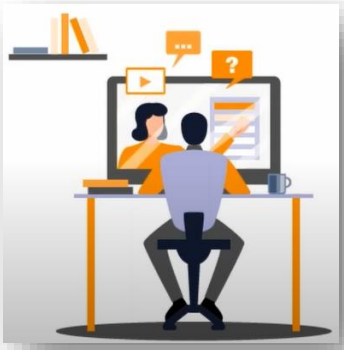
PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE
VideoJuegos (T2)

IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

- Denominación del Programa de Formación: Desarrollo de videojuegos y Entornos interactivos
- Código del Programa de Formación: 228108
- Nombre del Proyecto DESARROLLO DE EXPERIENCIAS INTERACTIVAS PARA EL ENTORNO PRODUCTIVO
- Fase del Proyecto PRODUCCIÓN
- Actividad de Proyecto ESTRUCTURAR EL CÓDIGO DEL VIDEOJUEGO
- Competencia
  - Producción del código del videojuego
- Resultados de Aprendizaje Alcanzar:
  - RAP 43 DEFINIR LA ESTRUCTURA DEL CÓDIGO DEL VIDEOJUEGO DE ACUERDO CON EL DOCUMENTO DE DISEÑO.
- Duración de la Guía 110 Horas



2. PRESENTACIÓN

A partir de este momento inicia con la programación de aplicaciones en un lenguaje específico, actividad muy importante y determinante en su proyecto que constituye una etapa del ciclo de desarrollo de Videojuegos.

Considerando la importancia que tiene la codificación en la fase de construcción del videojuego, se inicia con esta actividad de programación, cuyo objetivo principal es acercarlo aún más al lenguaje de programación C#, presentando elementos del lenguaje y el manejo de estructuras de datos que son indispensables para el dominio de entornos de desarrollo de videojuegos como UNITY.

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

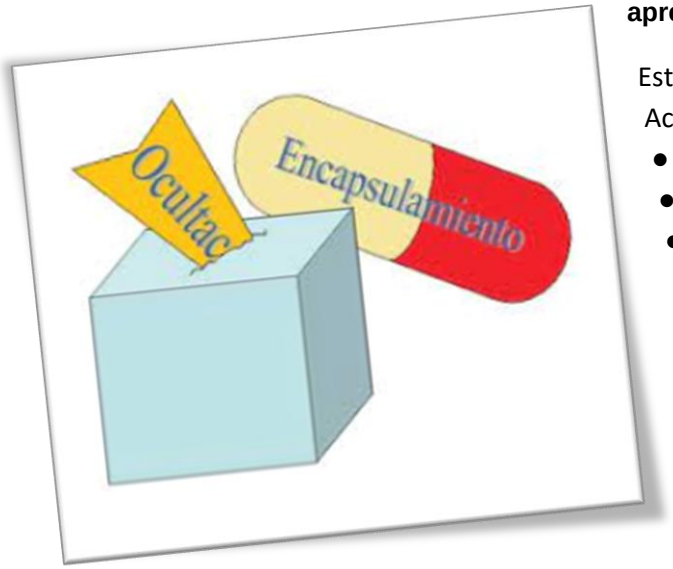
3.1 Actividades de Reflexión inicial.

De su opinión sobre:

- ¿Por qué debería aprender a programar en C# y que ventajas y desventajas tiene en comparación con otros lenguajes (mínimo 4 lenguajes)?.
- ¿Por qué los algoritmos son la base para los lenguajes de programación?
- Haga un resumen del video "Todo el mundo debería saber programar"

Debata educadamente y respetuosamente los comentarios de mínimo dos compañeros.

3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje.



