar que una explicación, petición u orden es un argumento, por ello debemos entender bien lo que leemos para discernirlo.

Un Argumento puede estar constituido por una *inferencia inmediata*: una conclusión basada en una única premisa; aunque lo normal es que contenga dos o más premisas, es decir, que sea una inferencia *mediata*.

Indicadores de Premisa y de Conclusión:

Existen *indicadores de conclusión* y entre los más comunes se cuentan:

Por lo tanto	*Por ende*	*Así*	*luego*
Por consiguiente	*Se sigue que*	*Podemos inferir*	*Podemos concluir*

En conclusión	*En consecuencia*	*Podemos afirmar*	*Para terminar*
Por lo tanto	*Luego*	*Se sigue que*	*En consecuencia*
Se concluye que	*Implica que*	*Se desprende que*	

- Entre las más comunes de estos *indicadores de premisa* figuran:

Puesto que	*Porque*	*Pues*	*Ya que*	*Por la razón de que*
Puede ser	*Seguidamente*	*En tanto que*	*Dado que*	*Tomando en cuenta*
Visto que	*Como*	*Debido a que*	*Puede ser inferido de*	
Si seguimos este planteamiento				

Cuando en un razonamiento no consta ninguno de estos tipos de indicadores, se debe buscar cuál es la tesis principal planteada por el autor que lo ha construido y descubrir la intención de demostrar su tesis, la cual constituye la conclusión del argumento.

Ej.

Creo que no hay más formularios en nuestra sección; hemos rebuscado por todas partes

y no hemos encontrado ninguno

Yo creo, seguro que, sin ninguna duda, nadie dudaría respecto de que… son cláusulas de convencimiento (convicción) personal que acompañan a la conclusión y, por consiguiente, son buenos indicios de qué frase es la tesis que defiende el autor del razonamiento: sirven también como indicadores de conclusión. Además existen datos que el interlocutor conoce y que son compartidos entre él y el argumentador, que para ambos aparecen como verdaderos.

En casi todos los casos, explorar el contexto, el contenido y el tono o estilo del argumento hará que encontremos la conclusión. Para lograrlo nos preguntaremos: ¿Qué se quiere probar aquí? ¿De qué nos quiere persuadir el autor?, etc.

No olvidemos que la conclusión es la afirmación opinable cuya verdad no es obvia, que el autor defiende y por lo tanto quiere que creamos, y que recibe el apoyo de otras afirmaciones adyacentes.

Ambigüedades.

Existen varios **tipos de expresiones lingüísticas**:

- Preguntas, órdenes, consejos, recomendaciones, ruegos, solicitudes… Todas ellas cumplen casi siempre una *función directiva* (con ellas, el emisor del discurso trata de dirigir, intervenir o cambiar la conducta del receptor que escucha o lee: hacer que éste haga o deje de hacer algo).

De todas ellas decimos que son pertinentes o impertinentes, racionales o irracionales, adecuadas o inadecuadas con respecto a la situación en la que se lanzan.

- Otras oraciones constituyen una queja, crítica mordaz, ironía, burla, insulto, o una alabanza, felicitación… Todas ellas son expresión de afectividad (jubilo, amor o cariño, tristeza, ira, odio…) positiva o negativa (a favor o en contra de algo); por eso decimos que poseen una *función emotiva* o *expresiva*, pues expresan emociones o sentimientos de parte de quien las emite.

- Hay muchas más clases de oraciones, pero al arte de razonar le interesan solo aquellas que cumplen una *función informativa* (pretenden afirmar o negar algo y por eso son verdaderas o falsas). A estas las llamamos **Proposiciones.**

Funciones del lenguaje (resumen)

- Informativa o Referencial → (usualmente Discurso de forma declarativa). Se da cuando queremos trasmitir información; por ejemplo, opiniones, enumeraciones, relatos, informes, explicaciones o **razonamientos**. *Sus frases son verdaderas o falsas*: son **proposiciones**.

- Expresiva o Emotiva → (usualmente Discurso de forma exclamativa). Es típico de la poesía, del intercambio de emociones en la vida cotidiana; por ej., sátiras, insultos, alabanzas. De tales expresiones no se puede decir que sean verdaderas o falsas, sino *sinceras* o insinceras, *auténticas* o inauténticas.

Directiva → (usualmente Discurso interrogativo o imperativo); por ej., preguntas, órdenes o pedidos, que no pueden ser ni verdaderos ni falsos, pero sí *pertinentes* o no, *adecuados* o inadecuados a la situación en la cual se emiten.

En general, casi todas las frases, según su contexto informativo, tienen propiedades de verdad o falsedad, cualidades de ser razonables o adecuadas, son sinceras o no; vale decir, cumplen una función mixta o una mezcla de funciones. En el ámbito del razonamiento nos interesa ante todo la función informativa.

Las meras **opiniones** referenciales pueden confundirse con los argumentos lógicos y podemos distinguir *dos tipos* de opiniones:

- *Simples*: solo se las emite sin ningún tipo de aclaración o apoyo

- *ligeramente elaboradas*: se trata de un párrafo formado por varias oraciones, cada una de las cuales amplía la idea de la anterior, solo reafirman, aclaran la idea, y no llegan a demostrar nada. Pueden ser, o no, parte de un razonamiento.

- **Explicaciones**; suele tener una estructura y usar indicadores que se asemejan mucho a un argumento (**P, porque Q**), pero la conclusión (P) marca la diferencia entre este tipo de expresiones y los **razonamientos**. En estos últimos la afirmación P es un punto de vista discutible cuya verdad es dudosa (unos la aceptan como verdad, mientras otros la rechazan), por lo cual su autor la defiende como *conclusión* de un argumento y trata de convencer a los demás de que sí se trata de una verdad, y para ello la apuntala o apoya con motivos o

razones, que son las premisas. Por otro lado, la misma estructura *P*, *porque Q* es un **discurso explicativo** cuando la proposición P es un HECHO –histórico, científico o de la vida cotidiana– y lo que hace su autor es dar los *motivos o razones* de ese hecho: las causas que lo han producido o generado, estos constituyen la explicación.

Lo primero es saber distinguir un razonamiento de otras expresiones informativas. En el lenguaje cotidiano los argumentos suelen ir envueltos en innecesaria palabrería →

Las *repeticiones* o *redundancias*: la "lata".

