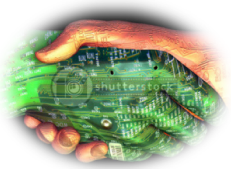
Proyecto MotorTherapy



Instituto Tecnológico de Costa Rica
Área de Ingeniería en Computadores
CE3104 – Lenguaje, Compiladores e Intérpretes
II Semestre 2019

Objetivo general

- ➔ Implementar un dispositivo que ayude a la personas a en sus terapías de motricidad gruesa.

Objetivos específicos

- ➔ Utilización de algún tipo de dispositivo para la ayuda de terapia de motricidad gruesa.
- ➔ Implementación de un lenguaje de programación y su correspondiente Compilador.

Datos Generales

- ➔ El valor del Proyecto: 25%
- ➔ Nombre código: MotorTherapy
- ➔ El proyecto debe ser implementado por grupos de máximo de 5 personas.
- ➔ La fecha de entrega del proyecto es de **9/Noviembre/2019**
- ➔ Cualquier indicio de copia de proyectos será calificado con una nota de 0 y será procesado de acuerdo al reglamento.

Marco Conceptual

El desarrollo del movimiento en las personas puede categorizarse en motricidad gruesa y motricidad fina. La motricidad gruesa tiene que ver con los cambios de posición del cuerpo, los movimientos globales y la capacidad de mantener el equilibrio.

Caminar, correr, saltar, subir, baja. La motricidad gruesa está directamente relacionada con estas y muchas otras habilidades que los niños desarrollan en un periodo de crecimiento fundamental.

La motricidad gruesa es nuestra capacidad para mover los músculos del cuerpo de forma coordinada y mantener el equilibrio, además de la **agilidad**, **fuerza** y **velocidad** necesaria en cada caso. Hace referencia a los movimientos amplios que engloban varios grupos musculares como el control de cabeza, la sedestación, girar sobre sí mismo, gatear, mantenerse de pie, caminar, saltar, etc.

La evolución del área motora sigue dos leyes psicofisiológicas fundamentales: Céfalo-caudal (desde la cabeza hacia los pies) y próximo-distal (desde el eje central del cuerpo hacia las extremidades). Esto supone que las bases principales del desarrollo motor se asentarán sobre la motricidad gruesa y, posteriormente, podrán evolucionar hacia el desarrollo de la motricidad fina.

El desarrollo de la motricidad gruesa en la etapa infantil es de vital importancia para la exploración, el descubrimiento del entorno, la autoestima, la confianza en sí mismo y resulta determinante para el correcto funcionamiento de la psicomotricidad fina más adelante.

Como incluye movimientos musculares de piernas, brazos, cabeza, abdomen, espalda, y además se centra en la habilidad del niño para moverse, desplazarse y conocer el mundo que lo rodea con todos sus sentidos, la motricidad gruesa es un proceso fundamental para procesar y guardar información del entorno y además, un proceso que permite expresar destrezas no sólo físicas, sino cognitivas. Observar el proceso de desarrollo de la motricidad gruesa y trabajar las actividades indicadas es esencial para entender además de las capacidades, las dificultades y sobre todo, los progresos.

Objetivos que persiguen los ejercicios

- Trabajar el esquema corporal favoreciendo definir la lateralidad.
- Favorecer la atención, la concentración y prevenir o reducir dificultades en lectoescritura.
- Mejorar el equilibrio, la coordinación manual, y trabajar la coordinación ojo - mano (Óculo - Manual). Capacidad Viso - espacial.
- Trabajar la coordinación entre ambas piernas y óculo - podal.
- Estimular y ejercitar en general la motricidad gruesa y fina asociando el mundo táctil, el sentido kinestésico y la vista.
- Conseguir la independencia brazo-tronco, es el factor más importante de la precisión en la coordinación óculo-manual. "La mano depende del tronco, del cuerpo, pero no debe estar soldada a él".

https://youtu.be/EFnUQATYYhM
https://youtu.be/vgh88VOABDg
https://youtu.be/OVZQxq699bA
https://youtu.be/u1J7mnJ-uyE

El proyecto solicitado para efectos del curso de “Compiladores e Interpretes” del 2do Semestre 2019 consta de varias cosas. Deben implementar los siguientes 4 juegos:

1. Raqueta Globo

El objetivo de este juego es permitir al niño “golpear” un globo de tal manera que le ayude a nivelar la fuerza del golpe, así como la posición del mismo. Tiene que haber algún tipo de “bate” que permita que niño “jale” al globo y lo pueda golperar. El niño puede golpear el globo varias veces seguidas.