Esta guía de aprendizaje consta de cuatro actividades:

Actividad No 1:

- Conceptos Básicos
- Variables – Tipos de Datos
- Operadores
- Math

Actividad No 2:

- Condicionales
- Ciclos
- Vectores
- Matrices

Actividad No 3:

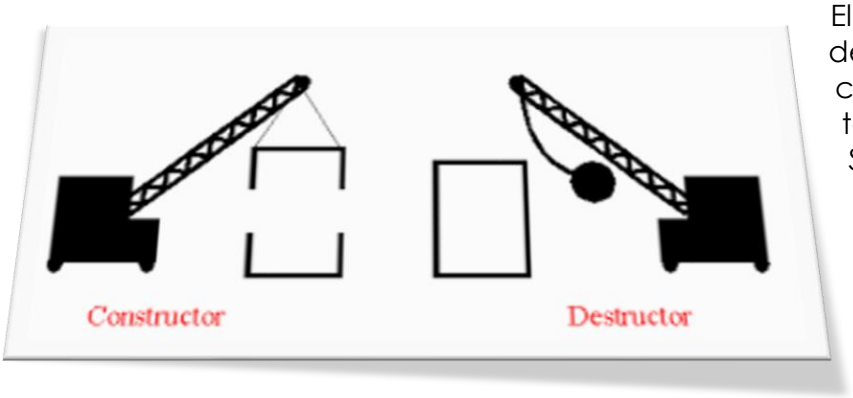
- ¿Qué es una clase en programación orientada a objetos?
- Características de una clase
- This
- Constructores
- Sobrecarga de constructores
- Métodos
- Métodos void - Sin retorno
- Métodos de tipo – Con retorno
- Modificadores de acceso
- Sobrecarga de métodos
- ¿Qué es un objeto en programación orientada a objetos?
- Identidad
- Comportamiento
- Estado
- Herencia
- Sobreescritura de Métodos - Polimorfismo
- Clases Abstractas
- Interfaces en C#

Actividad No 4:

- Estructuras de Datos
- Listas
- Pilas

- Colas
- Métodos de Ordenamiento
- Recursividad

3.2 Actividades de apropiación del conocimiento.



El instructor presenta el contenido de la actividad y relacionado los conceptos previos que deben tenerse en cuenta.
Se revisa cada uno de estos conceptos con los aprendices y su pertinencia con el contenido mostrado.

También los videos:

- ¿Qué es un algoritmo?
- Todo mundo debería saber programar (español)

Para la segunda actividad es necesario hacer las lecturas:

- CONDICIONALES Cap. 3 Pag. 72
- SENTENCIAS REPETITIVAS Cap. 4 Pag. 112
- VECTORES Y MATRICES Cap. 6 y 7 Pag. 188

Los videos:

- Condicionales
- Ciclo DO-WHILE
- Ciclo FOR
- Vectores
- Matrices

En la actividad 3 es necesario las lecturas:

- OBJETOS Y CLASES Cap. 10 Pag. 318
- MÉTODOS Cap. 5 Pag. 148
- PARÁMETROS EN LOS MÉTODOS Cap. 10 Pag. 318
- CONSTRUCTORES
- RELACIONES ENTRE OBJETOS
- HERENCIA Cap. 10 Pag 319
- INTERFACES

Los videos:

- Constructores
- Herencia
- Sobrescritura

- 16- Clase abstracta
- 17- Polimorfismo
- 18- Interfaces

Por último, la actividad No 4, realizar las lecturas:

- LISTAS, PILAS Y COLAS
- METODOS DE ORDENAMIENTO
- RECURSIVIDAD

3.3 Actividades de apropiación del conocimiento (Conceptualización y Teorización).

Realizar los talleres:

- Taller Estructuras de datos en C#.
- Taller Introducción a la POO en C#.
- Taller sobre clases abstractas
- Taller Interfaces

4. ACTIVIDADES DE EVALUACIÓN

Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Evidencias de Conocimiento : • **Cuestionario No 1. Conceptos básicos.** • **Cuestionario No 2. Condicionales, Ciclos, Vectores y matrices** • **Cuestionario No 3. POO en C#** **Evidencias de Desempeño** • **Realiza el taller: ESTRUCTURAS DE DATOS EN C#** • **Realiza el taller: TALLER: CLASES ABSTRACTAS** • **Realiza el taller: TALLER: INTERFACES** **Evidencias de Producto:** • **Desarrolla Caso de estudio propuesto por el instructor aplicando POO**	**UTILIZA LAS HERRAMIENTAS DE DESARROLLO, PARA LA CODIFICACIÓN DE LOS MÓDULOS DEL SISTEMA, APLICANDO LAS FUNCIONES PROPIAS DEL LENGUAJE DE PROGRAMACIÓN SELECCIONADO, DE ACUERDO CON LAS NECESIDADES DEL SISTEMA DE INFORMACIÓN.**	**Cuestionario** **Cuestionario** **Cuestionario** **Taller** **Taller** **Taller** **Lista de chequeo**