Las partes de un texto que son irrelevantes nos hacen perder el tiempo y gastar energía mental inútilmente. Lo que importa es expresar *claramente* lo que uno quiere decir en base a la consigna: el *máximo de contenido* en el *mínimo de palabras*.

Algunas veces, en un argumento se repite la conclusión sin motivo, en otras ocasiones el lector debe pasar por alto –ignorar – una introducción demasiado larga a un argumento.

Ej_1

"Según una investigación reciente, el humo del tabaco pronto se difumina en el aire; esto lo dice el director de salud; luego el humo del tabaco no produce un daño serio en los no fumadores "

Traducción a *lenguaje neutro*: Según el director de salud, el humo del tabaco no produce daño serio en personas que no fuman, *ya que* se difumina pronto en el aire.

Ej_2

"Sabiendo la lechuza de quien es el día, susurra impaciente al nacer el alba, quisiera arrancarle el sentimiento al alma, de la luz del sol que se le priva."

Este párrafo tiene carácter poético y habla sobre la nostalgia de la noche, representado por un animal nocturno. Para entenderlo mejor nos vemos en la necesidad de eliminar el exceso de palabras y expresar su contenido en la forma más sucinta pero clara y sin perder información: pura información carente de afectividad.

Traducción a lenguaje neutro

"Algunos animales nocturnos como la lechuza no pueden ver durante el día"

Proposición Atómica (o singular) es una frase que cumple función informativa cuyo sujeto es un individuo concreto y bien determinado.

Ej. Luis es listo

El Chimborazo es un nevado

Bucéfalo fue un caballo famoso

Dos o más proposiciones atómicas unidas entre sí por uno de los conectivos lógico-sintácticos forman una **Proposición Molecular**, -cuya complejidad es mayor.

Ej.	Luis es apuesto Y María está feliz	(CONJUNCIÓN)
	Luis es apuesto O María es feliz	(DISYUNCIÓN)
	SI Luis es apuesto ENTONCES María es feliz	(IMPLICACIÓN)
	María es feliz SI Y SOLO SI Luis es apuesto	(EQUIVALENCIA)
	Luis NO es apuesto	(NEGACIÓN)

Conectivos lógico – sintáctico Y, O, Si Entonces.

Estos conectivos modifican el sentido de una frase, dada la riqueza de nuestro idioma.

- La **Conjunción**, cuyo conectivo es Y, se supone que une dos verdades. Por ej. Quito es la capital del Ecuador **Y** Lima la capital de Perú

- **Disyunción inclusiva**, cuyo conectivo es O expresa afirmaciones alternativas, de las cuales puede darse la una o la otra, o ambas a la vez (por incluir esta última posibilidad se llama O inclusivo); por ej.: *para postre del buffet puedes tomar helado **O** fruta*.

- **Disyunción exclusiva** También hay un segundo significado del conectivo O, en el que las dos alternativas que se ofrecen son incompatibles entre sí, es decir, se excluyen mutuamente, y por eso se llama Disyunción exclusiva –"O exclusivo". Ej.: (en una familia pobre en que hay dos hermanos y solo dos piezas de fruta para postre, la madre dice a los hijos:) *Escoge la pera **O** la manzana* (no ambas a la vez, para que cada quien coma solo una fruta)

- **Proposición Condicional**.- Su conectivo es SI … ENTONCES –también "IMPLICA"–. Ella convierte a la primera frase en condición suficiente de la otra, o a la segunda en la condición necesaria de la primera. Ej.: si me cae el gordo de la lotería, entonces me daré la vuelta al Mundo.

- **Equivalencia**.- Su conectivo es EQUIVALE A, SI Y SOLO SI, ES CONDICIÓN SUFICIENTE Y NECESARIA DE"… y significa que las dos frases que se unen tienen el mismo valor de verdad: o ambas son falsas o ambas son verdaderas. Por ej., Cervantes fue autor del Quijote (en este caso el verbo ser "fue" significa "equivale a")

La Implicación es el conectivo lógico-sintáctico más rico, pero también el más difícil de manejar, pues a diferencia de los otros no es volteable. Así, mientras que la frase "si un ser vivo es persona, entonces ese ser vivo es animal" es verdadera, al darle la vuelta: "si un ser vivo es animal, entonces ese ser vivo es persona" se convierte en una falsedad flagrante. Sin embargo, las frases que tienen los esquemas **A y B**, **A o B**, **A equivale a B**, sin importar de qué se esté hablando se pueden expresar como **B y A**, **B o A** y **B equivale a A**; obviamente entendidas como pura información, sin introducir criterios afectivos o actitudinales que

exigen algún tipo de preferencia entre A y B. Por ej. la frase "El papa y el párroco del lugar entraron a la reunión" está, desde un punto de vista jerárquico (enjuiciamiento que depende de una valoración dentro de la cultura de Occidente) es más correcta (educada, cortés) que ""el párroco del lugar y el papa entraron a la reunión", aunque ambas frases contienen la misma información para alguien que no está dentro de esa cultura.

La **Negación** afecta tanto a proposiciones singulares o atómicas como a proposiciones moleculares y el efecto que produce en ellas es cambiar su valor de verdad.

Cada uno de los cinco conectivos lógico-sintácticos básicos se pueden expresar de muchas maneras un idioma determinado. Lo importante es identificar en cada frase a cuál de ellos corresponde una expresión que une a dos proposiciones. Veamos listas de **indicadores lingüísticos** frecuentes para los diversos conectivos.

Indicadores Lingüísticos de los conectivos lógico-sintácticos:

NO:

Nunca	*es una fábula*	*escurrir el bulto a*	*ni en broma*
Jamás	*es pura ficción*	*ni mucho menos*	*ni de chiste*
es mentira	*esquivar (+ inf)*	*de ningún modo*	*rehusar*

es un mito	*inhibirse de*	*en absoluto*	*negarse a*
evitar	*es contra(rio a) la verdad que*	*ni por asomo*	*dejar de* (+ inf)
regir	*de ninguna manera*	*nada de que*	nanay/nada: que
es un cuento	*no se da/es el caso (de) que*	*por nada lo*	*para nada*
es una quimera	*no es sino/más que falaz/ilusorio*		

O inclusivo:

(*o*) *bien...*	(*o*) *bien...*	*ya..., ya...*
ora.., ora...	*así…, así…*	*sea..., sea…*
(*fuera…, fuera…*en pasado)	*unas/a veces…, otras (veces)…*	*tanto si… como si…*
o también		

Expresan una alternativa en el antecedente de una proposición condicional, pueden darse una u otra o ambas.

