

UNIVERSIDAD NACIONAL DE TRUJILLO
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA ACADÉMICO PROFESIONAL DE INFORMÁTICA



CURSO

Tópicos Especiales en Sistemas Operativos

DOCENTE

Bravo Escalante Jorge David

INTEGRANTES

Marquina Gonzales Alessandra

Medina Carbajal Diego

Medina López Jahir

Mendoza Campos Renato

CICLO

VII

TRUJILLO – PERÚ

2019

TABLA DE FIGURAS

Figura 1: Características del síndrome de down 7

Figura 2: Educación en los niños con síndrome de down 10

Figura 3: Esquema de los componentes de un sistema monolítico 14

Figura 4: Esquema de los componentes de un sistema Microkernel 14

Figura 5: Directorios de un solo nivel 16

Figura 6: Directorios jerárquicos 16

Figura 7: Listas simplemente enlazadas 18

Figura 8: Listas doblemente enlazadas 18

Figura 9: Listas circulares 18

Figura 10: Listas circular doblemente enlazadas 19

Figura 11: Representación de una pila 19

Figura 12: Representación de una cola 21

Figura 13: Metodología kendall & kendall 22

Figura 14: Identificación de problemas 23

Figura 15: Determinación de los requerimientos 23

Figura 16: Análisis de las necesidades del sistema 24

Figura 17: Diseño del sistema recomendado 25

Figura 18: Desarrollo y documentación 26

Figura 19: Pruebas y mantenimiento 27

Figura 20: Implantación y evaluación del sistema 28

CAPÍTULO I: MARCO METODOLÓGICO

1. REALIDAD PROBLEMÁTICA:

El centro de educación básica especial “Carlos A. Manucci”, ubicado en Elvira García #684 brinda el servicio de enseñanza a los niños que padecen síndrome de down, dando una educación integral.

La enseñanza a los niños con síndrome de down demanda atención, cuidado y un trabajo progresivo en el salón de clases, sin embargo, el mundo actual permite a los niños tener un fácil acceso a diversos dispositivos tales como teléfonos, tablets, computadoras, etc. Los cuales cuentan con un sistema operativo enfocado para una persona promedio.

Hoy en día se han desarrollado sistemas operativos los cuales son aplicados en distintas áreas, pero es escaso ver sistemas para los niños que padecen problemas de trastorno genético.

Con este sistema operativo lo que queremos es ayudarlos a tener más éxito en las actividades con la comunidad y su integración educativa, es por esto que el grupo ha propuesto como solución el desarrollo de un sistema operativo para mejorar la interacción persona-computador para los niños con síndrome de down en el centro de educación básica especial “Carlos A. Manucci”.

FORMULACIÓN DE PROBLEMA

¿Mejorará la interacción persona - computador el desarrollo de un sistema operativo para los niños con síndrome de down en el Centro de Educación Básica Especial “Carlos A. Manucci”?

1. ANTECEDENTES

1. Antecedente 1: Aplicación para lectura rápida en niños con síndrome de down

Autor: Alberto delgado, phd. Profesor asociado

Lugar: Departamento de psicología. Universidad nacional de Colombia.

Año: 1999

Aporte: La Confederación Española de Organizaciones a favor de las Personas con Discapacidad Intelectual o del Desarrollo han creado una nueva app para facilitar la lectura a personas con discapacidad intelectual o del desarrollo, o cualquier discapacidad que suponga dificultades a la hora de leer

1. Antecedente 2: Clasificación de imágenes y sonidos

Autor: Álvaro Martínez Martínez

Lugar: Escuela Psicología, Madrid

Año: 2014

Aporte: Esta aplicación permite aprender a reconocer palabras, identificando y emparejando dibujos con sonidos. Tiene distintos niveles de dificultad que el niño irá superando. Dicha aplicación consta de 96 palabras y, además, da la posibilidad de ordenarlas de manera personalizada e incluso añadir nuevas palabras familiares para el niño.

1. Antecedente 3: Técnicas de aprendizaje para niños con síndrome de down y autismo

Autor: Dra. Nora la Serna Palomi

Año: 2008

Lugar: Madrid, España

Aporte: App para niños autistas y con síndrome de down. El objetivo es para mejorar las competencias básicas de niños y adolescentes autistas y con síndrome de down. Incluye actividades de matemáticas, lenguaje, conocimiento del entorno, así como autonomía y habilidades sociales. Apta tanto para pequeños y jóvenes con discapacidad

cognitiva, visual o auditiva. Pensada para el aula como herramienta de apoyo.

1. **Antecedente 4: Aplicación para facilitar la comunicación entre persona-computador.**

Autor: Eduardo Rivero.

Año: 2001

Aporte: una App de AAC (aprendizaje asistido por computador) para facilitar la comunicación del niño, que podrá dar aviso de sus deseos o identificar rápidamente determinados objetos y situaciones. Cuenta con grabación de voz y capturas especiales para sumar a los botones predeterminados.

1. **Antecedente 5: Software para estimulo cognitivo de niños con síndrome de down**

Autor: Deya cano. Editora en jefe de salud180. Egresada de la septién.

Año: 2012

Aporte: Una App de realidad virtual que permite que los niños trabajen varios sentidos a la vez por medio de estímulos únicos y experiencias inolvidables. La aplicación se ha creado con el fin de estimular cognitivamente a niños con esta anomalía durante la etapa preescolar ya que resulta muy atractiva para los pequeños y les facilita el aprendizaje en temas básicos. Además, permite trabajar elementos de motricidad fina y coordinación viso-manual. La aplicación está compuesta por varias actividades, cada una con diferentes ejercicios, desarrollados según las necesidades curriculares y educativas de los niños. Las actividades están diferenciadas por niveles, dependiendo de las habilidades y capacidades de cada niño con síndrome de down. Puede ser usada a través de tabletas y smartphones que contengan sistemas operativos ios o android. Las actividades se accionan por medio de un 'target', que es la tarjeta o ficha que capta la cámara incluida en los dispositivos móviles. La cámara reconoce la ficha y la traduce en información, enviando textos, figuras 3d, animaciones, entre otras, sobre la pantalla del dispositivo; este es el momento donde aparece la realidad aumentada con la cual se puede interactuar fácilmente.

1.

JUSTIFICACIÓN

1.

JUSTIFICACIÓN ECONÓMICA

Sabemos que es importante el uso de un computador, ya sea para realizar tareas personales o algún trabajo secular. Los niños que padecen trastorno también necesitan saber usar un sistema operativo y esto lo aprenden en las escuelas, allí normalmente suelen contratar algún profesor para realizar esta tarea de enseñanza, pero con la implementación de un sistema operativo especializado en los niños con síndrome de down, esto ya no será necesario porque el aprendizaje será intuitivo y esto va a beneficiar a que las escuelas ahorren dinero en la contratación de personal dedicado, mejorando de esta forma el proceso de enseñanza a un menor costo económico.

JUSTIFICACIÓN SOCIAL

Como un fin primordial para la creación de nuestro sistema operativo es, ya que los niños con síndrome de Down no son casi capaces de usar bien la tecnología digital (celulares, Tablet y ordenadores) las cuales mayormente se usan para interactuar en las redes sociales y/o realizar actividades; con la implementación de nuestro sistema operativo, podremos hacer que los niños puedan interactuar mejor con la computadora.

JUSTIFICACION TECNOLÓGICA

Nuestro proyecto busca optimizar ciertas tareas que se llevan a cabo en el área de educadores especializados, como es el caso de interacción persona-computador en los niños con síndrome de Down, haciendo uso de la tecnología.

La implementación de nuestro sistema operativo busca generar mejoras en el área mencionada, ayudando a los especialistas con la interacción persona-computador en los niños con síndrome de Down.

Con lo propuesto en el sistema operativo se busca mejorar el aprendizaje, enfatizando de este modo que los niños puedan utilizar mejor la computadora.

1. OBJETIVOS

1. OBJETIVO GENERAL

Mejorar la interacción persona - computador para los niños con síndrome de down en el Centro de Educación Básica Especial “Carlos A. Manucci” mediante el desarrollo de un sistema operativo.

1. OBJETIVOS ESPECÍFICOS

- Aumentar la satisfacción de un niño al usar una computadora.
- Disminuir el tiempo de aprendizaje de uso de un sistema operativo.
- Aumentar la satisfacción de los profesores al enseñar este sistema operativo a sus alumnos.

1. **HIPOTESIS:**

El desarrollo de un sistema operativo si lograra mejorar la interacción persona – computador en los niños con síndrome de down en el Centro de Educación Básica Especial “Carlos A. Manucci”.