Consideraciones:

- El globo de alguna manera debe “flotar”
- Se debe tener una reacción al esfuerzo que el niño ejerza

2. Usando los pies

Cuando se implementa este juego en físico, se amarra una cuerda de una silla a otra para que quede extendida. Sobre la cuerda, se colocan unas cintas o pequeñas banderas de colores que cuelguen. Una vez hecho esto, se le da a los niños la instrucción de tocar con los pies distintas banderas. Por ejemplo, si se dice: “rosado, amarillo, blanco, rosado”, el niño usará uno de sus pies para tocar las banderas en la secuencia pedida.

https://youtu.be/N1LAOOu-81o

Consideraciones:

- Debe tener cierta cantidad de “banderas” o elementos debidamente identificados
- La distancia entre cada elemento debe ser el adecuado
- Se debe llevar un monitoreo que identifique si el niño siguió correctamente las instrucciones y las veces que falló.
- Si el niño falla en el orden o el golpe con el pie no impacta al elemento se debe explicitamente enviar una señal de fallo.

3. Telarañana

Este juego en fisico se realiza de la siguiente manera: Usando letras de plástico o de papel, cinta adhesiva de color, pequeños pedazos de cartulina y un marcador. Lo primero que hay que hacer es escribir algunas palabras en las cartulinas. Luego se tendrá que hacer la telaraña con la cinta adhesiva y poner en cada intersección de ésta una palabra. El niño debe recorrer la telaraña según el contexto de las palabras escritas. Al final de la telaraña, se ubica el concepto que agrupa a todas las palabras del camino seleccionado. Utilizando el equilibrio, los niños tendrán que caminar sobre las líneas rectas de la telaraña hasta llegar a la palabra y leerla. Después deben recoger las palabras que están distribuidas por la telaraña y llevarlas hasta la palabra escrita, siempre caminando por la líneas. Cuando el niño encuentra todas las palabras y las ubique sobre la cartulina escrita, es el turno de otro niño.

Consideraciones:

- La idea es que el niño pueda mover sus extremidades, no necesariamente tiene que movilizarse en una gran zona. Esto es, por medio de ciertos movimientos de las piernas se puede "simular" un paso hacia delante, hacia atrás, hacia los lados.
- Si el niño pisa fuera de la "linea" se debe explicitamente indicar el error, pero puede proseguir.
- El juego permite ejercicio físico y mental del niño. El niño debe alcanzar cada palabra y finalmente debe descubrir el contexto de todas las palabras recogidas.

4. Alcanzando el objetivo

Se coloca algo a cierta altura, y la persona deberá intentar alcanzarlo. El juego se "pone" interesante haciendo cambios de altura y posición del objeto de tal manera que el niño se sienta retado a alcanzar al objeto.

Consideraciones:

- El elemento se debe colocar a una altura tal que el niño pueda alcanzarlo ya sea saltando o "estirándose" lo más que pueda.
- Se debe contar con un tiempo determinado para realizar la prueba.
- Se debe llevar un conteo de las veces en que se pudo realizar el reto de forma satisfactoria.