O exclusivo:

A menos que	A no ser que	Salvo que	Excepto que	Una de las dos	o al menos

De una alternativa se da solo una de ellas excluyendo la otra o viceversa

Equivale A:

es idéntico/igual a	*significa / es lo mismo que / o sea*	*{Es} exactamente [en el caso de que]*
si y/pero: solo si	*es condic. suficiente y necesaria de*	*ni más ni menos en el caso de que*
Y recíprocamente	*L@s X son l@s únic@s que… A*	*solo los X sin excluir a ninguno*
cuando… es cuando	*justamente en el caso de que*	*No es más que (= coincide con ser)*
Es idéntico	*Es igual*	*Es exactamente*
Es lo mismo	*Es mas que*	*Justamente*
O sea	*Significa*	*Exactamente*
En el caso de que		

Que significan Y que aunque vayan dos juntos (Ej. pero sin embargo) no constituyen sino una sola conjunción:

También	*de modo semejante*	*así mismo*	*cuando no*
(a) más/aparte (de)	*de manera similar*	*a semejanza de*	*en tanto que*
además (de)	*en igual forma*	*de modo semejante*	*mientras que*
(así) como	*al) igual que*	*de manera similar*	*al (mismo) tiempo que*
tal como	*a la par que*	*en igual forma*	*tanto...) como...*
al modo de	*equivalentemente*	*otro(a) que tal*	*no solo... sino también...*
otro tanto	*Igualmente*	*asimismo/ lo mismo*	*por un lado... por otro...*
Ídem	*eso mismo*	*así mismo*	*por una parte.. por otra...*
otro(a) que tal	*e incluso*	*a semejanza de*	*a su vez/ a la vez que*
asimismo/ lo mismo			

- Conjunciones Adversativas:

Mas (= pero)	*Aunque*	*Aun cuando*	*Si bien*	*Aun (gerundio)*
Si bien	*Bien que*	*Sin embargo*	*No obstante*	*Con todo*

Así y todo	*Frente a ello*	*En compensación*	*Mas que*	*Contra lo cual*
En cambio	*Sino*	*Como contra peso*	*Pese a que*	*A pesar de que*
Antes bien	*Pero si*	*Para equilibrar eso*	*Con (+ infinitivo)*	

Precedidos de negaciones:

sino (más bien)
antes bien
pero sí

Y No: ni, tampoco, ninguno de los dos, sin que, sin (+ infinitivo), […, *tampoco* hay/viene…]

- Asociados a un SI condicional, es decir las Condiciones Suficientes y Necesaria:

Si CONDICIONAL

En las proposiciones condicionales con la forma *SI **P** Entonces **Q***, la expresión SI va con la **Condición Suficiente (CS)**, que se coloca como antecedente (***P***); mientras que la cláusula **entonces**, que se puede omitir, va seguida de la **Condición Necesaria (CN)** y se coloca como consecuente (***Q***). Ya vimos que este orden no se puede variar, cuando más arriba

indicamos que los demás conectivos lógicos conservan su sentido al voltearlos, pero que una proposición condicional cambia radicalmente su sentido si se le da la vuelta. Por ej.:

SI P ENTONCES Q Si te pagan, bailo -- Si bailo, te pagan

Una misma expresión condicional se puede expresar de muchas maneras

Ej.

Todas las expresiones siguientes equivalen a: SI A, ENTONCES B

B si A

en caso de que A, B

B en caso de que A

A solo si B

solo si B, A

solo A si B.

Todas estas formulaciones comparten **la misma estructura sintáctica profunda**; aunque, para traducir cada una de ellas a esa estructura, que podemos llamar también *forma normal* (**Si CS, entonces CN**) hay que identificar con precisión cuál es la condición suficiente y cuál la necesaria.

Hay que poner atención a los usos del SI que equivalen a una conjunción adversativa y no a una proposición condicional "Si…Entonces". Por ej.: "Si (bien) María era un poco loca todo el mundo sabía que su hermana era tremendamente sensata". Por el sentido, sabemos que esta frase no es una proposición condicional sino una conjunción: "María era un poco loca Y su hermana era tremendamente sensata"

Indicadores lingüísticos equivalentes al "Si" condicional.

- Seguidos de una Condición Suficiente:

siempre que	al (+ infinitivo)	al rato en que	al momento en que
la primera vez	al instante en que	al punto en que	cuando
siempre y cuando	cada y cuando	las veces que	una vez que
con tal que	a condición de que	en la medida en que	dado que
en el caso de que	en el evento de que	en el supuesto de que	es suficiente que
con solo	no hace falta	nada mas que	de haber
en la hipótesis de que	dada la situación de que	en la circunstancia de que	en la eventualidad de que

si hubiera hasta que (prece. de ~)	todo lo que se requiere es	luego de haber tan pronto como (+subjuntivo)

- Unido a una negación:

en no habiendo	si no hay	de no hacerlo	es suficiente para que
tiene por efecto	ocasiona que	genera que	produce que
reside en	consiste en	se fundamente en	se basa en
hace que	significa que	basta para que	genera que
a falta de	tiene como resultado		

Toda situación que sea requisito o condición ***sine qua non*** (= sin la cual algo es imposible) para que se dé, produzca, genere o cause otra cosa será una **CN**. El **indicador modelo** de **CN** es **SOLO SI** (= *solo que, no otr@/s [cosa/s] que*). **Para formar equivalentes lingüísticos de *solo si*, a un indicador del *SI condicional* de los párrafos precedentes, hay que añadir, por delante un equivalente de SOLO:** *(tan) solo, únicamente, exclusiva-te, mera-te, simple-te, nada más (que), sin más, a secas, nadie que no sea, nadie/ nada/ nunca: sino/salvo*. Pero **Cuidado**, pues pueden presentarse expresiones engañosas. Por ej., dado que **la CS equivale a la**

SUMA DE TODAS LAS CN para que se genere un efecto, expresiones tales como: *todo lo que se necesita es*…, *no es necesario sino*… o *no hace falta más que*… van seguidas de una CS.

Toda situación que sea un requisito o condición *sine qua non* (= sin la cual algo es imposible), para que se de, produzca, genere o cause otra cosa será una condición necesaria (CN).

El indicador modelo (principal) de **CN** es ***SOLO SI***, que equivale a decir: SOLO QUE, NO OTRAS COSAS QUE, etc.