5. GLOSARIO DE TÉRMINOS

- **Modelo algorítmico: Análisis, diseño e implementación de un software usando objetos de "software".**
- **Objetos de software : Componentes que integran o conforman el modelo; pueden ser unidades de código para resolver situaciones específicas, shorts, uso de DB, prints, funciones, vectores, etc.**
- **Modelo mecánico: Análisis, diseño e implementación de prototipo a escala de un sistema físico usando objetos concretos.**
- **Objetos concretos: Partes físicas del modelo mecánico, ojo del modelo, no del sistema real; o sea, los objetos planeta no son los planetas reales.**
- **Modelo: Es una vista de un sistema del mundo real, es decir, una abstracción de dicho sistema considerando un cierto propósito. Así, el modelo describe completamente aquellos aspectos del sistema que son relevantes al propósito del modelo y a un apropiado nivel de detalle.**
- **Diagrama: Representación gráfica de un modelo.**
- **Abstraction: Propiedad y/o técnica de software que oculta los detalles de la implementación. Java soporta abstracción de clases y abstracción de métodos. La abstracción de métodos se define separando el uso de un método sin conocer cómo está implementado ese método. Si decide combinar la implementación, el programa cliente será afectado. De modo similar la abstracción de clases oculta la implementación de la clase del cliente.**
- **Algoritmo: Método que describe cómo se resuelve un problema en término de las acciones que se ejecutan y especifica el orden en que se ejecutan estas acciones. Los algoritmos ayudan al programador a planificar un programa antes de su escritura en un lenguaje de programación.**
- **Análisis: Proceso de identificación, modelado y descripción de lo que hace un sistema y de cómo trabaja.**
- **Aplicación: Programa autónomo Java tal como cualquier programa escrito utilizando un lenguaje de alto nivel. Las aplicaciones se pueden ejecutar desde cualquier computadora con un intérprete Java. Las aplicaciones no están sometidas a las restricciones impuestas los applets de Java. Una clase aplicación debe contener un método main. Se utiliza como sinónimo de programa.**
- **Argumento : Información pasada a un método. Los argumentos se suelen llamar también parámetros. Un método que espera recibir argumentos debe contener una declaración de argumentos formales por cada argumento actual como parte de la cabecera del mismo. Cuando se invoca a un método, los valores de los argumentos actuales reales) se copia en los correspondientes argumentos formales. Véase parámetro actual (actual parameter).**
- **Asignación: Almacenamiento de un valor en una variable. La sentencia de asignación es aquella que implementa la asignación y utiliza un operador de asignación.**
- **Biblioteca de clases: Colección organizada de clases que proporciona un conjunto de componentes y abstracciones reutilizables.**
- **Binario: Representación numérica en base 2. En esta base sólo se utilizan los dígitos 0 y 1. Las posiciones de los dígitos representan potencias sucesivas de 2. Véase bit.**
- **Boolean: Tipos primitivos de datos en Java. El tipo boolean puede tomar sólo dos valores: true (verdadero) y false (falso).**
- **Clase: Colección encapsulada de datos y operaciones que actúan sobre los datos. El concepto de clase es fundamental en programación orientada a objetos. Una clase consta de métodos y**

datos. Los métodos de una clase definen el conjunto de operaciones permitidas sobre los datos de una clase (sus atributos). Una clase puede tener muchas instancia de la clase u objetos.