1. **VARIABLES:**

1. **VARIABLES DEPENDIENTES**

Interacción persona – computador en los niños con síndrome de down

1. **VARIABLES INDEPENDIENTES**

Un sistema operativo

CAPÍTULO II: MARCO REFERENCIAL

1. **MARCO TEORICO**

1. **SÍNDROME DE DOWN**

1. **CONCEPTO**

El Síndrome de Down (SD), también llamado trisomía 21, es la causa más frecuente de retraso mental identificable de origen genético. Se trata de una anomalía cromosómica que tiene una incidencia de 1 de cada 800 nacidos, y que aumenta con la edad materna. Es la cromosomopatía más frecuente y mejor conocida

1. **CARACTERÍSTICAS**

Los niños con SD se caracterizan por presentar una gran hipotonía e hiperlaxitud ligamentosa. Fenotípicamente presentan unos rasgos muy característicos.

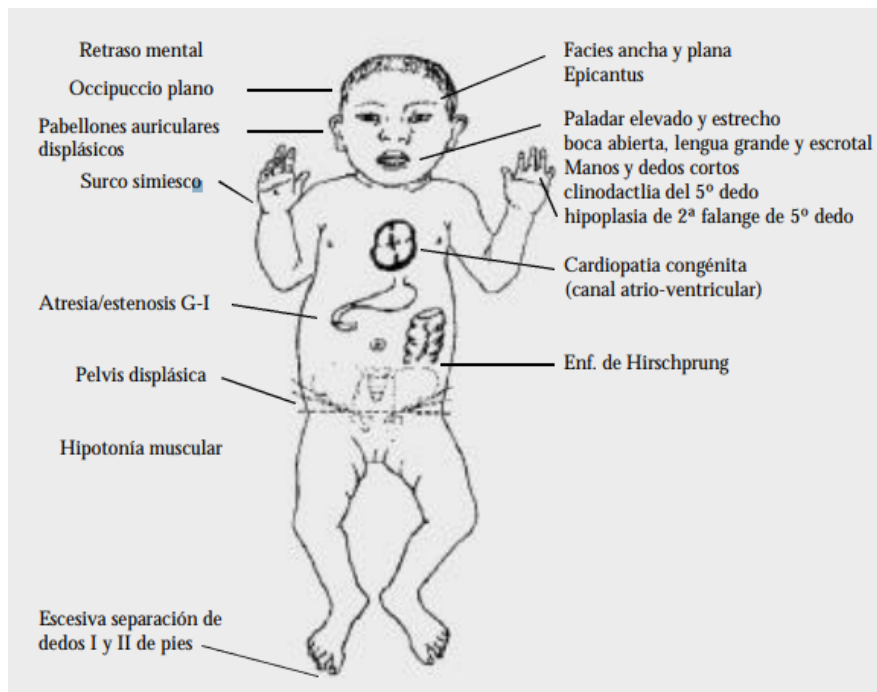


Figura : Características del síndrome de down

- Cabeza y cuello: leve microcefalia con braquicefalia y occipital aplanado. El cuello es corto.
- Cara: los ojos son “almendrados”, y si el iris es azul suele observarse una pigmentación moteada, son las manchas de brushfield. Las hendiduras palpebrales siguen una dirección oblicua hacia arriba y afuera y presentan un pliegue de piel que cubre el ángulo interno y la carúncula del ojo (epicanto). la nariz es pequeña con la raíz nasal aplanada. la boca también es pequeña y la profusión lingual característica. las orejas son pequeñas con un hélix muy plegado y habitualmente con ausencia del lóbulo. el conducto auditivo puede ser muy estrecho.
- Manos y pies: manos pequeñas y cuadradas con metacarpianos y falanges cortas (braquidactilia) y clinodactilia por hipoplasia de la falange media del 5º dedo. Puede observarse un surco palmar único. En el pie existe una hendidura entre el primer y segundo dedo con un aumento de la distancia entre los mismos (signo de la sandalia).
- Genitales: el tamaño del pene es algo pequeño y el volumen testicular es menor que el de los niños de su edad, una criptorquidia es relativamente frecuente en estos individuos.
- Piel y faneras: la piel es redundante en la región cervical sobretudo en el período fetal y neonatal. Puede observarse livedo reticularis (cutis

marmorata) de predominio en extremidades inferiores. Con el tiempo la piel se vuelve seca e hiperqueratósica.

El retraso mental es constante en mayor o menor grado.

1. RIESGO DE RECURRENCIA

El SD puede diagnosticarse prenatalmente realizando un estudio citogenético de vellosidades coriónicas o de líquido amniótico. El riesgo depende de la edad materna, pero también del cariotipo de los progenitores.

En el caso que se trate de una trisomía 21, el riesgo de recurrencia para las mujeres de edad superior a los 30 años es el mismo que le da su edad. En las mujeres más jóvenes es algo más alto. En el caso de que exista una translocación y alguno de los progenitores sea portador, no influye la edad materna, pero existe un riesgo más alto de recurrencia si el portador de la translocación es la madre. En el caso de que alguno de los padres tenga una translocación Robertsoniana entre dos cromosomas 21 el riesgo de recurrencia es del 100% independientemente del sexo que lo transmita. Si ninguno de los progenitores es portador de una translocación el riesgo de recurrencia es de alrededor de un 2-3%.

1. LA EDUCACIÓN EN LOS NIÑOS CON SÍNDROME DE DOWN

El síndrome de Down es una alteración genética producida por la presencia de un cromosoma extra (entero o una parte de él) en la pareja cromosómica 21. Por lo tanto, las células de estas personas tienen 47 cromosomas con tres cromosomas en dicho par (de ahí el nombre de trisomía 21), cuando lo habitual es que sólo existan dos. El motivo de esta anomalía genética no se conoce.

Aunque no ocurre en todos los casos, muchos niños con síndrome de Down tienen patologías asociadas. Se trata complicaciones de salud relacionadas con su alteración genética: cardiopatías congénitas, hipertensión pulmonar, problemas auditivos o visuales, anomalías intestinales, neurológicas, endocrinas, etc. Estas situaciones requieren cuidados específicos y sobre todo un adecuado seguimiento desde el nacimiento. En cuanto a capacidad intelectual, suelen tener un retraso mental de leve a moderado. Pero cada caso es distinto, existiendo personas con síndrome de Down que han llegado incluso a conseguir una titulación universitaria.



Figura : Educación en los niños con síndrome de down

1. DIFICULTADES DE APRENDIZAJE

Como hemos dicho, cada persona con síndrome de Down es diferente, con sus dificultades y habilidades. No obstante, podemos encontrar una serie de características bastante comunes que dificultan o retrasan el aprendizaje de estos niños y niñas:

- En general, el proceso de aprendizaje es más lento.
- Suelen precisar de más tiempo para conseguir los objetivos curriculares, lo que implica más años de escolaridad.
- Presentan dificultades con el procesamiento de la información: tanto en la recepción de la misma, como a la hora de aplicarla a situaciones concretas.
- Les cuesta correlacionar y elaborar los conceptos aprendidos para tomar decisiones secuenciales y lógicas
- Problemas para manejar diversas informaciones, especialmente si se les presentan de forma simultánea
- Dificultades de abstracción y de conceptualización por sus limitaciones cognitivas.
- Mayor facilidad para olvidar lo aprendido.
- Escasa iniciativa y pro actividad.
- Menor capacidad de respuesta y reacción frente a los problemas y situaciones adversas.
- No suelen pedir ayuda cuando no entienden algo o les cuesta llevar a cabo una actividad.

- Tratan de evitar enfrentarse a nuevas actividades o retos.

1. MÉTODOS DE APRENDIZAJE

Las dificultades de aprendizaje anteriormente mencionadas son subsanables o mejorables si se utilizan una serie de recomendaciones y estrategias en la enseñanza de los alumnos con síndrome de Down.

- Dada su mejor percepción visual, aprenden con mayor facilidad si se apoyan en signos, gestos, señales, imágenes, dibujos, gráficos, pictogramas o cualquier otro tipo de clave visual.
- Aprovechar su alta capacidad de observación y de imitación para favorecer y reforzar la adquisición de los distintos aprendizajes, utilizando el denominado aprendizaje por observación o vicario siempre que sea posible.
- Su capacidad de aprendizaje es continua, es decir, no se produce ningún parón ni estancamiento. Esto debe ser aprovechado para, aunque sea en un mayor espacio de tiempo, acabar consiguiendo las competencias planteadas.
- Utilizar actividades y ejemplos concretos para contrarrestar su déficit de pensamiento abstracto.
- Es necesario aplicar programas específicos de autonomía personal, entrenamiento en habilidades sociales y educación emocional, dirigidos a promover su independencia.
- En la mayoría de casos es necesario confeccionar adaptaciones curriculares individuales, puesto que en el aula ordinaria pertenecen al grupo de alumnos con necesidades educativas especiales (NEE).