Para formar **equivalentes lingüísticos de SOLO SI**, a un indicador del SI condicional de los párrafos precedentes hay que añadir, por delante un equivalente de SOLO:

tan solo, únicamente, exclusivamente, meramente, simplemente, sin mas,

nada más que, a secas, nadie que no sea, nadie/nunca/nada/sino/salvo

Hay que tener cuidado con las expresiones engañosas. Por ejemplo, dado que una CS equivale a la suma de todas las condiciones necesarias para que se genere el efecto que ella produce, expresiones tales como: *todo lo que se necesita es*…, *no es necesario sino*…, *no hace falta más que*... van seguidas de un CS.

Son indicadores de CN (casi todos precedidos de un ***es***):

es:

necesario	*obligativo*	*obligatorio*	*inevitable*
indefectible	*el único medio (que)*	*vital*	*Se requiere*
forzoso	*impepinable*	*de obligación*	*inaplazable*
la única vía/camino //	*exigible*	*ineludible*	*por fuerza mayor*
preciso	*urgente*	*de rigor*	*inexcusable*
irremediable	*imperativo*	*impajaritable*	*insoslayable*
imperioso	*por obligación*	*indispensable*	*inapelable*
menester	*apremiante*	*quiera que no*	*esencial*
perentorio	*imprescindible*		

Indicadores cuyo sujeto es (= que van precedidos de) una CN:

es efecto de	*resulta de (que)*	*es generado por*	*es engendrado por*
es implicado por	*surge de que*	*deriva de que*	*es suscitado por*

es fruto de	*es resultado de*	*es originado en que*	*es producto de*
es causado por	*es producido por*	*es ocasionado por*	

Indicadores seguidos de una CN:

hay que se tiene que	*no puede sino*	*no queda más que*
no puede	*por menos que*	*no hay más remedio que*
se debe	*hace falta*	*no le toca sino*
no hay otra solución/ salida que		

Ejemplos: (con su expresión ya ordenada)

- Los estafadores **son** los **únicos** ladrones = *Solo si es* estafador *es* ladrón = *Todo* ladrón *es* estafador

- Los estafadores **no son los únicos** ladrones = *Algunos* ladrones *no son* estafadores

- Tu felicidad **resultara** de tu trabajo asiduo = *Si* trabajas asiduamente, *entonces* lograras tu felicidad

Estructura SI CS, entonces CN

Si el SOLO, o un equivalente lingüístico suyo va al inicio de la frase separado de una expresión equivalente al SI, dicho SOLO debe reubicarse delante del SI; y, en consecuencia, el "solo si" se forma delante de la segunda mitad de la frase, de modo que dicha mitad constituye la CN. La estructura resultante siempre será **si CS, entonces CN**

A la frase "**Solo** compro casa **si** me toca el premio gordo de la lotería" se la podría entender de dos maneras:

1- **Solo si** compro casa me toca el premio gordo de la lotería

2- **Solo si** me toca el premio gordo de la lotería compro la casa

Enseguida nos damos cuenta, por el sentido, que solo la segunda alternativa es la afirmación que tiene el mismo sentido que la frase original; de modo que la CN es "*me toca el premio gordo de la lotería*" y la traducción a forma normal de proposición condicional será: "Si me compro casa, entonces me tocó el premio gordo de la lotería".

Como se ve, es muy importante entender lo que se escribe y lo que se lee.

Esquemas Frecuentes que Expresan CS o CN:

Para enfrentar/hacer frente/ o dar o hallar respuesta/solución/salida a (o: en/frente a/ante/en vista de) la/el (aquí puede ir algún adjetivo que matice fáctica y/o emotivamente la afirmación):

1.- circunstancia, eventualidad, particularidad, situación, coyuntura, contingencia, posibilidad + DE QUE…

2.- hecho, evento, suceso, acontecimiento, caso, acaso + DE QUE…

- Aquí viene el hecho o circunstancia que sea, que puede ser un bloque

- Y, tras un indicador de CS o CN sigue la afirmación que adquiere este papel

Ej.

Ante la dura coyuntura de una

de iniciarse una ola de violencia masiva / …

de suceder que se iniciara una:

A: Habría que

se tendría que

se necesitaría dictar leyes restrictivas de las libertades (CN)

se deberían:

B: Seria suficiente

bastaría con dictar leyes restrictivas de las libertades (CS)

La Traducción a forma normal de proposición condicional es…

A: Si se iniciara una ola de violencia masiva, entonces se dictarían leyes restrictivas de las libertades.

B: Si se dictaran leyes restrictivas de las libertades, entonces ya se inicio una ola de violencia masiva.

Indicadores de Bloque.

Los Indicadores de Bloque sirven para iniciar un conjunto de afirmaciones. Un bloque se hace imprescindible si se trata de atribuir la misma negación a dos o más proposiciones.

No acaece el hecho de que: al levantarse, Teófilo se mire al espejo y maldiga su suerte mientras que su esposa ronca plácidamente.

Indicadores de Bloque:

sucede	acontece	se presenta
se efectúa	se hace realidad	ocurre
(sobre-)viene se produce	se verifica	se realiza
se convierte en un hecho	se ofrece	Acaece
la circunstancia	peculiaridad	Coyuntura
el evento / caso	eventualidad	Singularidad
contingencia	acaso	Particularidad
posibilidad acontecimiento hecho	situación	Suceso
se da		

Otro indicador usual de Bloque es PARA, y sus expresiones parientes que significan *finalidad* (a fin de que, con el objeto de que, con la finalidad de que, con el objetivo de que, para lograr el propósito de que, a fin de alcanzar la meta de que…) suelen ir en una proposición condicional, y la frase que les sigue puede ser CS o CN, según el indicador que conste en la otra mitad de la frase, que puede ir antes o después de la cláusula acompañada del PARA (o de uno de sus equivalentes).

Ej.

Para que triunfe debe conseguir más estudios.

Indicador de CN = *debe*; en este caso, la cláusula que va con el *para* será la CS

Basta esforzarse con toda el alma a fin de conseguir los sueños

Indicador de CS = *basta*; aquí, la cláusula que va con la expresión *a fin de* será la CN

MÉTODO Y EJEMPLO DE EXTRACCIÓN DE LA ESTRUCTURA LÓGICO-SINTÁCTICA

Pasos para reconocer y extraer la estructura lógico-sintáctica de un argumento:

1.- Reconocer su conclusión

2.- Según van apareciendo sus proposiciones, se las enlista y a cada una le asigna una letra

3.- Volver a escribir el razonamiento pero esta vez sustituyendo por sus respectivas letras

4.- Por fin, obtener la estructura lógico-sintáctica de cada una de las premisas y de la conclusión tras interpretar todos y cada uno de los indicadores de los conectivos lógico-sintácticos.