- Clase abstracta: Superclase que contiene características comunes compartidas por las subclases. Se declaran utilizando la palabra reservada abstract. Las clases abstractas pueden contener datos y métodos, pero no se pueden instanciar (crear objetos); es decir, no se pueden crear objetos de esta clase.
- Clase cliente: Clase que hace uso de otra clase.
- Clase concreta: Una clase diseñada para crear (tener) instancias de objetos
- Clase hija: Véase subclase.
- Clase interna: Una clase interna es una clase empotrada en otra clase. Las clases internas permiten definir pequeños objetos auxiliares y unidades de comportamiento que hacen a los programas más simples y concisos.
- Clase miembro: Término general utilizado para describir una clase declarada dentro de otra declaración de clases.
- Comentario: Trozo de texto que tienen como objetivo documentar el programa y mostrar como se ha construido. Los comentarios no son sentencias de programación y son ignorados por el compilador. En Java los comentarios están precedidos por dos barras (//) en una línea o encerrados entre /+ y */ en múltiples líneas.
- Compilación: Proceso de traducción de un lenguaje de programación. Normalmente este proceso implica la traducción de un lenguaje de programación de alto nivel a lenguaje de programación de bajo nivel, o el formato binario de un conjunto de instrucciones específicas. La traducción se realiza con un programa denominado compilador. Un compilador java traduce los programas en bytecodes.
- Compilador: Programa de software que realiza un proceso de compilación (traducción del lenguaje fuente a lenguaje máquina) de un programa escrito en un lenguaje de programación de alto nivel. En el caso de Java, es un programa que traduce el código fuente Java en bytecode. El compilador de J2SDK se denomina javac.
- Compilador en tiempo de ejecución: Compilador capaz de compilar cada bytecode de una vez, y a continuación se reinicia al código compilado repetidamente cuando se ejecuta el bytecode.
- Constante: Una variable declarada en final en Java. Una constante de la clase normalmente está compartida por todos los objetos de la misma clase; por consiguiente, una constante de clase se declara normalmente como static. Una constante local es una constante declarada dentro de un método.
- Constante de la clase: Variable definida como final y static.
- Constructor: Método especial utilizado para inicializar el estado de un nuevo objeto. El constructor permite crear objetos utilizando el operador new. El constructor tiene exactamente el mismo nombre que la clase que lo contiene. Los constructores se pueden sobrecargar con el objetivo de facilitar la construcción de objetos con diferentes tipos de valores iniciales.
- Declaración: Define las variables, métodos y clases en un programa.
- Definición: Término sinónimo de declaración, aunque en el proceso de escritura de un programa se suele diferenciar.
- Depuración: Proceso de encontrar, fijar y eliminar errores en un programa. Para estas tareas se suele utilizar una herramienta de programación conocida como depurador.
- Depurador: Herramienta para ayudar a la localización de errores de un programa: jdbc se proporciona como parte del J2SDK. Un depurador puede establecer puntos de interrupción (breakpoint), parada simple a través de un programa e inspecciona el estado de las variables.

- **Diagrama de clases: Una representación gráfica construida utilizando una notación formal para visualizar y documentar las relaciones entre clases de un sistema.**
- **Diseño: Actividad de definir como se debe estructurar e implementar un programa.**
- **Encapsulación: Localización y protección de las características internas y estructura de un objeto. Combinación de métodos y datos en una única estructura de datos. En Java se conoce como clase.**
- **Entero: Un número completo (no es un número real con coma decimal) tal como –5, 1, 10 y 2002. Los enteros se pueden representar en Java de dos formas: utilizando el tipo primitivo int o utilizando una instancia de una clase integer.**
- **Excepción: Un suceso (evento) no previsto que indica que un programa ha fallado en alguna forma. Las excepciones se representan por objetos excepción en java. Las excepciones se manejan con un bloque de sentencias try/catch.**
- **Expresión: (expresión) Una subparte de una sentencia que representa un valor. Por ejemplo, la expresión aritmética ´2+5´ representa el valor 7. En Java, cualquier construcción sintáctica legal que represente un valor es una expresión.**
- **Expresión booleana , lógica: (Boolean expresión) Una expresión cuyo resultado es del tipo lógico (boolean), Operadores tales como && y || toman operandos lógicos y producen un resultado lógico. Los operadores relacionales toman operandos de tipos diferentes y producen un resultado lógico.**
- **Expresión booleana , lógica: (Boolean expresión) Una expresión cuyo resultado es del tipo lógico (boolean, bol), Operadores tales como && y || toman operandos lógicos y producen un resultado lógico. Los operadores relacionales toman operandos de tipos diferentes y producen un resultado lógico.**
- **Función: (function) Construcción matemática a la que se pueden aplicar valores y que devuelve un resultado.**
- **Herencia: (inheritance) Una relación entre clases en que una subclase se extiende desde una superclase.**
- **Implementación: (implementation) La actividad de escribir, compilar, probar y depurar el código de un programa.**
- **Instancia: (instance) Objeto de una clase Instanciación (instantiation) Proceso de creación de un objeto de una clase.**
- **Instanciación: (instantion) Proceso de crear un objeto de una clase.**
- **Interfaz: (interface) Una interfaz se trata como una clase especial. Cada interfaz se compila en un archivo independiente de bytecode, tal como una clase ordinaria. No se puede crear una instancia de la interfaz. La estructura de una interfaz Java es similar al de una clase abstracta en la que se puede tener datos y métodos. Los datos, sin embargo, deben ser constantes y los métodos pueden tener sólo declaraciones sin implementación. En NET existe sólo herencia simple y una clase puede heredar de una superclase. Esta restricción se puede superar por el uso de una interfaz.**
- **Manejador de sucesos: (eventhandler) Un método en el que el objeto "oyente" se ha diseñado para hacer algún proceso especificado cuando ocurre un suceso determinado.**
- **Método: (method) Una colección de sentencias que se agrupan juntos para ejecutar una operación. Method object**
- **Método abstracto: (abstract method) Método que sólo tiene signatura y no tiene cuerpo, y debe estar contenido dentro de una clase abstracta. Su implementación se realiza en la subclase. Se representa mediante el modificador abstract. Los métodos abstractos deben implementarse en una subclase no abstracta incluso aunque no se utilicen.**