1. SISTEMAS OPERATIVOS

El sistema operativo es el principal programa que se ejecuta en toda computadora de propósito general. Los hay de todo tipo, desde muy simples hasta terriblemente complejos, y entre más casos de uso hay para el cómputo en la vida diaria, más variedad habrá en ellos

1. FUNCIONES Y OBJETIVOS

- **Abstracción:** Los programas no deben tener que preocuparse de los detalles de acceso a hardware, o de la configuración particular de una computadora.

El sistema operativo se encarga de proporcionar una serie de abstracciones para que los programadores puedan enfocarse en resolver las necesidades particulares de sus usuarios. Un ejemplo de tales abstracciones es que la información está organizada en archivos y directorios (en uno o muchos dispositivos de almacenamiento).

- **Administración de recursos:** Una sistema de cómputo puede tener a su disposición una gran cantidad de recursos (memoria, espacio de almacenamiento, tiempo de procesamiento, etc.), y los diferentes procesos

que se ejecuten en él compiten por ellos. Al gestionar toda la asignación de recursos, el sistema operativo puede implementar políticas que los asignen de forma efectiva y acorde a las necesidades establecidas para dicho sistema.

- **Aislamiento:** En un sistema multiusuario y multitarea cada proceso y cada usuario no tendrá que preocuparse por otros que estén usando el mismo sistema —Idealmente, su experiencia será la misma que si el sistema estuviera exclusivamente dedicado a su atención (aunque fuera un sistema menos poderoso).

Para implementar correctamente las funciones de aislamiento hace falta que el sistema operativo utilice hardware específico para dicha protección.

1. ORGANIZACIÓN DE LOS SISTEMAS OPERATIVOS

Hay dos formas primarias de organización interna del sistema operativo:

- **Monolíticos:** La mayor parte de los sistemas operativos históricamente han sido monolíticos: esto significa que hay un sólo proceso privilegiado que opera en modo supervisor, y dentro del cual se encuentran todas las rutinas para las diversas tareas que realiza el sistema operativo.

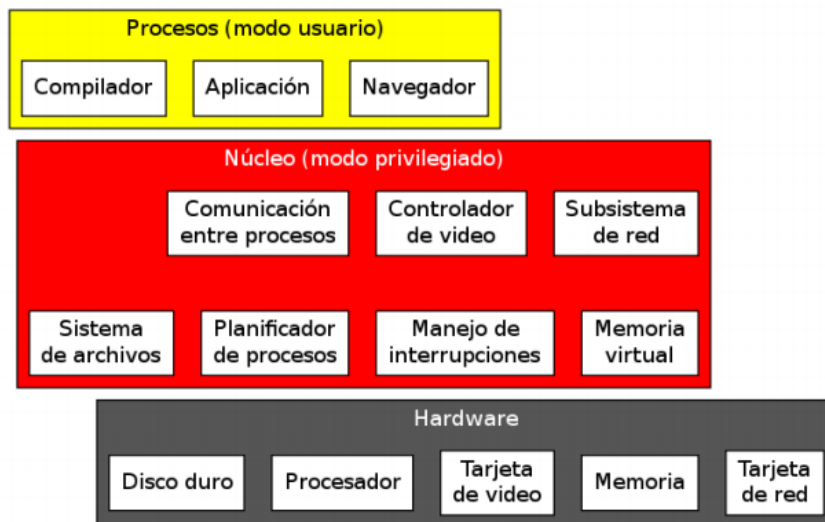


Figura : Esquema de los componentes de un sistema monolítico

- **Microkernel:** El núcleo del sistema operativo se mantiene en el mínimo posible de funcionalidad, descargando en procesos especiales sin privilegios las tareas que implementan el acceso a dispositivos y las diversas políticas de uso del sistema.

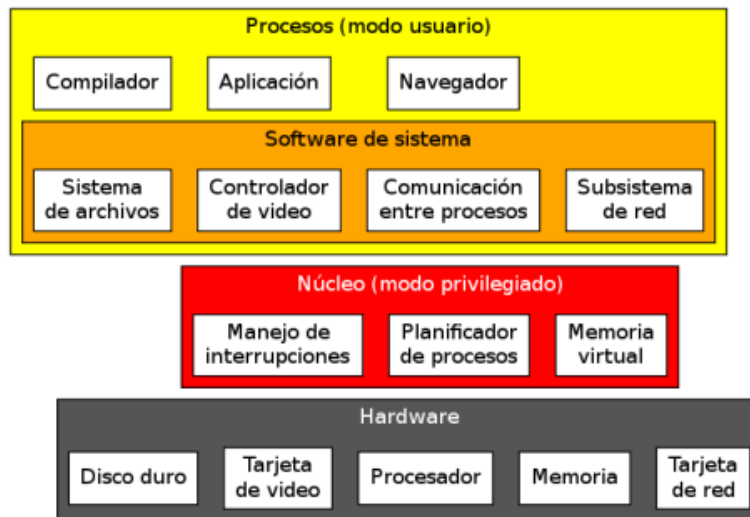


Figura : Esquema de los componentes de un sistema Microkernel

La principal ventaja de diseñar un sistema siguiendo un esquema monolítico es la simplificación de una gran cantidad de mecanismos de comunicación, que lleva a una mayor velocidad de ejecución (al requerir menos cambios de contexto para cualquier operación realizada). Además, al manejarse la comunicación directa como paso de estructuras en memoria, el mayor acoplamiento permite más flexibilidad al adecuarse para nuevos requisitos (al no tener que modificar no sólo al núcleo y a los procesos especiales, sino también la interfaz pública entre ellos).

Por otro lado, los sistemas Microkernel siguen esquemas lógicos más limpios, permiten implementaciones más elegantes y facilitan la comprensión por separado de cada una de sus piezas. Pueden auto-repararse con mayor facilidad, dado que en caso de fallar uno de los componentes (por más que parezca ser de muy bajo nivel), el núcleo puede reiniciarlo o incluso reemplazarlo.

1. DIRECTORIOS O CARPETAS

Para llevar el registro de los archivos, los sistemas de archivos por lo general tienen directorios o carpetas, que en muchos sistemas son también archivos.

- **Sistemas de directorios de un solo nivel:** La forma más simple de un sistema de directorios es tener un directorio que contenga todos los archivos. Algunas veces se le llama directorio raíz, pero como es el único, el nombre no importa mucho. En las primeras computadoras personales, este sistema era común, en parte debido a que sólo había un usuario.

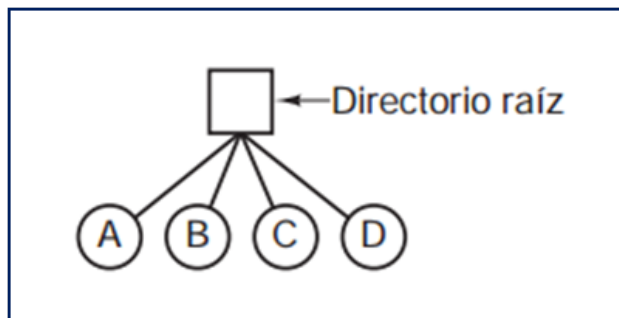


Figura : Directorios de un solo nivel

- **Sistemas de directorios jerárquicos:** Tener un solo nivel es adecuado para aplicaciones dedicadas simples (e incluso se utilizaba en las primeras computadoras personales), pero para los usuarios modernos con miles de archivos, sería imposible encontrar algo si todos los archivos estuvieran en un solo directorio.

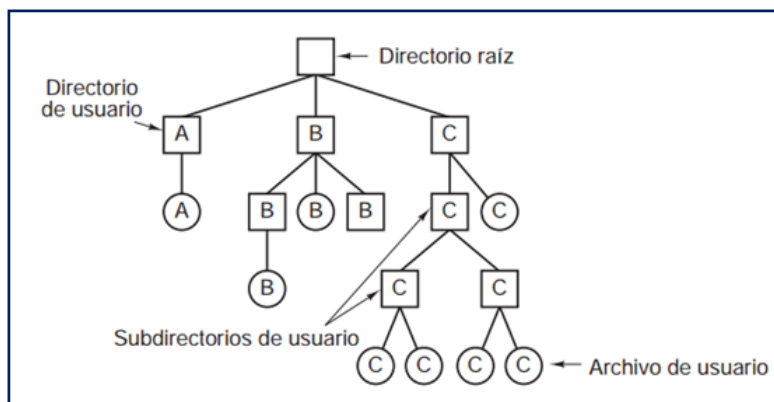


Figura : Directorios jerárquicos

En consecuencia, se necesita una forma de agrupar los archivos relacionados. Por ejemplo, un profesor podría tener una colección de archivos que en conjunto formen un libro que está escribiendo para un curso, una segunda colección de archivos que contienen programas enviados por los estudiantes para otro curso, un tercer grupo de archivos que contenga el código de un sistema de escritura de compiladores avanzado que está construyendo, un cuarto grupo de archivos que contienen proposiciones de becas, así como otros archivos para correo electrónico, minutas de reuniones, artículos que está escribiendo, juegos, etcétera. Lo que se necesita es una jerarquía (es decir, un árbol de directorios). Con este esquema, puede haber tantos directorios como se necesite para agrupar los archivos en formas naturales. Además, si varios usuarios comparten un servidor

de archivos común, como se da el caso en muchas redes de empresas, cada usuario puede tener un directorio raíz privado para su propia jerarquía.