Bibliografía usada en el marco teórico:

https://eligeeducar.cl/10-formas-trabajar-la-motricidad-gruesa-los-ninos

http://www.eneso.es/blog/desarrollar-la-motricidad-gruesa/

http://www.psico-vida.com/2014/11/motricidad-psicomotricidad-fina-y-gruesa/

https://youtu.be/iZD3YJRJSYY

https://youtu.be/O18JUKk0kak

https://youtu.be/vgh88VOABDg

https://youtu.be/vQL7-KpR4jw

https://www.youtube.com/watch?v=nj_SG2XYId4

https://www.youtube.com/watch?v=KZHJZiL9CuU

Dispositivo (hardware)

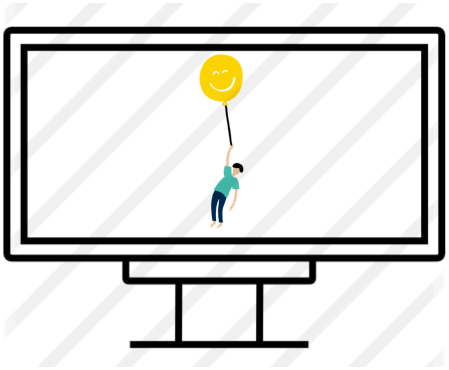
Se requiere que de alguna manera (buscar solución) se puedan identificar los movimientos de las extremidades de una persona en lo que respecta a qué parte de la persona se mueve (brazos, piernas), en qué dirección (adelante, atrás, a los lados) y con cuanta fuerza se realiza. Se tiene que determinar si el niño salta, o se mueve hacia alguna dirección.

Al identificar las anteriores señales, estas deben ser mostradas "inteligentemente" en algún tipo de monitor dentro del cual se presentará un "avatar" o similar mostrando el juego o ejercicio previamente indicado, además se debe incluir los objetos del reto o juego, y su impacto en cada movimiento dado por el niño.

La experiencia del niño debe ser lo más real posible, de tal manera que sus movimientos pueda visualizarlos de alguna forma. Para visualizar el movimiento puede hacer uso del "avatar" y un monitor que represente el juego, o bien, pueden implementar algún tipo de dispositivo de hardware (robot) mostrando los movimientos.

Las consideraciones al respecto son las siguientes:

- Debe ser totalmente inalambrico las conexiones entre el niño y la forma de visualizar el juego o ejercicio.
- De todas maneras se debe contar con algún tipo de software y pantalla que muestre los puntajes y los errores encontrados.
- Debe ser en tiempo real o lo más aproximado posible.
- Cada uno de los 5 juegos deben identificarse plenamente de forma individual.



Compilador de MotorTherapy

Se debe crear un compilador que permita la ejecución de las instrucciones necesarias para llevar a cabo cada uno de los juegos indicados anteriormente.

Para hacer lo anterior, deberán construir una gramática que soporte la funcionalidad que más adelante se indique. El lenguaje construido debe ser flexible desde el punto de vista, que pueda permitir crear distintos programas que parametrizará a los juegos indicados. Puede hacer uso de Lex y Yacc, y sus distintos "sabores" en el mercado.

A continuación se mostrarán las sentencias que deben tomarse en cuenta en el nuevo lenguaje. Aunque dichas sentencias se muestran en cada juego, eso no implica que solamente pertenecen a dicho juego, sino que es parte del "repertorio" completo del lenguaje de programación.

Sintaxis

El programa debe tener la siguiente estructura:

```
Begin
Main
{
  …    // Instrucciones.  Se espera el Menú principal
}

Game1
{
  …    // Instrucciones.
}

Game2
{
  …    // Instrucciones.  }

Game3
{
  …    // Instrucciones.
}

Game4
{
  …    // Instrucciones.
}

End;
```

Si hace falta alguna de estas secciones, se debe enviar un mensaje de error.
Se debe trabajar en un ambiente visual en donde se presenten tanto los movimientos de los objetos programados, como el puntaje y el avance en cada juego.

- Raqueta Globo

Un globo sujeto a una raqueta. El globo consta de dos coordenadas las cuales le indicarán la altura y latitud de donde se encuentra el globo.