Al realizar esta ultima tarea, allí donde constan los conectivos Y, O, NO o EQUIVALE A hay que escribir la estructura sin variar la colocación de las frases del texto original. Pero hay que poner atención en el caso de las proposiciones condicionales, pues a ellas no se las puede dar la vuelta (no puede sustituirse la CS por la CN, o al revés), hay que identificar una de las dos, la CS o la CN, de modo que la proposición se traduzca a su forma normal (o estructura profunda) en el único orden correcto: **Si CS, entonces CN**. Esto quiere decir que si encontramos una frase en la que la CN esta primero, debemos voltear dicha frase.

Ej_1.

La guerra se forma por ambiciones, los países, directa o indirectamente, se ven involucrados en ella, ya por vínculos económicos o por alianzas estratégicas, mas siempre traerá devastación para todos, pobreza a unos cuantos y riqueza a pocos,

Ej_2.

Para que Grecia juegue no hay otra vía posible fuera de que España lo haga; dado que, el hecho de que Grecia no participe en el juego se produce al ser falso que, a pesar que Dinamarca no juegue Francia lo haga. A más de eso el que Francia no juegue al no hacerlo España, depende de que Dinamarca no tome parte en el juego.

1.- Conclusión: Para que Grecia juegue no hay otra vía posible fuera de que España lo haga.

2.- a = Grecia juega b = España juega c = Dinamarca juega d = Francia

3.- 3.1- dado que el hecho de que no A se produce al ser falso que, a pesar de que no C, D

3.2- el que no **D** al no **B**, depende de que no **C**

Concl- Para que **A** no hay otra vía posible fuera de que **B**

4.- Concl- B es condición necesaria y A es condición suficiente.

Para que es indicador de bloque y afecta solo a A.

No hay otra vía posible fuera de que afecta a B y señala condición necesaria.

Si A entonces B

4.1- *El hecho de que*, es indicador de bloque y afecta a la expresión no A, además de que *se produce al* indica que lo que viene es generado por lo anterior, es CN, luego queda señalado por *al* que es indicador de CS. Quedando:

si no (no C y D), entonces no A

4.2- *El que* es abreviatura del indicador de bloque EL HECHO DE QUE y afecta a l proposición molecular compleja “no D al no B”, y *depende de* “no C”; esto indica que “no C” es la CS y de que no D al no B será la CN. Quedando:

Si no C, entonces (Si no B, entonces no D)

si no (no C y D), entonces no A. si no C, entonces (si no B, entonces no D)

si A entonces B

CAPÍTULO IV

Pruebas y Ejecución del

Programa Multimedia Interactivo

4. DESARROLLO DE LAS PRUEBAS Y TERMINACION DEL PROGRAMA MULTIMEDIA INTERACTIVO

4.1 DESARROLLO DE PRUEBAS AL PROGRAMA

La siguiente lista de pruebas que se detallan a continuación nos han servido para medir la efectividad de nuestro programa en base a los requerimientos de HW, SW, y lo que es mas importante la satisfacción del Usuario final que en si será el beneficiado y quien deberá sentirse a gusto con la herramienta.

Se a tratado de verificar posibles fallos de diseño, código, de secuencia en los temas tratados en resumen de calidad. Por otro lado también se corrieron pruebas funcionales ya que debemos verificar que se ejecute correctamente en diferentes entornos, sean estos *Sistemas Operativos* y *Hardware*, se valoraran los mas comunes en el mercado actual de computadores, esto nos asegurara que el común denominador de equipos existentes en la universidad y los mas comunes en casa n tengan problemas al usar el programa.

Las siguientes pruebas tienen como finalidad la comprobación del software en la ejecución bajo diferentes plataformas…

Prueba funcional

La prueba funcional esta basada en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. La finalidad es comprobar que funcione todo el paquete bajo diferentes plataformas. Se han realizado pruebas sobre la marcha del desarrollo, mas para cerciorar la efectividad del programa se verificara su ejecución sobre el programa terminado y sobre varias plataformas (S.O.)

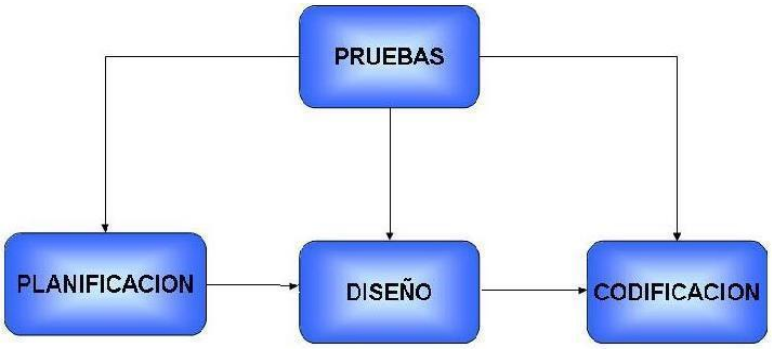


Figura 9

Se definen a continuación conceptos los cuales permiten comprobar el grado de cumplimiento versus las especificaciones determinadas para el producto, y son:

- DEFECTO.- para el caso en estudio, software, es un procesamiento erróneo en un programa, lo que es de los datos.

- FALLO.- se refiere a problemas en el rendimiento de un sistema en sus funciones.

- ERROR.- Son valores erróneos, es la diferencia entre un valor calculado, observado o medido y el valor verdadero, especificado o teóricamente correcto.

A continuación se presentan las pruebas realizadas:

4.1.1 EQUIPOS QUE LOS ESTUDIANTES DE LA PUCE TIENEN A SU DISPONIBILIDAD.

Los equipos de los que dispone la Universidad Católica del Ecuador varían según su uso, así para algunas carreras como Arquitectura o Ingeniería, las demás Facultades manejan el estándar, y para el caso en desarrollo se aplicaran algunas pruebas de funcionamiento ya que, el programa, además de usarse en las aulas, al poseer portabilidad, se lo llevará a sus casas.