1. ESTRUCTURAS DE DATOS

Las estructuras q vamos a utilizar para poder realizar este proyecto son:

A. Listas Enlazadas:

Es una colección de elementos dispuestos uno detrás del otro, en la que cada elemento se conecta al siguiente por un “Enlace” o “Puntero”.

Los nodos de las listas al igual que las colas y pilas, está compuesta por una parte de información (que puede ser datos enteros, flotantes, caracteres, estructuras.) y el puntero que mantiene el enlace entre un nodo y otro.

Existen varios tipos de Listas, pero para efectos de comprensión y sintetización, hablaremos de cuatro tipos esenciales de listas:

- **Lista simplemente enlazada:** Cada nodo, contiene un único apuntador hacia el siguiente nodo, por lo cual hace de él una estructura muy eficiente, ya que el último de la lista apunta hacia null, por ello, es fácil hacer recorridos directos. La implementación la puede ver en el **Anexo 1**.



Figura : Listas simplemente enlazadas

- **Listas Doblemente enlazadas:** Esta lista se caracteriza por que sus nodos contienen dos punteros, uno hacia el nodo siguiente y otro hacia el nodo anterior. La implementación la puede ver en el **anexo 2**.

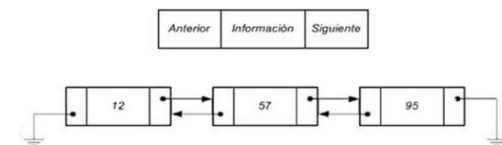


Figura : Listas doblemente enlazadas

- **Listas Circulares:** Este tipo de lista, es sólo una extensión de las listas simplemente enlazadas, con la diferencia que el último

elemento se enlaza al primer elemento de la lista, lo cual permite el recorrido en forma de anillo. La implementación la puede ver en el **Anexo 3**.

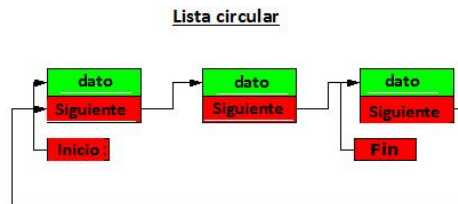


Figura : Listas circulares

- **Lista Circular Doblemente enlazada:** Quizá este tipo de lista, sea la más compleja, ya que es la combinación de la lista circular y las doblemente enlazadas, ya que es una lista doblemente enlazada donde el primer elemento se conecta con el último y viceversa. La implementación la puede ver en el **anexo 4**.

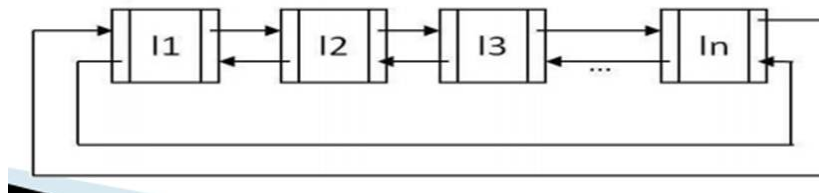


Figura : Listas circular doblemente enlazadas

- A. **Pilas:** Una pila (stack) es una colección ordenada de elementos en la cual los datos se insertan o se retiran por el mismo extremo llamado “parte superior” de la pila. Si tenemos un par de elementos en la pila, uno de ellos debe estar en la parte superior de la pila, que se considera “el más alto” en la pila que el otro. La implementación la puede ver en el **Anexo 5**.

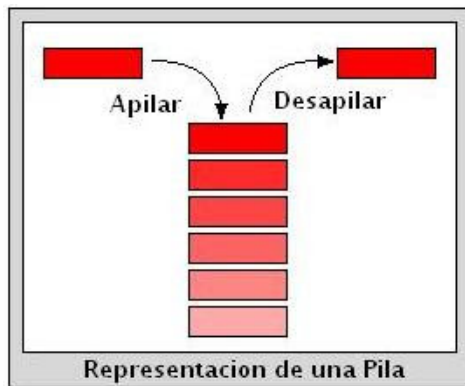


Figura : Representación de una pila

a. **Operaciones básicas en Pilas**

- **Apilar (Push):** Existe solamente un lugar en donde cualquier elemento puede ser agregado a la pila. Después de haber insertado el nuevo elemento, G ahora es el elemento en la cima.
- **Desapilar (Pop):** Basta indicar que sea retirado un elemento. No podemos decir “retirar C”, porque C no está en la cima de la pila.

a. **Aplicaciones de las pilas:**

- **Navegador Web:** almacenan los sitios previamente visitados, cuando el usuario quiere regresar (presiona el botón de retroceso), simplemente se extrae la última dirección (pop) de la pila de sitios visitados.
 - **Editores de texto:** Los cambios efectuados se almacenan en una pila, usualmente implementada como arreglo, cada usuario puede deshacer los cambios mediante la operación “pop”, la cual extrae el estado del texto antes del último cambio realizado.
- A. **Colas:** Una cola es un tipo especial de lista abierta en la que sólo se pueden insertar nodos en uno de los extremos de la lista y sólo se pueden eliminar nodos en el otro. Además, como sucede con las pilas, las escrituras de datos siempre son inserciones de nodos, y las lecturas siempre eliminan el nodo leído. Este tipo de lista es conocido como lista FIFO (First In First Out), el primero en entrar es el primero en salir. La implementación la puede ver en el **Anexo 6**.

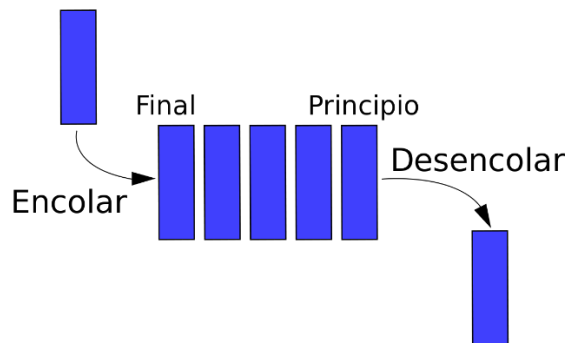


Figura : Representación de una cola

a. **Operaciones básicas en colas:** Las operaciones básicas de una cola son “encolar” (meter) y “desencolar”.

- Encolar: añade un nuevo elemento al final de la cola.
- Desencolar: elimina (saca) el primer elemento de la cola

Otras operaciones usualmente incluidas en el tipo abstracto COLA son:

- isEmpty (está vacía): verifica si la cola está vacía.
- isFull (está llena): verifica si la cola está llena.

a. **Aplicaciones de las colas:** En general, operaciones en:

- Redes de computadoras.
- Trabajos enviados a una impresora.
- Solicitudes a un servidor.
- Clientes solicitando ser atendidos por una telefonista.

1. METODOLOGIA KENDALL Y KENDALL

Según la metodología de Kendall y Kendall el ciclo de vida de un sistema consta de siete partes: siendo la primera la identificación del problema, la segunda identificación de requisitos de información, la tercera es el análisis de las necesidades del sistema, la cuarta es el diseño del sistema recomendado, la quinta desarrollo y documentación del sistema, la sexta prueba y mantenimiento y la última implementación y evaluación. Cada fase se explica por separado, pero nunca se realizan como pasos aislados, más bien es posible que algunas actividades se realicen de manera simultánea, y algunas de ellas podrían repetirse.