La altura toma como base el piso, y la latitud toma como base el lado derecho. Así por ejemplo, se puede tener la siguiente sentencia:

```
Int alt = 2;
Int lat = 4;

Balloon(alt, lat);
```

O bien

```
Balloon(2,4);
```

Cada vez que el niño logra golpear el globo, este debe tener un movimiento que simule dicho impacto. El objeto permanecerá en la posición hasta que el niño pueda golpearlo. Existe una sentencia que puede usar con esto o bien con los otros juegos, y es la siguiente:

```
int cantidad = 5;

Dow(cantidad)

    Balloon(alt, lat);
    Inc(alt, 1);
    Dec(lat, 2);


Enddo;
```

El Dow ejecutará una cantidad determinada de veces el cuerpo de la sentencia. En el ejemplo anterior, se realizará 5 veces. La sentencia Balloom se ejecutará 5 veces, siempre luego de que el niño golpee el globo. En cada loop, se incrementará la variable "alt" en una posición. La sentencia Inc siempre recibirá una variable, y un valor numérico, la cual puede ser también una variable. La sentencia Dec hace lo mismo pero decrementando el valor. Se debe llevar un puntaje cada vez que el niño "acierta" su golpe al globo. Siempre es necesario indicar la posición inicial del globo al iniciar el juego.

Se deben reconocer los golpes que el niño da e intenta dar.

- Usando los pies

Se debe parametrizar los colores y el puntaje en cada uno de estos. La sentencia es:

```
texto(10)  Color[10];
int puntaje[10];

 Color[1] = "Azul";
 puntaje[1] = 10;
```

Donde NombreColor es un texto con el color. Puede ser una variable de texto.
Puntaje es un número o una variable numérica, que indica el premio obtenido si el niño logra patear en el orden indicado.

Debe existir un operador que genere aleatoriamente los colores a ser "pateados", este operador es:

```
Random(Color,Cantidad, tiempo);
```

Esta sentencia tomará el arreglo "Color" y seleccionará un orden en sus índices. Cantidad es la "cantidad" máxima de los índices usados del arreglo que se usará en el juego y que se ordenará. Si la cantidad es mayor al índice máximo del arreglo, se debe generar un error. Este orden se debe guardar de alguna manera, pues es el orden en que se espera que el niño golpee cada color. Tiempo es la cantidad de segundos en que el juego estará activo mientras que el niño no logre llegar al objetivo.

Ej.

```
Random(Color, 3, 60);
```

Toma los tres primeros nodos del arreglo para reordenarlos, y espera a que el niño logré golpear los colores en ese orden para finalizar, o bien, que transcurra un determinado tiempo. Se debe pintar (visualmente) cada color permitiendo al niño poderlo golpear. Debe estar cerca del piso permitiéndole dar el golpe. La cantidad de colores se debe limitar al máximo de 10 colores. Cuando el niño golpea en el orden solicitado, se debe indicar el puntaje según los colores y emitir algún tipo de sonido. Se puede tener la sentencia FOR para ejecutar el juego más de una vez. La iteración se da cada vez que el niño logra golpear en orden los colores.

```
int tiempo = 60;
int cant = 3;

FOR 5 times using Color
   Random(cant, tiempo);
   Inc(cant, 3);
   Dec(tiempo, 10);
FOREND;
```

Observe que este FOR es exclusivo para Random, inclusive la sintaxis de Random cambia del mostrado originalmente. La anterior sentencia ejecuta 5 veces el juego siempre y cuando el tiempo haya transcurrido en cada iteración, o el niño haya acertado en los golpes. En cada iteración se incrementa la cantidad de colores usados en la siguiente iteración, y se decrementa el tiempo. Las dos sentencias anteriores son opcionales y se podrían incluir otras sentencias dentro del cuerpo de este FOR.

Se debe reconocer los golpes que el niño da e intenta dar.

- Telarañana

Se debe tener un arreglo de letras. Para esto se define un arreglo de la siguiente manera:

```
TEXTO(15)  MiArreglo[5];
```

En cada nodo del arreglo se podrá incrustar una palabra:

```
MiArreglo[1] = "Oceano";
MiArreglo[2] = "Rio";
```

Si se utiliza un índice mayor al tamaño del arreglo, se debe mostrar un error. Cada palabra será ubicada en la intersección de la telaraña. La construcción de la telaaraña se podrá hacer similar a la creación de una "tabla" en excel, usando la sentencia:

```
TelaAraña(Fila, Columna);
```

donde Fila y Columna son valores numéricos y podría ser variables.