4.1.2 PRUEBAS (Todas las pruebas fueron realizadas en SO de 32 bit)

PRUEBA # 1:

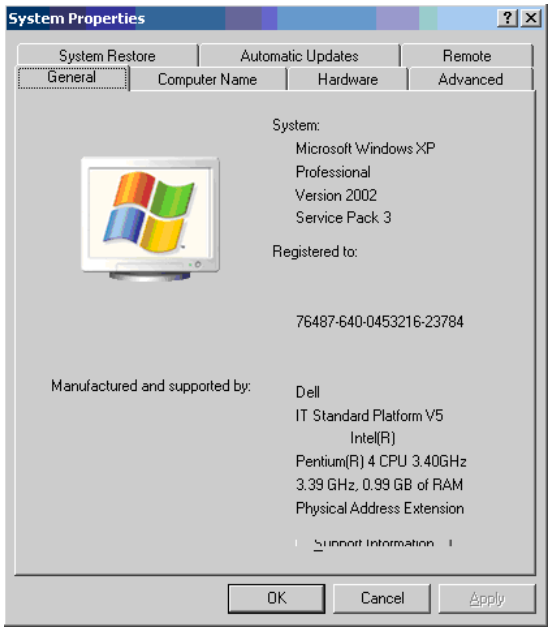


Figura 10

Para este tipo de equipo se corrieron pruebas de funcionalidad y de correcta ejecución:

HARDWARE	SOFTWARE
Equipo: HP	Sistema Operativo: XP Service Pack 2 version 2002
Procesador: Pentium 4, 3.4 Ghz	Versión Adobe Flash Player: 10.0 activex
Memoria RAM: 0.99GB	Versión Adobe Reader: 8.1.1
Disco Duro: 90 GB	Versión Adobe Flash Player: 10.0 plugin
Unidad de CD/DVD - RAM	
Puertos USB	

Tabla 3

RESULTADOS:

ELEMENTO A PRUEBA	FUNCIONA	OBSERVACIONES

Botones		Los botones realizan su respectiva acción.
Diseño		El diseño de la animación mantiene tamaño y resolución.
Código Action Script		Código cumple con sus funciones
Movie Clip		Funcionamiento correcto
Escenas		Correcto ligamiento entre escenas
Frame		Sin errores

Tabla 4

PRUEBA # 2:

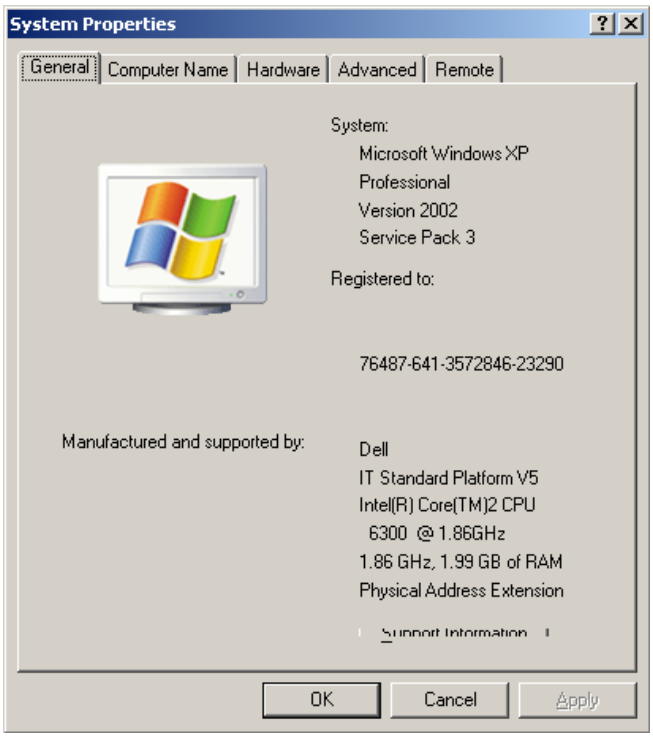


Figura 11

Para este tipo de equipo se corrieron pruebas de funcionalidad y de correcta ejecución:

HARDWARE	SOFTWARE
Equipo: HP	Sistema Operativo: XP Service Pack 3 version 2002
Procesador: Intel Core 2, 1.86 Ghz	Versión Adobe Flash Player: 10.0 activex

Memoria RAM: 1.99GB	Versión Adobe Reader: 8.1.1
Disco Duro: 160 GB	Versión Adobe Flash Player: 10.0 plugin
Unidad de CD/DVD - RAM	
Puertos USB	

Tabla 5

RESULTADOS:

ELEMENTO A PRUEBA	FUNCIONA	OBSERVACIONES
Botones	✓	Los botones realizan su respectiva acción.
Diseño	✓	El diseño de la animación mantiene tamaño y resolución.
Código Action Script	✓	Código cumple con sus funciones

Movie Clip	✓	Funcionamiento correcto
Escenas	✓	Correcto ligamiento entre escenas
Frame	✓	Sin errores

Tabla 6

PRUEBA # 3:

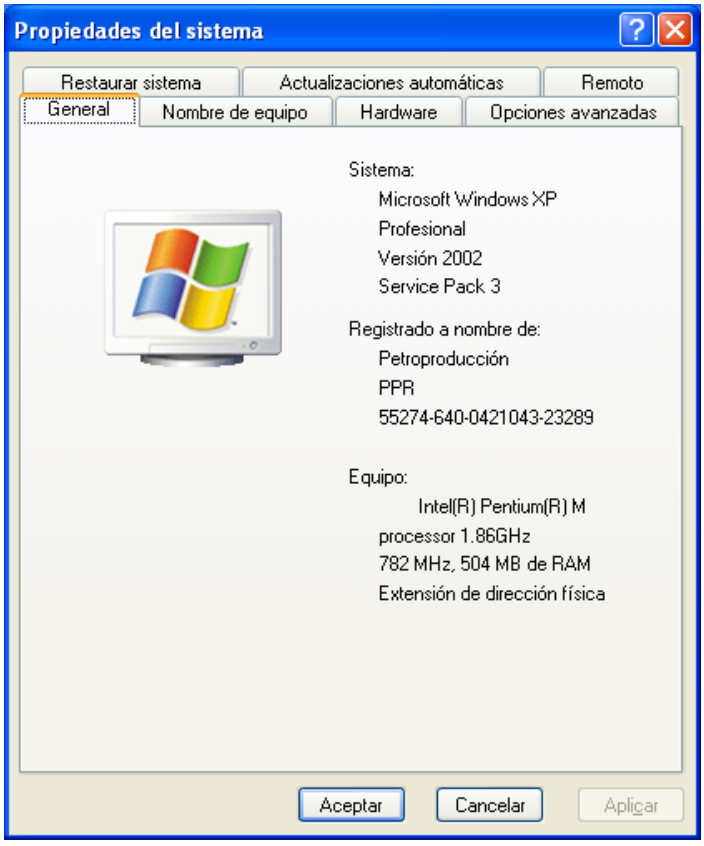


Figura 12

Para este tipo de equipo se corrieron pruebas de funcionalidad y de correcta ejecución:

HARDWARE	SOFTWARE
Equipo: HP	Sistema Operativo: XP Service Pack 2 version 2002
Procesador: Pentium 4, 1.86 Ghz	Versión Adobe Flash Player: 10.0 activex
Memoria RAM: 504 RAM	Versión Adobe Reader: 8.1.1
Disco Duro: 60 GB	Versión Adobe Flash Player: 10.0 plugin
Unidad de CD/DVD - RAM	
Puertos USB	

Tabla 7

RESULTADOS:

ELEMENTO A	FUNCIONA	OBSERVACIONES

PRUEBA		
Botones		Los botones realizan su respectiva acción.
Diseño		El diseño de la animación mantiene tamaño y resolución.
Código Action Script		Código cumple con sus funciones
Movie Clip		Funcionamiento correcto
Escenas		Correcto ligamiento entre escenas
Frame		Sin errores

Tabla 8

PRUEBA # 4:

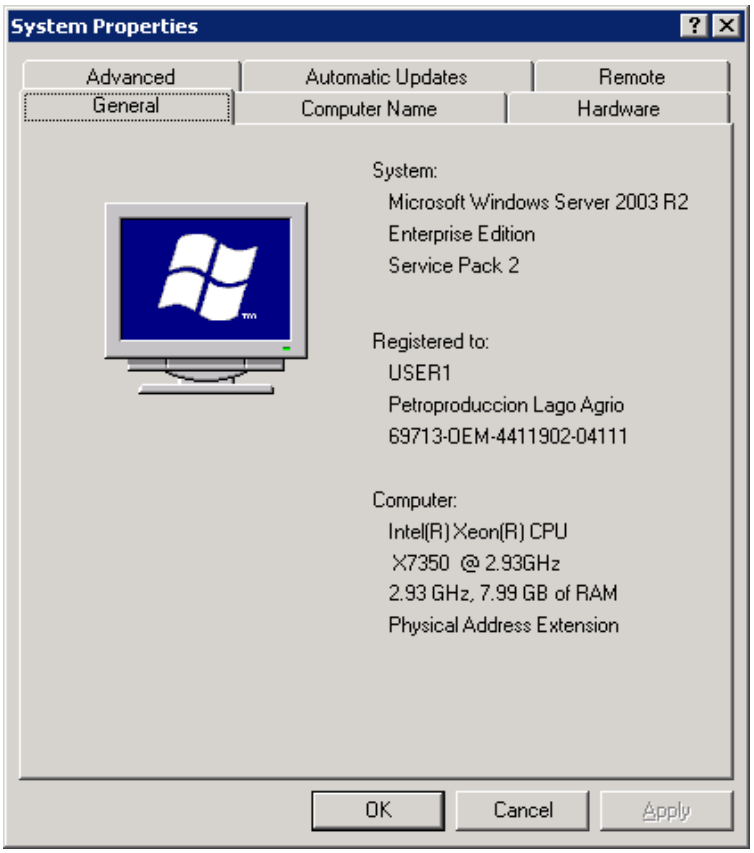


Figura 13

Para este tipo de equipo se corrieron pruebas de funcionalidad y de correcta ejecución:

HARDWARE	SOFTWARE
Equipo: HP	Sistema Operativo: Server 2003 R2
Procesador: Intel Xeon, 2.93 Ghz	Versión Adobe Flash Player: 10.0 activex
Memoria RAM: 7.99GB	Versión Adobe Reader: 8.1.1

Disco Duro: 300 GB	Versión Adobe Flash Player: 10.0 plugin
Unidad de CD/DVD - RAM	
Puertos USB	

Tabla 9

RESULTADOS:

ELEMENTO A PRUEBA	FUNCIONA	OBSERVACIONES
Botones	✓	Los botones realizan su respectiva acción.
Diseño	✓	El diseño de la animación mantiene tamaño y resolución.
Código Action Script	✓	Código cumple con sus funciones
Movie Clip	✓	Funcionamiento correcto

Escenas	✓	Correcto ligamiento entre escenas
Frame	✓	Sin errores

Tabla 10

PRUEBA # 5:

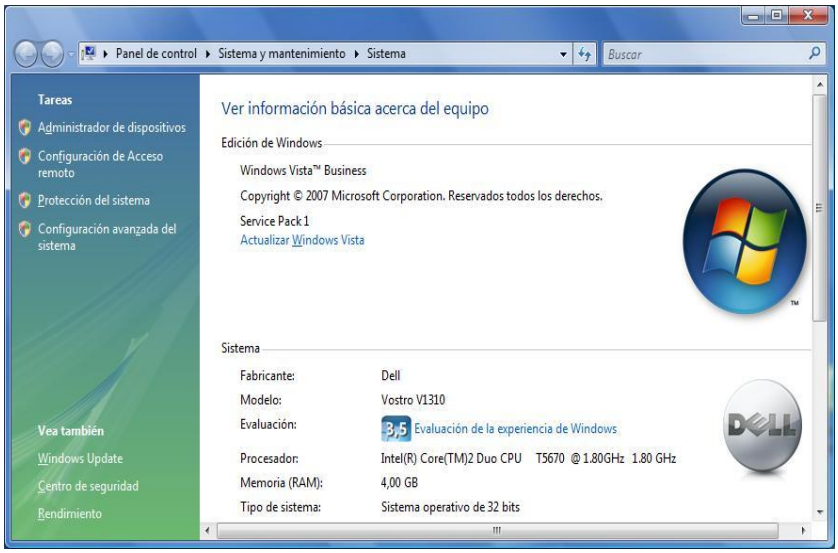


Figura 14

Para este tipo de equipo se corrieron pruebas de funcionalidad y de correcta ejecución:

HARDWARE	SOFTWARE
Equipo: HP	Sistema Operativo: Vista Home Business version 2002
Procesador: Core 2 Duo, 1.8 Ghz	Versión Adobe Flash Player: 10.0 activex
Memoria RAM: 4 GB	Versión Adobe Reader: 8.1.1
Disco Duro: 160 GB	Versión Adobe Flash Player: 10.0 plugin
Unidad de CD/DVD - RAM	
Puertos USB	

Tabla 11

RESULTADOS:

ELEMENTO A PRUEBA	FUNCIONA	OBSERVACIONES

Botones		Los botones realizan su respectiva acción.
Diseño		El diseño de la animación mantiene tamaño y resolución.
Código Action Script		Código cumple con sus funciones
Movie Clip		Funcionamiento correcto
Escenas		Correcto ligamiento entre escenas
Frame		Sin errores

Tabla 12

4.2 EJECUCION Y ENTREGA DEL PROGRMA MULTIMEDIA INTERACTIVO

4.2.1 GENERALIDADES

Terminado el plazo para la entrega del programa, procederemos a la ejecución final, que conllevará a su uso por parte de Usuarios finales. Para llegar a la finalización de este estudio y desarrollo se debieron cumplir con algunos puntos y pasar muchos filtros para conseguir calidad del producto terminado.

Los puntos son:

- Revisiones continuas para mejoras en el programa
- Revisión de la teoría.
- Manejo de los tiempos de ejecución y entrega.
- Manejo de estándares en todo el desarrollo del proyecto
- Puesta en prueba para su correcta verificación en funcionamiento y uso.

CAPÍTULO V

Conclusiones y Recomendaciones

5. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- La avanzada tecnología que hoy manejamos nos obliga a depender de los ordenadores, por lo que al contrario de ser una traba, es imperioso sacar buen provecho de ella. Por ese motivo los estudiantes que estudien Lógica y cualquier otra asignatura o ciencia tendrán la ayuda para complementar sus cátedras con ejercicios interactivos multimedia, teoría y test.

- Los estudiantes, según se ha observado en la historia desde los grandes pensadores griegos, han necesitado no solo de la teoría sino de la práctica, mirar a su alrededor para comprender el mundo que nos rodea y en el que actuamos, por ello lo imperioso de contar con una herramienta que despierte el interés de aprender por parte de los alumnos.

- El programa representa un escenario real, ya que es un producto destinado al usuario y tendrá utilidad en la vida diaria de los estudiantes, por lo que brinda un panorama de trabajo que podrá o no, según la especialización que se siga, tener ventajas directas para nuestro desenvolvimiento profesional.

- La lógica y el estudio lingüístico, muy al contrario de lo que se piense, nos ayuda en todas las ramas de las ciencias; ya que, indistintamente de la profesión, para desarrollarse en cualquiera de ellas se necesita aprender a razonar para alcanzar una comprensión debida y una mejora continua, que va mas allá de las aulas.
- Cada una de las distintas metodologías existentes para el desarrollo de software tiene sus fortalezas y debilidades. La usada en el presente estudio tiene por fortaleza la continua

mejora; mas aparecen problemas al momento en que algún cambio de fondo se presenta en medio o final del desarrollo, pues hay que revisar todo el desarrollo, teniendo en cuenta que es menor el impacto cuando se trata de cambios de forma. Con las demás metodologías, el problema es la cantidad de documentación antes, durante y después del desarrollo, así como la no inclusión del Cliente en todo el proceso.

5.2 RECOMENDACIONES

- Mas allá de una separación entre facultades, se debe considerar la interacción entre ellas, como sucede en nuestro caso, en que la *ciencia informática* a través de programas digitaliza y ameniza la teoría y practica, y la *Lógica* brinda las pautas para un correcto razonamiento; por lo cual es importante que las profesiones interactúen, otorgando así productos de investigación completos, aplicados al entorno en que nos movemos.

- Se deben adoptar metodologías ágiles que permitan obtener el máximo desarrollo, funcionalidad y desempeño del producto que se este realizando, de modo que la metodología utilizada brinde flexibilidad y obligue a una mejora continua.

- Forzar la lectura dentro de programas interactivos para despertar el interés en el uso de estas herramientas garantiza que de alguna manera se despierte expectativas en los usuarios sobre las facilidades que presta el resumen condensado de un libro, su portabilidad electrónica y también anima la investigación de libros especializados que tocan a fondo los temas tratados en los programas, es conveniente crear programas para las distintas asignaturas sin requerirse para ello que estas sean técnicas.

- El ámbito profesional obliga a los estudiantes a tener una formación holística que abarque el tema en estudio, que es el desarrollo de un programa interactivo destinado a apoyar el estudio de metodologías, la Lógica como estudio, y con las correspondientes metodologías de desarrollo, todo lo cual es aplicable a proyectos de distinta índole. Por lo que estos proyectos de investigación involucran a los estudiantes en ámbitos que engloban no solo temas de estudio de su facultad, sino muchos otros.

- Facilitar y gestionar charlas en las que el estudiante pueda ser parte activa de las mismas, dejando camino a que ellos mismo las gestionen y buscando luego temas de investigación común para distintos grupos de interés.

- Cambiar, de la manera ya señalada, algunas de las materias dictadas hoy en la facultad de Ingeniería, en su plan de estudios, el cual ayudará a impulsar una restructuración de la realidad profesional que esta viviendo nuestro país; los viajes a campo que expliquen la teoría de las respectivas cátedras, entre otras.

- Adicionar algunas materias dentro del pensum de Ingeniero en Sistemas, puesto que con la ayuda de nuevos conocimientos, como por ejemplo bases de datos, multimedia, redes, simulación, inteligencia artificial, entre tantas otras, se podría desarrollar productos más completos que abarquen problemas y soluciones a un alto nivel de requerimientos. Esto a la larga podría convertirse en un semillero de empresarios ya desde las mismas aulas de clases.

- Incentivar el estudio de las materias dictadas en clase con giras en distintas empresas que estén directa o indirectamente relacionadas con Ingeniería en Sistemas. Este procedimiento ampliara el panorama del estudiante, el cual verá como nuestra profesión brinda ayuda y es parte principal en muchas instituciones.

CAPÍTULO 6

BIBLIOGRAFIA

Internet:

- http://www.programacionextrema.org/cgi-bin/wiki.pl?TraduccionElTituloOriginal

- http://xprogramming.com/index.php

- http://www.programacionextrema.org/

Personas:

- Emilio Cerezo, Doctor

- María Cristina Luzuriaga Larrea

Papers:

Libros:

- Copi,