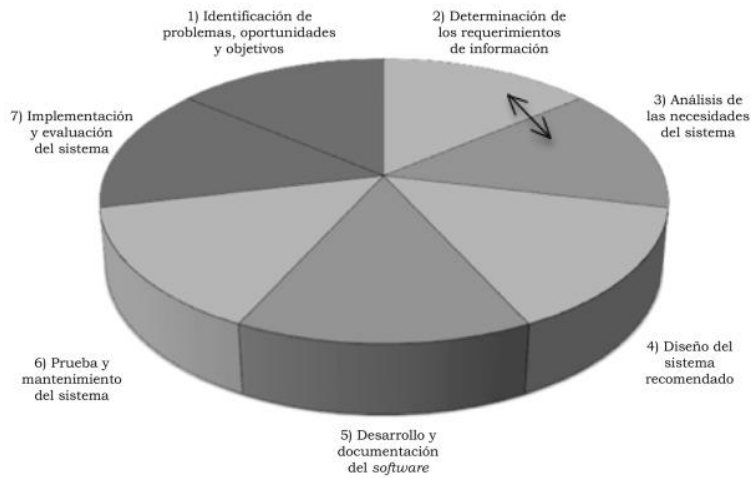


Figura : Metodología kendall & kendall

Estas se desarrollan en 7 fases:

1. IDENTIFICACIÓN DE PROBLEMAS, OPORTUNIDADES Y OBJETIVOS

Esta fase es crucial para el éxito del resto del proyecto requiere que se observe de forma objetiva lo que ocurre en una organización, luego en conjunto con otros miembros de la organización hacer notar los problemas. Las oportunidades son aquellas situaciones que se considera que pueden mejorarse, perfeccionarse mediante el uso de los sistemas de información. También es un componente importante de la primera fase, en esta etapa se deberá descubrir lo que la organización intenta realizar, luego determinar si el uso de los sistemas de información apoyaría a la organización para alcanzar sus metas.



Figura : Identificación de problemas

1. DETERMINACIÓN DE LOS REQUERIMIENTOS DE INFORMACIÓN

Esto se hace a partir de los usuarios particularmente involucrados, para determinar los requerimientos de información dentro de una organización pueden utilizarse diversos instrumentos, los cuales incluyen: muestreo, el estudio de los datos y formas usadas para la organización, la entrevista, los cuestionarios; la observación de la conducta de quien tomo las decisiones, así como de su ambiente. Se hace todo lo posible por identificar qué información requiere el usuario para desempeñar sus tareas.



Figura : Determinación de los requerimientos

1. ANALISIS DE LAS NECESIDADES DEL SISTEMA

Se analizan las necesidades propias del sistema, para ello existen herramientas y técnicas diseñadas para tal fin, estas incluyen entre otras el uso de los diagramas de flujo de datos que cuentan con una técnica estructurada para representar en forma gráfica la entrada de datos a la organización, los procesos y la salida de información. También se analizan las decisiones estructuradas por realizar, que son decisiones donde las condiciones, condiciones alternativas, acciones y reglas de acción podrán determinarse.



Figura : Análisis de las necesidades del sistema

1. DISEÑO DEL SISTEMA RECOMENDADO

Se usa la información recolectada con anterioridad y se elabora el diseño lógico de sistemas de información, se diseña también procedimiento es precisos de captura de datos, con la finalidad de que los datos que se introducen en el sistema de información, sean los correctos. Esta etapa también incluye el diseño de los archivos o la base de datos que almacenará aquellos datos requeridos por quien toma las decisiones en la organización.



Figura : Diseño del sistema recomendado

1. DESARROLLO Y DOCUMENTACIÓN

En la quinta fase del ciclo del desarrollo de sistemas, el analista trabaja de manera conjunta con los programadores para desarrollar cualquier software original necesario. Entre las técnicas estructuradas para diseñar y documentar software se encuentran los diagramas de estructuras, los diagramas de Nassi-Shneiderman y el pseudocódigo.

Durante esta fase el analista trabaja con los usuarios para desarrollar documentación efectiva para el software, como manuales de

procedimientos, ayuda en línea y sitios web que incluyan respuestas a preguntas frecuentes en archivos “léame” que se integrarán al nuevo software.

La documentación indica a los usuarios cómo utilizar el sistema y qué hacer en caso de que surjan problemas derivados de este uso.

Los programadores desempeñan un rol clave en esta fase porque diseñan, codifican y eliminan errores sintácticos de los programas de cómputo, algunas de sus tareas son:

- Evaluar los procedimientos que va a ser desarrollados por el programador.
- Mostrar y explicar cada procedimiento, función y operación al programador.
- Elaborar manuales de procedimientos internos del sistema.
- Elaborar manuales externos de ayuda a los usuarios del sistema.
- Elaborar demostraciones para los usuarios y la interacción con distintas interfaces.
- Elaborar actualizaciones para los diferentes procedimientos.
- Elaborar un informe con el tiempo que se llevó construir cada procedimiento.



Figura : Desarrollo y documentación

1. PRUEBAS Y MANTENIMIENTO

Todo sistema de información debe probarse antes de ser utilizado, ya que el costo es menor si se detectan los problemas antes de que entre en funcionamiento. En un principio, se hace una serie de pruebas, con datos tipo, para identificar las posibles fallas del sistema, más adelante, se utilizarán los datos del sistema real.

Una parte de las pruebas la realizan los programadores solos, y otra la llevan a cabo de manera conjunta con los analistas de sistemas. Primero se realizan las pruebas con datos de muestra para determinar

con precisión cuáles son los problemas y posteriormente se realiza otra con datos reales del sistema actual.

El mantenimiento del sistema de información y su documentación empiezan en esta fase y se llevan de manera rutinaria durante toda su vida útil.

Debemos realizar las siguientes:

- Realizar la programación de las pruebas del sistema.
- Realizar un instrumento para evaluar el sistema de información.
- El programador deberá elaborar un resumen de las pruebas del sistema.
- El analista deberá realizar un informe de sus pruebas y discutirlo con el –programador.
- Elaborar la planificación de las horas del mantenimiento del sistema.
- Elaborar la lista de las operaciones que pudieran sufrir modificaciones de códigos.



Figura : Pruebas y mantenimiento

1. IMPLANTACIÓN Y EVALUACIÓN DEL SISTEMA

En esta fase se realizan las siguientes funciones:

- Planificar gradualmente la conversión del sistema anterior.
- Instalar los equipos de hardware necesarios para el funcionamiento del software creado.
- Capacitar por medio de talleres a los usuarios en el manejo de equipos y software creados
- Evaluar la adaptabilidad de los usuarios al sistema.

Esta es la última fase del desarrollo de sistemas, y aquí el analista participa en la implementación del sistema de información. En esta fase se capacita a los usuarios en el manejo del sistema. Parte de la

capacitación la imparten los fabricantes, pero la supervisión de ésta es responsabilidad del analista de sistemas.

Se menciona la evaluación como la fase final del ciclo de vida del desarrollo de sistemas principalmente en áreas del debate. En realidad, la evaluación se lleva a cabo durante cada una de las fases.

El trabajo de sistemas es cíclico, cuando un analista termina una fase del desarrollo de sistemas y pasa a la siguiente, el surgimiento de un problema podría obligar a regresar a la fase previa y modificar el trabajo realizado.



Figura : Implantación y evaluación del sistema

ANEXOS

1. Implementación de la lista simplemente enlazada

```
10 #include <iostream>
11 #include <stdlib.h>
12 using namespace std;
13
14 struct nodo{
15     int nro;           // en este caso es un numero entero
16     struct nodo *sgte;
17 };
18
19 typedef struct nodo *Tlista;
20
21 void InsertarInicio(Tlista &lista, int valor)
22 {
23     Tlista q;
24     q = new(struct nodo);
25     q->nro = valor;
26     q->sgte = lista;
27     lista = q;
28 }
29
30 void InsertarFinal(Tlista &lista, int valor)
31 {
32     Tlista t, q = new(struct nodo);
33
34     q->nro = valor;
35     q->sgte = NULL;
36
37     if(lista==NULL)
38     {
```

```
39         lista = q;
40     }
41     else
42     {
43         t = lista;
44         while(t->sgte!=NULL)
45         {
46             t = t->sgte;
47         }
48         t->sgte = q;
49     }
50 }
51
52 int insertarAntesDespues()
53 {
54     int _op, band;
55     cout<<endl;
56     cout<<"\t 1. Antes de la posicion      "<<endl;
57     cout<<"\t 2. Despues de la posicion    "<<endl;
58
59     cout<<"\n\t Opcion : "; cin>> _op;
60
61     if(_op==1)
62         band = -1;
63     else
64         band = 0;
65
66     return band;
67 }
```



```
VERTE IR - YouTube x x Listas Enlazadas Simples Lineales: x Facebook x +
https://blog.martincruz.me/2012/10/listas-enlazadas-simples-en-c.html
69
70 void insertarElemento(Tlista &lista, int valor, int pos)
71 {
72     Tlista q, t;
73     int i;
74     q = new(struct nodo);
75     q->nro = valor;
76
77     if(pos==1)
78     {
79         q->sgte = lista;
80         lista = q;
81     }
82     else
83     {
84         int x = insertarAntesDespues();
85         t = lista;
86
87         for(i=1; t!=NULL; i++)
88         {
89             if(i==pos+x)
90             {
91                 q->sgte = t->sgte;
92                 t->sgte = q;
93                 return;
94             }
95             t = t->sgte;
96         }
97     }
98     cout<<" Error...Posicion no encontrada..!"<<endl;
99 }
```

```
Alex Rose - Toda (Remix) Ft. x x Listas Enlazadas Simples Lineales: x Facebook x +
https://blog.martincruz.me/2012/10/listas-enlazadas-simples-en-c.html
98     cout<<" Error...Posicion no encontrada..!"<<endl;
99 }
100
101 void buscarElemento(Tlista lista, int valor)
102 {
103     Tlista q = lista;
104     int i = 1, band = 0;
105
106     while(q!=NULL)
107     {
108         if(q->nro==valor)
109         {
110             cout<<endl<<" Encontrado en posición "<< i <<endl;
111             band = 1;
112         }
113         q = q->sgte;
114         i++;
115     }
116
117     if(band==0)
118         cout<<"\n\n Numero no encontrado..!"<< endl;
119 }
120
121 void reportarLista(Tlista lista)
122 {
123     int i = 0;
124
125     while(lista != NULL)
126     {
127         cout << " << i+1 <<" " << lista->nro << endl;
128     }
129 }
```