Para colocar la palabra en cada intersección se colocará las coordenadas de Fila, y Columna. Si se coloca un valor mayor a la de Fila o Columna de la TelaAraña implementada, se debe generar un error. Para realizar esto, se debe usar la sentencia compuesta:

```
ForAsignWord(Fila, Columna) DO
   AsignWord(Palabra, Puntaje);
```

Donde Puntaje, Fila y Columna son valores numéricos y pueden ser variables.
Palabra debe ser un dato dentro del arreglo previamente inicializado.

Así por ejemplo, se tiene:

```
int MiFila = 5;
int MiCol  = 5;

Texto(10) MiArr[25];
MiArr[1] = "Azul";
MiArr[2] = "Rojo";
```

```
...


int MiPuntaje[25];
MiPuntaje[1] = 10;
MiPuntaje[2] = 15;
....

TelaAraña(MiFila, MiCol);


 ForAsignWord(MiFila, MiCol) DO
    AsignWord(MiArr, MiPuntaje);
```

Así de esa manera, se asignará a la intersección 1,1 la palabra "Azul" y el puntaje 10. Y así sucesivamente. Cada vez que el niño llega a una intersección aparecerá el premio obtenido, y se deberá llevar visualmente las palabras encontradas. Cuando el niño recorre la TelaAraña completamente, se le solicitará al niño que indiqué el concepto que agrupa a todas las palabras recogidas, y posteriormente se le presentará el concepto y el puntaje obtenido.

Si el niño se sale de la línea de la TelaAraña, debe audiblemente indicar la situación.

- Alcanzando el Objetivo

Debe existir un objeto (visualmente agradable a la vista) flotando a cierta altura. El niño debe estirarse lo más que pueda para alcanzarlo. Se debe parametrizar el objeto, para tal caso, se colocará una distancia determinada en el aire, y cada cierto tiempo, moverlo. Esto se realizará usando el siguiente comando:

```
Object(Altura, longitud, Second)
```

Donde Altura es la distancia del suelo, y es un valor numérico, puede ser una variable.
Second es un valor numérico (puede ser una variable) e indica la cantidad de segundos que el objeto se mantendrá en la posición actual.

Longitud es la distancia que hay desde costado derecho. Puede ser un número o una variable.

Ejemplo:

```
Object(2,1, 25);
```

Esto significa que el objeto estará presente durante 25 segundos a una altura de 2 metros y una longitud de 1 metro a partir del costado derecho.

Se puede hacer uso de la sentencia FOR para variar este juego, de la siguiente manera:

```
int Dist[5];

Dist[1] = 2;
Dist[2] = 5;
Dist[3] = 1;
Dist[4] = 7;
Dist[5] = 1;

FOR  5  Times using Var1
   Object(2, Dist[Var1], 25)

FEnd;
```

A una altura de 2 metros, y usando la longitud del arreglo Dist, permanecerá el objeto en ese sitio durante 25 segundos. Se pueden utilizar otras sentencias del lenguaje dentro de la sentencia del FOR. Si el arreglo de longitud se accede por un índice fuera de su rango, se debe enviar un error semántico. En el momento en que el niño alcance tocar al objeto, se debe visualmente indicar que se ha superado el juego.

Se debe presentar visualmente un Menú con los 4 juegos anteriores. El niño de alguna manera, ya sea golpeando con su mano, o señalando escogerá uno de los juegos y se iniciará el mismo. Cada vez que finaliza un juego se deberá volver a este Menú. Debe haber una opción de salir.

Elementos básicos del Proyecto en la revisión:

1. Escritura del programa usando la gramática creado por ustedes y según los requerimientos dados en el presente documento. El programa usará en la sintaxis los comandos necesarios para determinar el comportamiento del dispositivo y las acciones que deberá hacer.

2. Implementar un "compilador" que tome el programa según la gramática generada y convierta dicha fuente en código que será enviado al dispositivo o de alguna manera se genere el comportamiento del dispositivo deseado.