- **Método de la clase: (classmethod) Sinónimo de método estático. Un método que se puede invocar sin crear una instancia de la clase. Para definir métodos de clases, se ha de poner un modificador static en la declaración del método.**
- **Método de la instancia: (Instance method) Un método (o procedimiento)declarado por un clase que se llama por sus objetos de instancias (o los de las subclases).**
- **Objeto instancia: (instance object) Un objeto instancia es un representación de un valor del tipo implementado por su clase. La clase declara un objeto de variables, instancia que forman la estructura de un objeto y un conjunto de métodos que se pueden llamar en un objeto.**
- **Parámetro actual o real: (actual parameter) Valor que se pasa a un método cuando se invoca este método. Los parámetros reales (actuales) deben concordar en tipo, orden y número con los parámetros formales. Cuando se invoca a un método, los valores de los argumentos actuales se copian en los correspondientes argumentos formales.**
- **Público (public) Un modificador de clases, datos y métodos a los que se puede acceder por todos los programas.**
- **Suceso: (event) Un tipo de señal que indica ha ocurrido alguna acción. Normalmente se asocia con sucesos de entrada de interfaces gráficas de usuario (p.e. el "clic" de un ratón, pulsación de una tecla, etc.) El programa puede responder o ignorar el suceso. Véase evento.**
- **Subclase (sub class) Una clase que hereda o se extiende de una superclase.**
- **Superclase (super class) Una clase que puede ser heredada de otra clase.**
- **Clase Principal: (main class)Una clase que contiene un método principal (main).**

6. REFERENTES BILBIOGRÁFICOS

- Programación orientada a objetos con C++ (4a. ed.), Ceballos Sierra, Francisco Javier, RA-MA Editorial, 2007
- Desarrolle Aplicaciones Windows con Visual Studio 2019, JÉRÔMe Hugon, Eni.
- C# Guia Total del programador, Nicolás Arrioja Landa Cosio,Editorial USERS.
- WebGrafia: https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/
- Borja, Orbegozo Arana. Desarrollo de aplicaciones c# con Visual Studio .Net. Curso práctico. Editorial Alfaomega. 2015.
- Ceballos, Francisco Javier. Microsoft c#. Curso de programación.
- Ceballos, Francisco Javier. Interfaces gráficas y aplicaciones para internet con WPF, WCF y Silverlight.
- Introducción a C#: https://docs.microsoft.com/es-es/dotnet/csharp/tutorials/intro-to-csharp/

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	**José Fernando Galindo S.**	**INSTRUCTOR**	**TELEINFORMATICA**	**18/04/2022**

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					