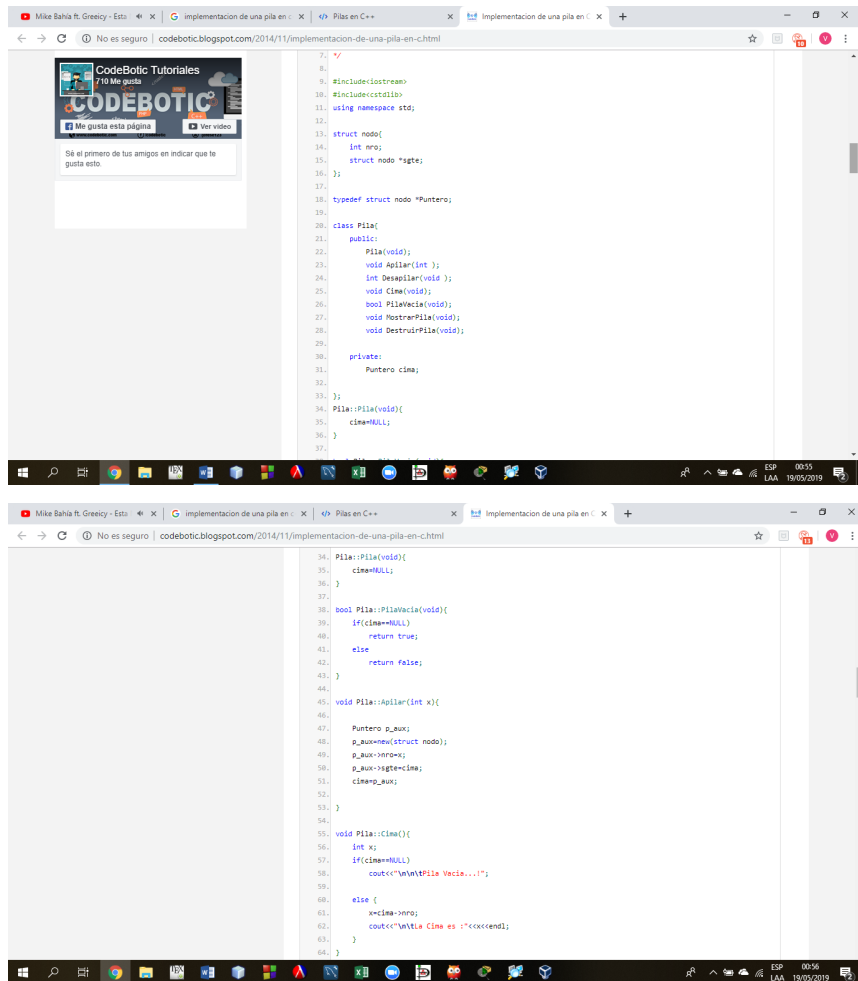
```
125 while(lista != NULL)
126 {
127     cout << "i+1 <<" << lista->nro << endl;
128     lista = lista->sgte;
129     i++;
130 }
131 }
132
133 void eliminarElemento(Tlista &lista, int valor)
134 {
135     Tlista p, ant;
136     p = lista;
137
138     if(lista=NULL)
139     {
140         while(p!=NULL)
141         {
142             if(p->nro==valor)
143             {
144                 if(p==lista)
145                     lista = lista->sgte;
146                 else
147                     ant->sgte = p->sgte;
148
149                 delete(p);
150                 return;
151             }
152             ant = p;
153             p = p->sgte;
154         }
155     }
156 }
157
158 void eliminarRepetidos(Tlista &lista)
159 {
160     Tlista q, ant;
161     q = lista;
162     ant = lista;
163
164     while(q!=NULL)
165     {
166         if(q->nro==valor)
167         {
168             if(q==lista)
169             {
170                 lista = delete(q);
171                 q = lista->sgte;
172             }
173             else
174             {
175                 ant->sgte = q->sgte;
176                 delete(q);
177             }
178         }
179         ant = q;
180         q = q->sgte;
181     }
182 }
183
184 void imprimir(Tlista &lista)
185 {
186     while(lista != NULL)
187     {
188         cout << "i+1 <<" << lista->nro << endl;
189         lista = lista->sgte;
190     }
191 }
```

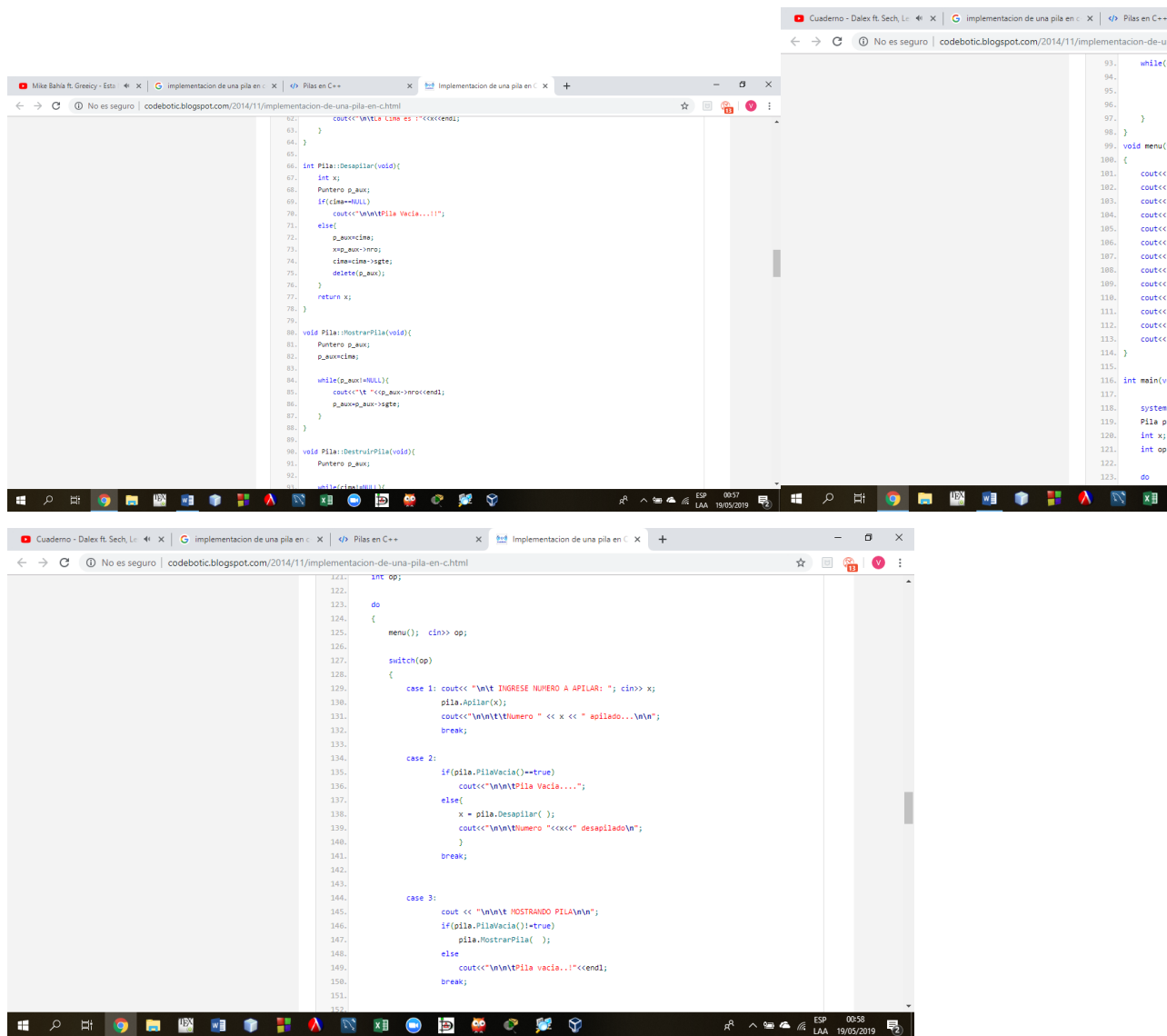
```
211 /* Funcion Principal
212 -----*/
213
214 int main()
215 {
216     Tlista lista = NULL;
217     int op; // opcion del menu
218     int _dato; // elemento a ingresar
219     int pos; // posicion a insertar
220
221     system("color 0b");
222
223     do
224     {
225         menu(); cin>> op;
226
227         switch(op)
228         {
229             case 1:
230                 cout<< "n NUMERO A INSERTAR: "; cin>> _dato;
231                 insertarInicio(lista, _dato);
232                 break;
233
234             case 2:
235                 cout<< "n NUMERO A INSERTAR: "; cin>> _dato;
236                 insertarFin(lista, _dato);
237                 break;
238
239             case 3:
240                 eliminarElemento(lista, _dato);
241                 break;
242
243             case 4:
244                 eliminarRepetidos(lista);
245                 break;
246
247             case 5:
248                 imprimir(lista);
249                 break;
250
251             default:
252                 break;
253         }
254     } while(op != 0);
255 }
```