3. Se envía el código generado vía inalambrica directamente al dispositivo de tal manera que inicie las actividades "programadas".

Otros aspectos de la Sintaxis básica:

- Se debe respetar la sintaxis previamente ejemplificada, pero el grupo puede "enriquecer" dicha gramática añadiendo las sentencias o comportamiento que consideren necesario, pero siempre deben respetar lo colocado en el presente documento. Además se deben tomar en cuenta las siguientes instrucciones:
    - Las variables usadas en el programa fuente deben tener un máximo de 10 posiciones. Debe iniciar siempre con una letra minúscula, las demás letras pueden ser minúsculas o mayúsculas, y solamente deberá permitir letras, números, y los símbolos especiales "_", "&", "-" y "@". Todo programa debe contener al menos una variable. De lo contrario debe **generar un error**.
    - Los comentarios en el código fuente es por línea, y siempre debe iniciar con // y se comenta toda la línea, ejemplo:

      // Esto es un comentario

      Todo código fuente debe tener al menos un comentario indicando el nombre y la funcionalidad del código. De lo contrario **debe generar un error.** Los comentarios pueden estar presentes en cualquier lugar dentro del código del programa, pero siempre debe existir al menos uno en la primera línea de este.
    - Las instrucciones se delimitarán con un punto y coma (;). Si no se encuentra debe **generar un error**.
    - En una misma línea se pueden colocar más de una instrucción delimitadas cada una con un punto y coma.
    - Se deben realizar todas las validaciones pertinentes como a nivel léxico, sintáctico, semántico, etc.

Para la solución del problema del proyecto presentado se espera del estudiante:

- Fuerte investigación sobre las posibles soluciones en las distintas etapas del problema
- Busqueda de distintas fuentes que servirán de insumo técnico-práctico para las soluciones así como la ayuda de otras áreas para cumplir con el objetivo del proyecto.
- Selección de criterios acordes al nivel de nuestro ambiente Tec en donde se seleccionen las mejores soluciones correspondientes.
- Una solución solvente de acuerdo a lo esperado.

Documentación y aspectos operativos

➔ Se deberá entregar un documento que contenga:

- ✓ Diagrama de arquitectura de la solución que refleje un nivel de detalle suficiente para entender a términos generales el funcionamiento del proyecto. Deben investigar cómo se hace un diagrama de arquitectura general a nivel profesional.
- ✓ Alternativas de solución consideradas y justificación de la seleccionada.
- ✓ Problemas conocidos: En esta sección se detalla cualquier problema que no se ha podido

solucionar en el trabajo.

- ✓ Actividades realizadas por estudiante: Este es un resumen de las bitácoras de cada estudiante (estilo timesheet) en términos del tiempo invertido para una actividad específica que impactó directamente el desarrollo del trabajo, de manera breve (no más de una línea) se describe lo que se realizó, la cantidad de horas invertidas y la fecha en la que se realizó. Se deben sumar las horas invertidas por cada estudiante, sean conscientes a la hora de realizar esto el profesor determinará si los reportes están acordes al producto entregado.
- ✓ Problemas encontrados: descripción detallada, intentos de solución sin éxito, soluciones encontradas con su descripción detallada, recomendaciones, conclusiones y bibliografía consultada para este problema específico.
- ✓ Conclusiones y Recomendaciones del proyecto. Conclusiones y recomendaciones con sentido y que confirmen una experiencia profunda en el proyecto.
- ✓ Bibliografía consultada en todo el proyecto

Atributos

Se debe **entregar un documento aparte** que contenga una portada y el siguiente detalle.

Debe comentar de qué manera se aplicaron los atributos que posee el curso y su impacto, esto por medio de una tabla en donde se detalle para cada atributo lo siguiente:

- Aplicación del atributo en la solución del proyecto
- Impacto del proyecto en la sociedad
- Retroalimentación obtenida gracias al proyecto

Favor colocar la información antes solicitada para cada uno de los atributos siguientes:

- **Conocimiento base de ingeniería**
- **Impacto de la ingeniería en la sociedad y el medio ambiente**
- **Aprendizaje continuo**

Conocimiento de ingeniería
Capacidad para aplicar los conocimientos a nivel universitario de matemáticas, ciencias naturales, fundamentos de ingeniería y conocimientos especializados de ingeniería para la solución de problemas complejos de ingeniería.