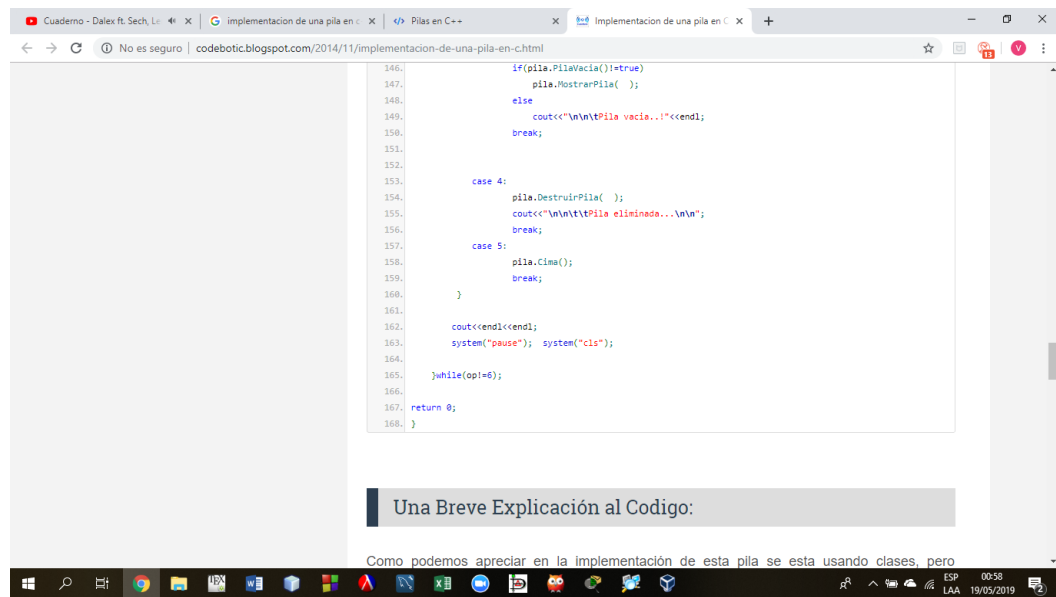
```
238         cout<< "\n NUMERO A INSERTAR: "; cin>> _dato;
239         insertarFinal(lista, _dato);
240     }
241     break;
242
243
244     case 3:
245
246         cout<< "\n NUMERO A INSERTAR: "; cin>> _dato;
247         cout<< " Posición : "; cin>> pos;
248         insertarElemento(lista, _dato, pos);
249     }
250     break;
251
252
253     case 4:
254
255         cout << "\n\n MOSTRANDO LISTA\n\n";
256         reportarLista(lista);
257     }
258     break;
259
260
261     case 5:
262
263         cout<<"\n Valor a buscar: "; cin>> _dato;
264         buscarElemento(lista, _dato);
265     }
266     break;
267
268
269     case 6:
270
271         cout<<"\n Valor a eliminar: "; cin>> _dato;
272         eliminarElemento(lista, _dato);
273     }
274     break;
275
276
277     case 7:
278
279         cout<<"\n Valor repetido a eliminar: "; cin>> _dato;
280         eliminarRepetidos(lista, _dato);
281     }
282     break;
283
284     }
285
286     cout<<endl<<endl;
287     system("pause"); system("cls");
288
289     while(op!=0);
290
291     system("pause");
292     return 0;
293 }
```

```
264
265
266     case 6:
267
268         cout<<"\n Valor a eliminar: "; cin>> _dato;
269         eliminarElemento(lista, _dato);
270     }
271     break;
272
273
274     case 7:
275
276         cout<<"\n Valor repetido a eliminar: "; cin>> _dato;
277         eliminarRepetidos(lista, _dato);
278     }
279     break;
280
281     }
282
283     cout<<endl<<endl;
284     system("pause"); system("cls");
285
286     while(op!=0);
287
288     system("pause");
289     return 0;
290 }
```

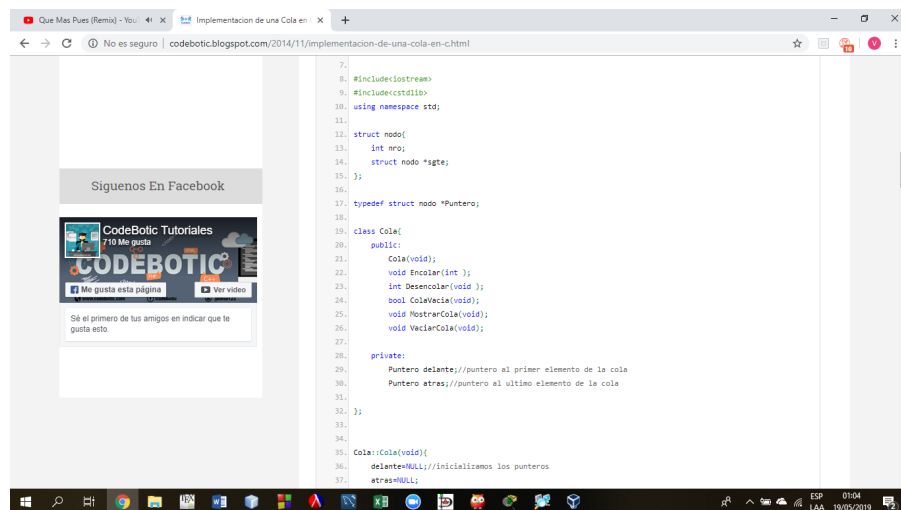
1. Implementación de la lista doblemente enlazada
2. Implementación de la lista circular simplemente enlazada
3. Implementación de la lista circular doblemente enlazada
4. Implementación de una pila

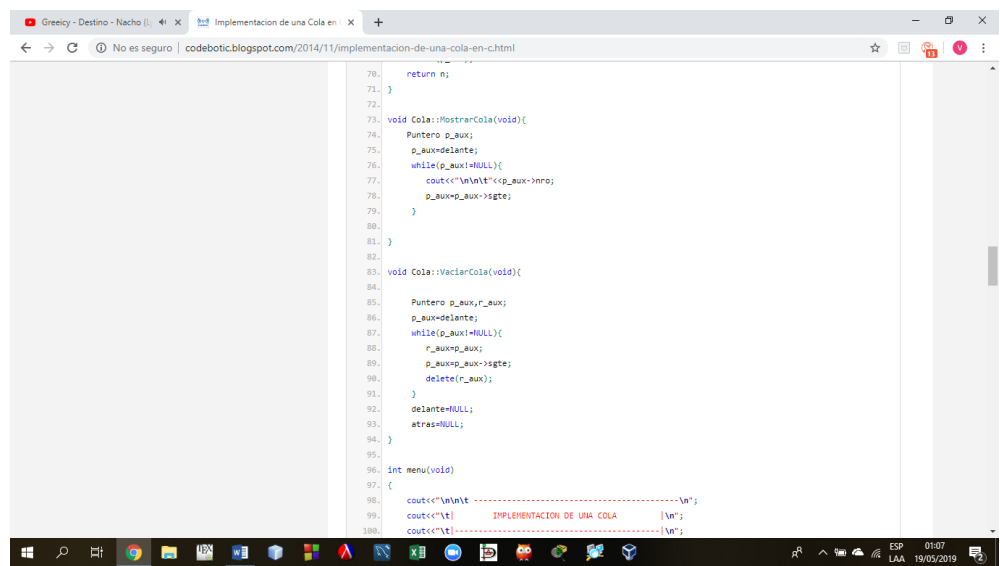


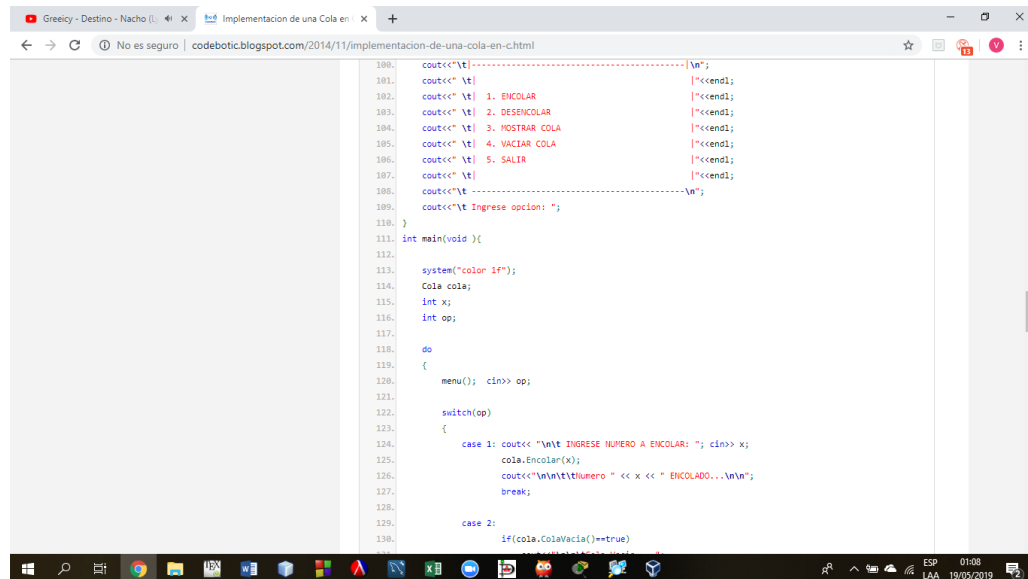




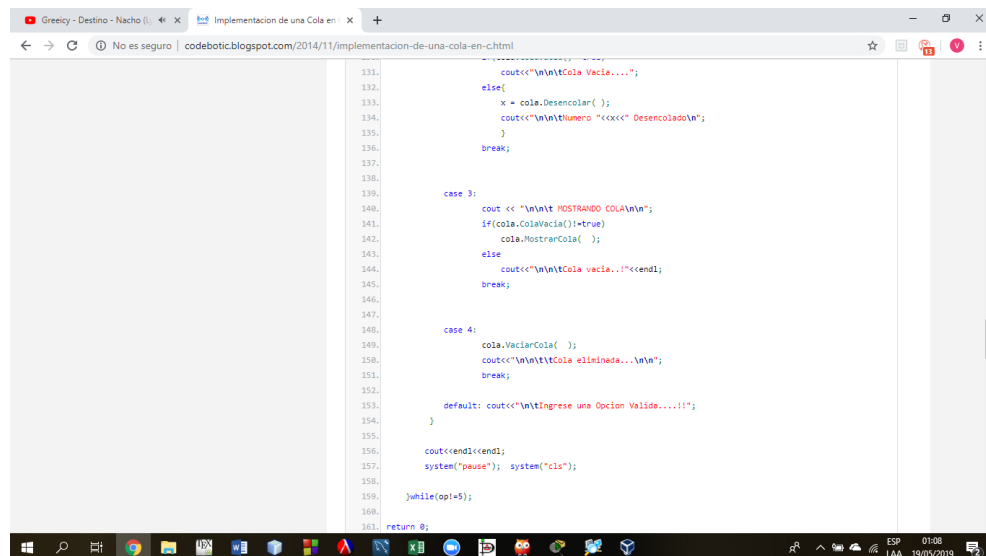
1. Implementación de una cola







```
100. cout<<"\n";
101. cout<<"\t|-----|";
102. cout<<"\t| 1. ENCOLAR";
103. cout<<"\t| 2. DESENCOLAR";
104. cout<<"\t| 3. MOSTRAR COLA";
105. cout<<"\t| 4. VACIAR COLA";
106. cout<<"\t| 5. SALIR";
107. cout<<"\t|-----|";
108. cout<<"\n";
109. cout<<"\t Ingrese opcion: ";
110.
111. int main(void){
112.
113. system("color 1f");
114. Cola cola;
115. int x;
116. int op;
117.
118. do
119. {
120. menu(); cin>> op;
121.
122. switch(op)
123. {
124. case 1: cout<<"\n\t INGRESE NUMERO A ENCOLAR: "; cin>> x;
125. cola.Encolar(x);
126. cout<<"\n\t\tNumero " <<x <<" ENCOLADO...\n\n";
127. break;
128.
129. case 2:
130. if(cola.ColaVacia()==true)
131. cout<<"\n\t\tCola vacia...\n\n";
132. else{
133. x = cola.Desencolar( );
134. cout<<"\n\t\tNumero "<<x<<" Desencolado\n\n";
135. }
136. break;
137.
138. case 3:
139. cout <<"\n\t MOSTRANDO COLA\n\n";
140. if(cola.ColaVacia())!=true)
141. cola.MostrarCola( );
142. else
143. cout<<"\n\t\tCola vacia..!"<<endl;
144. break;
145.
146. case 4:
147. cola.VaciarCola( );
148. cout<<"\n\t\tCola eliminada...\n\n";
149. break;
150.
151. default: cout<<"\n\t Ingrese una Opcion Valida....!!";
152. }
153.
154. cout<<endl<<endl;
155. system("pause"); system("cls");
156. }while(op!=5);
157.
158. return 0;
159. }
```



```
131. cout<<"\n\t\tCola vacia...";
132.
133. else{
134. x = cola.Desencolar( );
135. cout<<"\n\t\tNumero "<<x<<" Desencolado\n\n";
136. }
137. break;
138.
139. case 3:
140. cout <<"\n\t MOSTRANDO COLA\n\n";
141. if(cola.ColaVacia())!=true)
142. cola.MostrarCola( );
143. else
144. cout<<"\n\t\tCola vacia..!"<<endl;
145. break;
146.
147. case 4:
148. cola.VaciarCola( );
149. cout<<"\n\t\tCola eliminada...\n\n";
150. break;
151.
152. default: cout<<"\n\t Ingrese una Opcion Valida....!!";
153. }
154.
155. cout<<endl<<endl;
156. system("pause"); system("cls");
157. }while(op!=5);
158.
159. return 0;
160. }
```

REFERENCIAS BIBLIOGRAFICAS

1. Salvador Martínez Pérez. (2011). El síndrome de Down. Madrid: Consejo Superior de Investigaciones Científicas; Los libros de la Catarata.
2. Libby Kumin. (2014). Síndrome de Down: Habilidades Tempranas de Comunicación. España: Cepe.

3. Juan Perera Mezquida. (2010). Desarrollo de Habilidades Numéricas y Motoras Para Alumnos con Síndrome de Down y acceso a las tecnologías de información. España: Ciencias De La Educación Preescolar Y Especial.
4. Damian Milagros. (2010). Estimulación Temprana Para Niños con Síndrome de Down. México: Editorial Trillas Sa De Cv.
5. Guadalupe Morales Martínez. (2006). El Síndrome de Down y su Mundo Emocional. México: Editorial Trillas Sa De Cv.
6. Artigas López, M. A. L. Mercé. (s.f.). SÍNDROME de DOWN (Trisomia 21).
7. Universidad Internacional de Valencia. (s.f.). La educación de los niños con síndrome de Down.
8. Aponte L. E. (13 de octubre del 2010). Over Blog, recuperado de: <http://programandoenc.over-blog.es/article-listas-en-lenguaje-c-58802346.html>
9. Salvador P. C. (Julio del 2001) C++ con clase, recuperado de: <http://www.conclase.net/c/edd/?cap=003>