Medio ambiente y Sostenibilidad
Capacidad para comprender y evaluar la sostenibilidad y el impacto del trabajo profesional de ingeniería, en la solución de problemas complejos de ingeniería en los contextos sociales y ambientales.

Aprendizaje Continuo
Capacidad para reconocer las necesidades propias de aprendizaje y la hablidad de vincularse en un proceso de aprendizaje independiente durante toda la vida, en un contexto de amplio cambio tecnológico.

Evaluación

1. La implementación (funcionalidad a nivel del dispositivo) corresponde a un 45% de la nota final del proyecto.
2. El Compilador corresponde a un 45% de la nota final del proyecto y su comunicación con el dispositivo.
3. La documentación tendrá un valor de un 10% de la nota final del proyecto, cumplir con los puntos especificados en la documentación no significa que se tienen todos los puntos.
4. Cada grupo recibirá una nota en cada uno de los siguientes apartados:
    a. Funcionalidad
    b. Compilador
    c. Documentación
5. De las notas mencionadas en el punto anterior se calculará la Nota Final del Proyecto.
6. No debe imprimirse la documentación, y esta debe encontrarse en formato PDF.
7. Documentación digital:
    a. Incluya dentro de su documentación todo el código de programación generado
8. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
9. El profesor llevara un listado de todos y cada una de los requerimientos mencionados en el presente enunciado del proyecto, de tal manera que se pueda evaluar cada uno de ellos y dar el puntaje determinado.
10. El grupo debe contar con un grupo de procedimientos definidos previamente que demuestre los avances en la ejecución del proyecto. Estos procedimientos deben cumplir con las reglas indicadas previamente en este documento.
11. El profesor llevará a la revisión un programa ejemplo que cumple con toda la sintaxis básica indicada en el presente documento y que tendrá su puntaje en la revisión.
12. El grupo deberá demostrar la funcionalidad del dispositivo llevando los elementos necesarios para que el dispositivo cumpla con su función.
13. El grupo deberá presentar un programa fuente creado por ellos mismos en los cuales recreen el emulador según las pautas y sintaxis antes indicada.
14. Dentro de la revisión del proyecto se deberá presentar la gramática creada y dar una explicación detallada de la manera en que se implementó el intérprete mostrando todos los elementos necesarios para entender su funcionalidad.
15. Todos los integrantes del grupo deberán estar presente en la defensa del proyecto, y todos deben estar preparados para contestar las preguntas que se les puede realizar en base a la implementación

realizada. Solamente en casos justificados y vistos previamente con por lo menos un día de anticipación se podrá ausentar algún miembro del equipo.

16. Aun cuando el código y la documentación tienen sus notas por separado, se aplican las siguientes restricciones
    a. Si no se entrega documentación, automáticamente se obtiene una nota de 0.
    b. Si la documentación no incluye el diagrama de arquitectura se obtiene una nota de 0
    c. Sí el código y la documentación no se entregan en la fecha indicada se obtiene una nota de 0.
    d. Sí el código no compila se obtendrá una nota de 0, por lo cual se recomienda realizar la defensa con un código funcional.
17. La revisión de la documentación será realizada por parte del profesor, no durante la defensa del proyecto. El único requerimiento que se consultará durante la defensa del proyecto es el diagrama de arquitectura, documentación interna y cualquier otra documentación que el profesor considere necesaria.
18. Cada grupo tendrá como máximo 20 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa, y será responsabilidad del grupo administrar el tiempo dispuesto para su revisión de tal manera que el profesor pueda observar todo el producto terminado.
19. Todo error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
20. Cada grupo es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberán avisar al menos 2 días antes de la revisión a el profesor para coordinar el préstamo de estos.
21. No se revisarán funcionalidades incompletas o no integradas.
22. Durante la revisión únicamente podrán participar los miembros del grupo, asistentes, otros profesores y el coordinador del área.